Provably Mitigating Overoptimization in RLHF: Your SFT Loss is Implicitly an Adversarial Regularizer

Zhihan Liu[1*] **Miao Lu**[2*] **Shenao Zhang**[1] **Boyi Liu**[3]
Hongyi Guo[1] **Yingxiang Yang**[3] **Jose Blanchet**[2] **Zhaoran Wang**[1]
[1]Northwestern University [2]Stanford University [3]ByteDance Inc.
{zhihanliu2027, shenaozhang2028, hongyiguo2025}@u.northwestern.edu
{miaolu, jose.blanchet}@stanford.edu, zhaoranwang@gmail.com
{boyi.liu01, yingxiang.yang}@bytedance.com ([*]Equal contributions)

Abstract

Aligning generative models with human preference via RLHF typically suffers from overoptimization, where an imperfectly learned reward model can misguide the generative model to output undesired responses. We investigate this problem in a principled manner by identifying the source of the misalignment as a form of distributional shift and uncertainty in learning human preferences. To mitigate overoptimization, we first propose a theoretical algorithm that chooses the best policy for an adversarially chosen reward model; one that simultaneously minimizes the maximum likelihood estimation of the loss and a reward penalty term. The penalty term is introduced to prevent the policy from choosing actions with spurious high proxy rewards, resulting in provable sample efficiency of the algorithm under a *partial coverage* style condition. Moving from theory to practice, the proposed algorithm further enjoys an equivalent but surprisingly easy-to-implement reformulation. Using the equivalence between reward models and the corresponding optimal policy, the algorithm features a simple objective that combines: (i) a preference optimization loss that directly aligns the policy with human preference, and (ii) a supervised learning loss that explicitly imitates the policy with a (suitable) baseline distribution. In the context of aligning large language models (LLM), this objective fuses the direct preference optimization (DPO) loss with the supervised fine-tuning (SFT) loss to help mitigate the overoptimization towards undesired responses, for which we name the algorithm Regularized Preference Optimization (RPO). Experiments of aligning LLMs demonstrate the improved performance of RPO compared with DPO baselines. Our work sheds light on the interplay between preference optimization and SFT in tuning LLMs with both theoretical guarantees and empirical evidence.

1 Introduction

A key step in building state-of-the-art LLMs is Reinforcement Learning from Human Feedback (RLHF) [10, 58], which aligns pretrained LLMs with human preferences using human assessment data, making the model more helpful, truthful, and harmless [30, 8]. Typically, RLHF first learns a reward model from data (pair-wise comparisons of responses) to quantify the human preferences of LLM outputs. Then it fine-tunes the LLM to maximize the learned reward using RL techniques.

In this pipeline, a crucial challenge is *reward overoptimization* or *reward hacking* [28, 41, 19]. Since the reward model is learned from finite data, it might not be perfectly aligned with the underlying human preference. Optimizing the LLM towards such an imperfectly learned and potentially overfitted reward model leads to performance degeneration and a substantial decrease in the probability of choosing the preferred responses in the data [20, 33]. Given the importance of RLHF and the outlined challenge, a crucial research question is: *How to mitigate reward overoptimization in RLHF in a principled and efficient manner for better alignment?*

38th Workshop on Aligning Reinforcement Learning Experimentalists and Theorists (ARLET 2024).

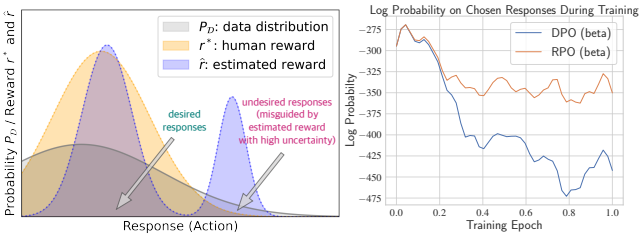

Figure 1: *Left*: Reward overoptimization due to the distributional shift and uncertainty in reward. *Right:* Overoptimization causes the probability of outputting preferred responses in the preference data to decrease substantially using original DPO proposed by [34]. Our algorithm (RPO) significantly alleviates this decrease. See more discussions in Section 6.

To answer the question, we model RLHF as an offline contextual bandit [30] and ascribe overoptimzation to distributional shifts and reward uncertainty. Intuitively, when fine-tuning an LLM, the response (action) distribution of the tuned LLM could deviate from that of the training data. For the out-of-distribution responses, which are dissimilar with (or not well covered by) the responses in the data, the high inherent uncertainty of underlying human preferences could make the learned reward model misleading for out-of-distribution responses. In this situation, reward overoptimization can occur because the LLM is fine-tuned towards maximizing a reward model with defective out-of-distribution prediction, giving a potential consequence that the LLM responses are favored by the learned reward but less preferred by a human [57]. We illustrate these types of distributional shift and reward uncertainty issues inherent to overoptimization in Figure 1.

In this paper, we propose a new RLHF algorithm to mitigate reward overoptimization. From a high level, our theoretical algorithm seeks the best LLM for an *adversarially* chosen reward model that minimizes the sum of its maximum likelihood estimation loss and its own expected reward value. Intuitively, since the reward value is also minimized when minimizing the sum, it can automatically prevent the misleadingly high reward caused by the uncertainty inherent in having access to finite data. Furthermore, we show that the theoretical algorithm enjoys an easy implementation: it simply adopts a supervised fine-tuning (SFT) loss as a regularizer during training. By explicitly regularizing the LLM to imitate high-quality responses (e.g., preferred responses in dataset), the algorithm can effectively mitigate the issue of overoptimization. We establish theoretical guarantees and conduct experiments to demonstrate our findings, which we summarize next.

1.1 Our Contributions and Related Works

We summarize our contributions in three areas as follows.

A theoretical algorithm under general function approximation. Our first contribution is a new theoretical algorithm (Algorithm 1). It features an *unconstrained maximin* problem, outputting the optimal policy (LLM) against an adversarially chosen reward model that minimizes the summation of: `(a)` the MLE loss for estimating the underlying reward; and `(b)` a reward expected value term as a penalty that aims to prevent spuriously high reward estimation caused by data uncertainty and insufficient coverage. Algorithm 1 is compatible with general function approximations of the reward model, meaning that we do not impose any specific structural form to the hypothesis class of reward, demonstrating its generality, especially in language modeling.

In this regime of reward class, we establish the finite-sample suboptimality gap of Algorithm 1 as $\widetilde{\mathcal{O}}(C_{\text{coverage}}^2\sqrt{\mathcal{N}_{\mathcal{R}}/N})$ when competing with any LLM in terms of the underlying true human reward (Theorem 5.3). Here N is the number of human comparison data, $\mathcal{N}_{\mathcal{R}}$ is the complexity of the reward model class $\mathcal{R}$, and C_{coverage} characterizes the coverage of the preference dataset with respect to the response distribution of the LLM to compete (please see Assumption 5.2 for details). This indicates that, as long as the training data well cover the LLM π to compete, the algorithm is guaranteed to align an LLM to output responses as good as π in terms of human reward, without suffering from overoptimization caused by distributional shifts and inherent uncertainty in human preference.

An easy-to-implement practical objective. Moving towards practice, we show that the objective of Algorithm 1 adopts a surprisingly simple and equivalent form for its use in practice. Specifically, with mild regularity conditions, we prove that the *maximin* objective (Algorithm1) is equivalent to the corresponding *minimax* objective, which is further reduced to a single minimization problem for the reward model since its inner problem adopts a closed form solution. Inspired from recent progress in RLHF that explores reward-model-free methods to align LLMs [34], we further re-parameterize the reward model via its corresponding KL-regularized optimal policy. Then the minimization objective of the reward modeling naturally translates to a target for directly aligning the LLM, which we call

Regularized Preference Optimization (RPO; Algorithm 2). The objective of RPO features a simple weighted combination of two losses:

$$\text{RPO objective} = \text{Preference optimization loss} + \text{Imitation (SFT) loss}.$$

Here the Preference optimization loss coincides with the DPO [34] objective, tending to optimize the LLM towards maximizing the underlying true reward. The Imitation (SFT) loss explicitly supervises the LLM to mimic the responses from a proper distribution well covered by the dataset. The choice of the distribution is guided and justified by our theory of Algorithm 1, but can also be flexibly adapted in practice, e.g., the preferred response in the dataset, or the responses of the initial model.

We highlight that the Imitation (SFT) loss serves as an important term to mitigate overoptimization. Even though the original DPO objective has already involved a KL regularization between the tuned LLM and the initial LLM, is not enough to prevent overoptimization. As we elaborate in Section 4, the KL-regularization weight of the DPO objective could only control the scale of the gradient per training example, while the RPO objective can further modify the gradient direction. Calling back to the theoretical Algorithm 1, such a modification of gradient direction originates from the reward penalty in the adversarial objective for the reward model. This modification, as we expose in our theoretical analysis, helps to mitigate overoptimization. *Thus, incoporating SFT loss in RLHF gives you a regularizer that provably mitigates overoptimization.*

Empirical evaluations. Following the training setup of two series of released chat models `Zephyr-7b-beta` (trained on the Ultrafeedback dataset [12] by DPO) and `Zephyr-7b-gemma` (trained on the Argilla-DPO-Mix-7K dataset [3] by DPO) [43], we implement RPO for the beta series and gemma series respectively to show that: (i) RPO is a flexible plug-in module and can be applied to different reference models. (ii) RPO can alleviate the overoptimization issue. (iii) RPO consistently achieves better alignment performance than DPO in in-data distribution. (iv) RPO can also achieve consistently better performance in standard LLM benchmarks like MT-bench and AlpacaEval 2.0, which shows its potential of mitigating overoptimization for better alignment performance, justifying our theory.

Related works. Due to space limitation, we refer the readers to Appendix A for a detailed discussion.

2 Preliminaries of RLHF

In this section, we introduce the mathematical framework of studying RLHF for aligning LLMs. We adopt the framework of offline contextual bandits [30], where we identify the context space $\mathcal{X}$ as the space of prompts and the action space $\mathcal{A}$ as the space of responses. An LLM, defined as a policy $\pi(\cdot|\cdot) : \mathcal{X} \mapsto \Delta(\mathcal{A})$, takes a prompt $x \in \mathcal{X}$ as input and output a response $a \in \mathcal{A}$ from $a \sim \pi(\cdot|x)$.

Preference model. Given any reward function $r : \mathcal{X} \times \mathcal{A} \mapsto \mathbb{R}$ belonging to certain reward class $\mathcal{R}$ that represents the "human's ratin" of LLM responses given prompts, we consider the Bradley-Terry model [7] of human preference. That is, given a prompt $x \in \mathcal{X}$ and two responses $a^1, a^0 \in \mathcal{A}$, the probability of a^1 being preferred to a^0 (denoted by $y = 1$, otherwise $y = 0$) is given by

$$\mathbb{P}_r(y = 1|x, a^1, a^0) = \frac{\exp(r(x, a^1))}{\exp(r(x, a^1)) + \exp(r(x, a^0))} = \sigma\big(r(x, a^1) - r(x, a^0)\big), \tag{2.1}$$

where $\sigma(z) = 1/(1 + \exp(-z))$ is the sigmoid function. For simplicity of future discussion, we explicitly write out the dependence of the preference probability $\mathbb{P}_r(\cdot)$ on the reward model $r \in \mathcal{R}$. In the section of theory, i.e., Section 5, we specify the assumptions on the reward model class $\mathcal{R}$.

Learning protocol. Typically, the RLHF pipeline starts from certain reference policy π^{ref} obtained from pretraining. Then RLHF aligns the LLM based on certain human preference data. In this work, we consider offline RLHF setup, where the LLM is aligned using a fixed offline preference dataset $\mathcal{D}$. It consists of N i.i.d. tuples in the form of $\mathcal{D} = \{(x_i, a_i^1, a_i^0, y_i)\}_{i=1}^N$. Here the prompt x_i and the responses a_i^1, a_i^0 are distributed according to: $(x, a^1, a^0) \sim \mu_{\mathcal{D}}(\cdot)$, and conditioning on (x_i, a_i^1, a_i^0), y_i is distributed according to (2.1) for an underlying true (but unknown) reward model $r^\star \in \mathcal{R}$.

Performance metric. The target of RLHF is to align an LLM, or equivalently, to learn a policy π, so as to maximize the expected true reward $r^\star$. Thus, we define the value function of any policy π as

$$J(\pi) = \mathbb{E}_{x\sim d_0, a\sim\pi(\cdot|x)}\big[r^\star(x, a)\big]. \tag{2.2}$$

Here we allow the prompt distribution $d_0(\cdot)$ to be different from that of the offline dataset distribution $\mu_{\mathcal{D}}(\cdot)$, but is assumed to be known. In the meanwhile, we consider the policies that share the same

Algorithm 1 Theoretical Algorithm: Maximin Objective

1: **Input**: Preference dataset $\mathcal{D}$, parameters $\beta, \eta > 0$, reference policy π^{ref}, baseline policy π^{base}.
2: **Output**: Policy $\widehat{\pi}$ given by (3.2) with the cross-entropy loss function $\mathcal{L}_{\mathcal{D}}$ defined in (3.1)..

support as the reference policy π^{ref} [47], that is, we take a policy class Π as

$$\Pi = \left\{ \pi : \mathcal{X} \mapsto \Delta(\mathcal{A}) \,\middle|\, \text{Supp}(\pi(\cdot|x)) \subseteq \text{Supp}(\pi^{\text{ref}}(\cdot|x)),\ \forall x \in \mathcal{X} \right\}. \tag{2.3}$$

The performance gap of a learned policy $\widehat{\pi} \in \Pi$ w.r.t. any other policy $\pi \in \Pi$ is measured as

$$\text{Gap}^{\pi}(\widehat{\pi}) = J(\pi) - J(\widehat{\pi}),\ \textit{given policy } \pi. \tag{2.4}$$

The goal is to propose a sample-efficient and also implementation-friendly algorithm to learn a policy $\widehat{\pi} \in \Pi$ able to compete with any given policy $\pi \in \Pi$ in terms of $\text{Gap}^{\pi}(\widehat{\pi}) \le \varepsilon$, with sample complexity polynomial in $1/\varepsilon$ and logarithmic in the complexity of $\mathcal{R}$.

3 A Theory-motivated Objective

Our method seeks to find the best policy $\widehat{\pi}$ against an *adversarially* chosen reward model $\widehat{r}_{\text{adv}}$ that minimizes a weighted sum of its expected value and the maximum likelihood estimation (MLE) loss. Intuitively, such a reward model can prevent the overoptimization issue by taking its own value into account when minimizing the MLE loss. Since the reward value is also minimized when minimizing the sum, this method prevents the misleadingly high reward caused by the uncertainty due to finite data. Formally, given two hyperparameters $\beta, \eta > 0$ and a "baseline policy" π^{base}, we define

$$T^{\text{adv}}_{\beta,\eta}(\pi) = \min_{r \in \mathcal{R}} \left\{ \eta \mathbb{E}_{\substack{x \sim d_0, a^1 \sim \pi(\cdot|x), \\ a^0 \sim \pi^{\text{base}}(\cdot|x)}} \Big[r(x, a^1) - r(x, a^0) - \beta \cdot \text{KL}\big(\pi(\cdot|x) \| \pi^{\text{ref}}(\cdot|x)\big) \Big] + \mathcal{L}_{\mathcal{D}}(r) \right\},$$

where the loss function $\mathcal{L}_{\mathcal{D}}(\cdot)$ is the average negative log-likelihood function of the BT model (2.1) (and here it becomes the cross-entropy loss) over the preference dataset $\mathcal{D}$, defined as

$$\mathcal{L}_{\mathcal{D}}(r) = -\widehat{\mathbb{E}}_{\mathcal{D}} \Big[y_i \log\big(\sigma\big(r(x_i, a_i^1) - r(x_i, a_i^0)\big)\big) + (1 - y_i) \log\big(\sigma\big(r(x_i, a_i^0) - r(x_i, a_i^1)\big)\big) \Big]. \tag{3.1}$$

As we can see, $T^{\text{adv}}_{\beta,\eta}(\pi)$ is the minimum value of a weighted sum of the MLE loss and the expected reward value of π, but with two important modifications that we explain in the following.

Firstly, we subtract another expected reward of certain policy π^{base}. This is because the BT model (2.1) essentially only uses the reward differences to define the preference probabilities. As a result, the data can only reveal information of the differences between the true reward $r^\star$ of different responses [50]. Accordingly, we subtract such a baseline expected reward value to match this observation. The choice of the baseline policy is discussed in the theory part (Section 5) and experiments (Section 6).

Secondly, we subtract a KL divergence between π and π^{ref} from the expected reward, weighted by the coefficient $\beta > 0$. Such a term is for practical considerations that would be explained in Sections 4 and 5.2. We note that the KL regularized reward is commonly adopted in RLHF practice to ensure the learned policy is not far away from the reference policy [30, 47].

Finally, the overall algorithm design (Algorithm 1) is to output the policy that maximizes $T^{\text{adv}}_{\beta,\eta}(\pi)$, i.e., $\widehat{\pi} \in \text{argmax}_{\pi \in \Pi} T^{\text{adv}}_{\beta,\eta}(\pi)$, which gives the following theoretical target:

$$\widehat{\pi} \in \operatorname*{argmax}_{\pi \in \Pi} \min_{r \in \mathcal{R}} \left\{ \eta \mathbb{E}_{\substack{x \sim d_0, a^1 \sim \pi(\cdot|x), \\ a^0 \sim \pi^{\text{base}}(\cdot|x)}} \Big[r(x, a^1) - r(x, a^0) - \beta \cdot \text{KL}\big(\pi(\cdot|x) \| \pi^{\text{ref}}(\cdot|x)\big) \Big] + \mathcal{L}_{\mathcal{D}}(r) \right\}. \tag{3.2}$$

Given the form of (3.2), we name it the *maximin* objective in the sequel. Upon seeing (3.2), one might be arguing that such a theory-motivated objective seems hard to implement in practice. Nevertheless, in the coming Section 4, we demonstrate that the maximin objective (3.2) adopts an easy-to-implement equivalent form, allowing us to design a practical algorithm for aligning LLMs.

4 An Equivalent and Implementation-friendly Objective

In this section, we propose another *minimax*-style objective that is equivalent to the maximin objective (3.2). Based on the minimax objective, we propose a new LLM aligning algorithm called Regularized Preference Optimization (RPO). It draws inspirations from the reparametrization technique originated in Direct Preference Optimization (DPO) [34] and goes beyond to further address the overoptimization issue in offline RLHF by incorprating an SFT loss as an explicit adversarial regularizer.

An equivalent minimax objective. If the reward model class $\mathcal{R}$ satisfies certain regularity conditions, which we discuss in detail in Section 5.2, the minimax theorem holds: solving the *maximin* objective (3.2) is *equivalent* to solving a *minimax* target, given by

$$\min_{r\in\mathcal{R}}\max_{\pi\in\Pi}\left\{\eta\mathbb{E}_{\substack{x\sim d_0,a^1\sim\pi(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1)-r(x,a^0)-\beta\cdot\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]+\mathcal{L}_{\mathcal{D}}(r)\right\}. \tag{4.1}$$

Such a minimax formulation (4.1) is the starting point of our practical algorithm. The magic of (4.1) is that the inner maximization problem adopts a closed form solution, which further simplifies such an objective. To see this, note that given any reward model $r\in\mathcal{R}$, the inner problem is equivalent to

$$\max_{\pi\in\Pi}\left\{\mathbb{E}_{x\sim d_0,a\sim\pi(\cdot|x)}\Big[r(x,a)-\beta\cdot\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]\right\}. \tag{4.2}$$

It has been well established that the policy that maximizes the KL-regularized expected reward (4.2) has a closed form solution. Due to its importance, we present it as the following lemma.

Lemma 4.1 (Oracle optimal KL-regularized policy)**.** *Given any reward model $r\in\mathcal{R}$, the optimal policy π_r to the maximization problem (4.2) is given by*

$$\pi_r(\cdot|x)=\frac{1}{Z_r(x)}\cdot\pi^{\mathrm{ref}}(\cdot|x)\cdot\exp\left(\beta^{-1}r(x,\cdot)\right),\ Z_r(x)=\int_{a\in\mathcal{A}}\exp\left(\beta^{-1}r(x,a)\right)\mathrm{d}\pi^{\mathrm{ref}}(a|x),$$

and correspondingly the optimal value of (4.2) is given by (4.2) $=\mathbb{E}_{x\sim d_0}[\beta\cdot\log(Z_r(x))]$.

Specifically, by Lemma 4.1, we can solve the inner maximization problem in (4.1) and obtain that

$$(4.1)=\min_{r\in\mathcal{R}}\left\{\eta\mathbb{E}_{x\sim d_0,a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[-r(x,a^0)+\beta\cdot\log\left(Z_r(x)\right)\Big]+\mathcal{L}_{\mathcal{D}}(r)\right\}.$$

Furthermore, from Lemma 4.1, one immediately see that given any reward model $r\in\mathcal{R}$, we can reparameterize it via its corresponding optimal KL-regularized policy π_r [34], that is,

$$r(x,\cdot)=\beta\cdot\log\left(\frac{\pi_r(\cdot|x)}{\pi^{\mathrm{ref}}(\cdot|x)}\right)+\beta\cdot\log(Z_r(x)). \tag{4.3}$$

Taking (4.3) back into (4.1), we are able to further simplify it as

$$(4.1)=\min_{r\in\mathcal{R}}\left\{\eta\mathbb{E}_{x\sim d_0,a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[-\beta\cdot\log(\pi_r(a^0|x))\Big]+\mathcal{L}_{\mathcal{D}}\left(\beta\cdot\log\left(\frac{\pi_r(\cdot|\cdot)}{\pi^{\mathrm{ref}}(\cdot|\cdot)}\right)\right)\right\}. \tag{4.4}$$

Thanks to the KL-regularization term in the original minimax objective (4.1) (or equivalently, the maximin objective (3.2)), we have the following theorem. It theoretically shows that the policy $\pi_{\widehat{r}}$ associated with the reward model $\widehat{r}$ solving (4.4) also solves the maximin target (3.2) of the theoretical algorithm (Algorithm 1) that enjoys finite-sample convergence guarantees. (Please see Section 5.2 for a formal statement and proof of Theorem 4.2).

Theorem 4.2 (Equivalence between *maximin* and *minimax* algorithm (informal))**.** *Under certain regularity assumptions on $\mathcal{R}$ and given $\eta,\beta>0$, solving the minimax objective (4.1) via (4.4), i.e.,*

$$\widehat{r}=\operatorname*{argmin}_{r\in\mathcal{R}}\left\{\eta\mathbb{E}_{x\sim d_0,a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[-\beta\cdot\log(\pi_r(a^0|x))\Big]+\mathcal{L}_{\mathcal{D}}\left(\beta\cdot\log\left(\frac{\pi_r(\cdot|\cdot)}{\pi^{\mathrm{ref}}(\cdot|\cdot)}\right)\right)\right\},$$

then the corresponding optimal KL-regularized policy $\pi_{\widehat{r}}$ also solves the maximin objective (3.2).

Regularized Preference Optimization. Target (4.4) gives a quite simple objective to use in practice! Since (4.4) depends on $r\in\mathcal{R}$ only through its corresponding optimal policy π_r, one can formulate

Algorithm 2 Practical Algorithm: Regularized Preference Optimization (RPO)

1: **Input**: Preference dataset $\mathcal{D}$, parameters $\beta, \eta > 0$, reference policy π^{ref}, baseline policy π^{base}.
2: **Output**: Policy $\pi_{\widehat{\theta}}$ obtained by optimizing objective (4.5).

a minimization objective over a parameterized policy π_θ, i.e., the LLM to be aligned, and directly optimize the parameters $\theta \in \Theta$. More formally, the new RLHF objective becomes

$$\min_{\theta\in\Theta}\left\{\mathcal{L}_{\text{RPO}}(\theta) := \eta\beta \cdot \underbrace{\mathbb{E}_{x\sim d_0, a^0\sim\pi^{\text{base}}(\cdot|x)}\Big[-\log(\pi_\theta(a^0|x))\Big]}_{\text{Imitation (SFT) loss}} + \underbrace{\mathcal{L}_{\mathcal{D}}\left(\beta\cdot\log\left(\frac{\pi_\theta(\cdot|\cdot)}{\pi^{\text{ref}}(\cdot|\cdot)}\right)\right)}_{\text{Preference opt. loss}}\right\}. \tag{4.5}$$

In (4.5), the second term coincides with the objective of DPO algorithm [34] which optimizes the policy towards maximizing the underlying true reward, and the first term stands for a regularization term weighted by $\eta \cdot \beta$ which *explicitly* regularizes the policy to imitate the baseline policy. Therefore, we name the resulting algorithm as Regularized Preference Optimization (RPO). We summarize it abstractly in Algorithm 2. As for DPO, implementing RPO does not require to maintain a reward model r. Thus it is computationally more friendly compared to reward-based algorithms.

How does RPO improve DPO? We illustrate the effect of the imitation loss by analyzing the gradient of the RPO target $\mathcal{L}_{\text{RPO}}(\theta)$ in (4.5). Notice that by (4.5) we have

$$\nabla_\theta\mathcal{L}_{\text{RPO}}(\theta) = \eta\beta\cdot\underbrace{\mathbb{E}_{x\sim d_0,a^0\sim\pi^{\text{base}}(\cdot|x)}\Big[-\nabla_\theta\log(\pi_\theta(a^0|x))\Big]}_{\text{increase the alignment with the baseline policy}} + \underbrace{\nabla_\theta\mathcal{L}_{\text{DPO}}(\theta)}_{\text{decrease the DPO Loss}},$$

where the derivative of the DPO loss $\nabla_\theta\mathcal{L}_{\text{DPO}}(\theta)$ is given by the following,

$$\nabla_\theta\mathcal{L}_{\text{DPO}}(\theta) = -\widehat{\mathbb{E}}_{\mathcal{D}}\Big[\underbrace{\beta\cdot\sigma\big(\widehat{r}_\theta(x,a_{\text{rej}}) - \widehat{r}_\theta(x,a_{\text{cho}})\big)}_{\text{gradient weight}}\cdot\Big(\nabla_\theta\log\pi_\theta(a_{\text{cho}}|x) - \nabla_\theta\log\pi_\theta(a_{\text{rej}}|x)\Big)\Big].$$

For simplicity we denote $\widehat{r}_\theta(x,a) = \beta\cdot\log(\pi_\theta(x,a))/\log(\pi^{\text{ref}}(x,a))$, a_{cho} for the chosen response and a_{rej} for the rejected response. Intuitively, RPO (4.5) modifies the **gradient direction** of DPO to ensure the alignment with the baseline policy π^{base}, and the hyper-parameter η controls the power of alignment. In comparison, the hyper-parameter β in DPO only controls the **gradient weight** when increasing the likelihood of a_{cho} and decreasing the likelihood a_{rej}. In this perspective, the hyper-parameter β only changes the scale of the gradient instead of the direction. By introducing η, we stabilize the training and reduce the side-effect of uncertain labels in data to prevent overoptimization.

5 Theoretical Analysis

In this section, we establish theoretical analysis for Algorithms 1 and 2. We take the space of prompts and responses as compact subsets $\mathcal{X} \subseteq \mathbb{R}^{d_{\mathcal{X}}}$ and $\mathcal{A} \subseteq \mathbb{R}^{d_{\mathcal{A}}}$. We take the policy class Π as (2.3).

5.1 Establishing the Sample Complexity of Maximin Objective (Algorithm 1)

Assumption 5.1 (True reward model)**.** *We assume that the true reward model $r^\star \in \mathcal{R}$, and for any $r \in \mathcal{R}$ and $(x,a) \in \mathcal{X}\times\mathcal{A}$, it holds that $r(x,a) \in [0,R]$.*

Assumption 5.2 (Partial coverage coefficient [50])**.** *Given a policy $\pi \in \Pi$, the coverage coefficient of the offline dataset distribution $\mu_{\mathcal{D}}$ w.r.t. reward model class $\mathcal{R}$, policy π, and the baseline policy π^{base}, denoted by $C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\text{base}})$, is defined as*

$$\max\left\{0, \sup_{r\in\mathcal{R}}\frac{\mathbb{E}_{x\sim d_0,a^1\sim\pi(\cdot|x),a^0\sim\pi^{\text{base}}(\cdot|x)}\big[(r^\star(x,a^1)-r^\star(x,a^0))-(r(x,a^1)-r(x,a^0))\big]}{\sqrt{\mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}}\Big[\big|(r^\star(x,a^1)-r^\star(x,a^0))-(r(x,a^1)-r(x,a^0))\big|^2\Big]}}\right\}.$$

We assume that $C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\text{base}}) < +\infty$ for the policy π to compete. We remark that the quantity $C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\text{base}})$ is upper bounded by the density ratio $\|d_0\otimes\pi\otimes\pi^{\text{base}}/\mu_{\mathcal{D}}\|_\infty$.

Assumption 5.1 is standard in sample complexity analysis [56, 50, 48]. Assumption 5.2 characterizes how well the dataset $\mathcal{D}$ covers the policy π to compete. To achieve provable sample efficiency, we only require that $\mathcal{D}$ covers the target policy π, a weak partial coverage style assumption for theoretical analysis. To illustrate it, when calling back to Figure 1, the data distribution therein well covers those nearly optimal responses under $r^\star$, but does not sufficiently cover the responses with low $r^\star$.

Under such a partial coverage data condition, however, human preference of responses $a \in \mathcal{A}$ that are not well covered by the dataset $\mathcal{D}$ can be poorly estimated, misguiding the policy $\widehat{\pi}$ to behave suboptimally if it is overoptimized (recall Figure 1). Fortunately, the following theorem shows that Algorithm 1 provably mitigates the overoptimization issue and achieves a finite-sample convergence of the suboptimality gap (2.4) competing with π. Proof is in Appendix C.

Theorem 5.3 (Suboptimality of Algorithm 1)**.** *Taking the policy class Π as (2.3), supposing that Assumptions 5.1 and 5.2 hold, and assuming that the reward model class $\mathcal{R}$ has a finite ε-epsilon covering number under $\|\cdot\|_\infty$-norm $\mathcal{N}_\varepsilon(\mathcal{R}, \|\cdot\|_\infty) < +\infty$ with $\varepsilon = (6 \cdot (1 + e^R) \cdot N)^{-1}$. Setting*

$$\eta = (1 + \exp(R))^{-2} \cdot \sqrt{24 \log\left(\mathcal{N}_\varepsilon(\mathcal{R}, \|\cdot\|_\infty)/\delta\right)/N}, \quad \beta = 1/\sqrt{N}$$

in Algorithm 1. Then the output policy $\widehat{\pi}$ of Algorithm 1 satisfies that with probability at least $1 - \delta$,

$$\mathrm{Gap}^{\pi}(\widehat{\pi}) \leq \frac{\sqrt{6}\big(1 + \exp(R)\big)^2 \Big(\big(C_{\mu_\mathcal{D}}(\mathcal{R}; \pi, \pi^{\mathrm{base}})\big)^2 + 1\Big)\iota + 4\mathbb{E}_{x \sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x) \| \pi^{\mathrm{ref}}(\cdot|x)\big)\Big]}{4\sqrt{N}},$$

where $\iota = \sqrt{\log\left(\mathcal{N}_\varepsilon(\mathcal{R}, \|\cdot\|_\infty)/\delta\right)}$ with $\varepsilon = (6 \cdot (1 + e^R) \cdot N)^{-1}$. Here, N denotes the number of preference pairs in $\mathcal{D}$, R denotes the upper bound of the reward models, and the partial coverage coefficient $C_{\mu_\mathcal{D}}(\mathcal{R}; \pi, \pi^{\mathrm{base}})$ is defined in Assumption 5.2.

Remark 5.4 (Choice of the baseline policy)**.** *As is indicated by Assumption 5.2, the least requirement is that π^{base} can be covered by the offline data distribution. E.g., we can take π^{base} as the distribution of the preferred responses in the data. In this case, the SFT loss in RPO explicitly regularizes the LLM to imitate the preferred responses. We choose this type of baseline policy in our experiments.*

5.2 Equivalence between Maximin and Minimax Objectives

Now we formally show that the theoretical target (maximin objective (3.2)) and the target for practical algorithm design (minimax objective (4.1)) are equivalent under certain regularity conditions. This can naturally extend the sample complexity of Algorithm 1 (Section 5.1) to that of minimax-based algorithms in Section 4, providing the theoretical guarantee for our practical algorithm design (RPO).

First, for notational simplicity, we denote the optimization target we investigate in Sections 3 and 4 as

$$\phi(\pi, r) := \eta \cdot \mathbb{E}_{\substack{x \sim d_0, a^1 \sim \pi(\cdot|x) \\ a^0 \sim \pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x, a^1) - r(x, a^0) - \beta \cdot D_{\mathrm{KL}}\big(\pi(\cdot|x) \| \pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_\mathcal{D}(r), \quad (5.1)$$

for any $(\pi, r) \in \Pi \times \mathcal{R}$. Our result relies on the following assumptions on the reward model class $\mathcal{R}$.

Assumption 5.5 (Regularity of reward model class)**.** *We assume the following things on the reward model class $\mathcal{R}$: (i) the space $\mathcal{R}$ is a compact topological space; (ii) the function ϕ in (5.1) is convex-like on $\mathcal{R}$, that is, for any $r_1, r_2 \in \mathcal{R}$ and $\alpha \in [0, 1]$, there exists $r_3 \in \mathcal{R}$ such that*

$$\phi(\pi, r_3) \leq \alpha \cdot \phi(\pi, r_1) + (1 - \alpha) \cdot \phi(\pi, r_2), \quad \forall \pi \in \Pi, \quad (5.2)$$

We note if $\mathcal{R}$ is convex, e.g., a linear model class [56, 47, 57] or more general the Lipschitz continuous model class $\mathcal{R}$, we can directly obtain that the function $\phi(\pi, \cdot)$ is *convex* over $\mathcal{R}$ (since the dependence on $r \in \mathcal{R}$ is linear terms plus a convex loss $\mathcal{L}_\mathcal{D}$ of $r \in \mathcal{R}$), which implies the convex-like property (5.2). Under Assumption 5.5, it holds that (Lemma D.1)

$$\max_{\pi \in \Pi} \min_{r \in \mathcal{R}} \phi(\pi, r) = \min_{r \in \mathcal{R}} \max_{\pi \in \Pi} \phi(\pi, r). \quad (5.3)$$

Furthermore, thanks to the KL-divergence regularization in ϕ which intuitively makes ϕ "strongly concave" over the policy π, (5.3) can gives us the following stronger result, proved in Appendix D.1.

Theorem 5.6 (Formal statement of Theorem 4.2)**.** *For the policy class Π defined in (2.3) and the reward model class $\mathcal{R}$ satisfying Assumption 5.5, consider the following policy defined as*

$$\pi_{\widehat{r}} \in \operatorname*{argmax}_{\pi \in \Pi} \phi(\widehat{r}, \pi), \quad \textit{where} \quad \widehat{r} \in \operatorname*{argmin}_{r \in \mathcal{R}} \max_{\pi \in \Pi} \phi(\pi, r). \quad (5.4)$$

Then the policy $\pi_{\widehat{r}}$ also satisfies the maximin objective (3.2) of Algorithm 1, that is,

$$\pi_{\widehat{r}} \in \operatorname*{argmax}_{\pi \in \Pi} \min_{r \in \mathcal{R}} \phi(\pi, r).$$

Theorem 5.6 shows that the optimal KL-regularized policy associated with the reward model solving the minimax objective (3.2) also solves the maximin objective (i.e., objective (4.1) of Algorithm 1). This further allows us to extend our theoretical guarantee of Algorithm 1 (Section 5.1) to that of minimax-based algorithms, justifying our practical algorithm design in Section 4.

Corollary 5.7 (Suboptimality of minimax-based algorithm)**.** *Take the policy class Π in (2.3) and the reward model class satisfying Assumption 5.5. Given any given policy π to compete, if Assumption 5.2 holds for π, then under the same choice of η and β as in Theorem 5.3, the policy $\pi_{\widehat{r}}$ defined in (5.4) satisfies that $\mathrm{Gap}^{\pi}(\pi_{\widehat{r}}) \leq \widetilde{\mathcal{O}}(1/\sqrt{N})$ with probability at least $1-\delta$.*

6 Experiments

Experiment setup. To show that RPO is a flexible plug-in module regardless of the reference model, we follow the training setup for two well-studied series of released chat models with around 7 billion parameters trained by DPO: `Zephyr-7b-beta` and `Zephyr-7b-gemma` [43] to implement RPO in beta and gemma series. Mirrored by their training configurations, we introduce how we select the reference model and the preference dataset for our training on these two series as follows. For the beta series, we use `mistral-7b-sft-beta` as the reference model π^{ref}. `mistral-7b-sft-beta` is a fine-tuned version of `Mistral-7b-v0.1` on the distilled version of the UltraChat dataset [13], which contains approximately 200k examples of multi-turn dialogues generated by GPT-3.5-TURBO. For the training preference dataset, we use Ultrafeedback Dataset [12], which consists of approximately 60k prompts. For the gemma series, we use `zephyr-7b-gemma-sft-v0.1` as our reference model π^{ref}. `zephyr-7b-gemma-sft-v0.1` is a fine-tuned version of `gemma-7b` on the Deita dataset [26], which involves around 10k distilled SFT data. For the training preference dataset, we use Argilla-DPO-Mix-7K Dataset [3], which is a mixture of multiple distilled public preference datasets. For simplicity, we denote Ref. (beta) as the reference model, DPO (beta) as the model trained by DPO, RPO (beta) as the model trained by RPO, all for the beta series. We use the same notations for the gemma series.

Practical implementation. According to Algorithm 2 and as discussed in Remark 5.4, we implement RPO by adding an SFT loss (log probability of chosen responses in the preference dataset) to the original DPO loss. By comparing the evaluation performance on the test split of the training dataset, we select the hyperparameter η as 0.005 for both RPO (beta) and RPO (gemma). During the training of DPO and RPO, We keep the remaining hyperparameters including β, batch size, and learning rate to be the same for a fair comparison. Please see Appendix E.1 for a detailed training configuration.

RPO alleviates overoptimization. As mentioned in the introduction part, DPO is observed to have a significant and continuous decrease in log probability on chosen responses [20, 33] during training and we regard it as the consequence of overoptimization. Implied by our theory, overoptimization could arise when the model maximizes its own proxy reward formed on the responses less covered by the data. Due to the overoptimization, the model tends to disprefer the chosen responses as they are away from the maximizers of the proxy reward despite that some chosen responses are highly preferred by humans. Consistent with our theoretical conclusion, we empirically find that RPO can indeed alleviate overoptimization in DPO. During the training phase of both beta and gemma series, we observe that the log probability given by the RPO-trained model is notably higher than that given by the DPO-trained model for the chosen responses, which are shown in Fig. 1 and 2.

RPO improves the alignment ability in in-data distribution. For the in-data distribution evaluation, we select the 200 prompts (which are not used in the selection of η) in the test split of the training dataset to let the reference model, DPO, and RPO generate the response respectively. We choose GPT-4 to annotate the preference in the response pairs. Though we instruct GPT-4 to give an annotation among win, lose, and, tie (please see the full prompt in Appendix E.2), GPT-4 may still give undesired annotations. Hence, we filter all the undesired annotations and collect 150 examples for evaluation. We report the pairwise win rate among Ref., RPO, and DPO in Table 1 for both the beta and gemma series. To show a more illustrative comparison between DPO and RPO, we provide the barplot to report the number of pairwise examples annotated by GPT-4 in Fig. 3 and Fig. 4. We observe that for both beta and gemma series, RPO has a better performance than DPO in terms of both RPO/DPO-SFT

and RPO-DPO win rates. The performance improvement matches our theoretical results in Corollary 5.7, which shows the credit of the alleviation of overoptimization.

Table 1: Pairwise win rate (left vs. right) among RPO-trained model, DPO-trained model, and the reference model. Annotated by GPT-4, evaluations of beta and gemma series are made on the 150 examples of the test split of the Ultrafeedback and the Argilla-DPO-Mix-7K dataset, respectively.

Win rate (%)	RPO (beta)	Ref. (beta)	DPO (beta)	Win rate (%)	RPO (gemma)	Ref. (gemma)	DPO (gemma)
RPO (beta)	50.0	**79.0**	**56.0**	RPO (gemma)	50.0	**71.7**	**54.0**
Ref. (beta)	21.0	50.0	22.7	Ref. (gemma)	28.3	50.0	32.7
DPO (beta)	44.0	77.3	50.0	DPO (gemma)	46.0	67.3	50.0

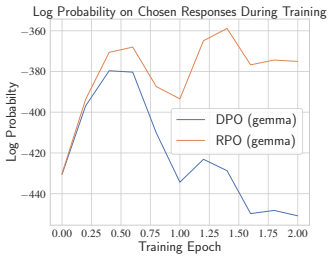

Figure 2: Log probability of the model for chosen responses during the training of RPO (gemma) and DPO (gemma).

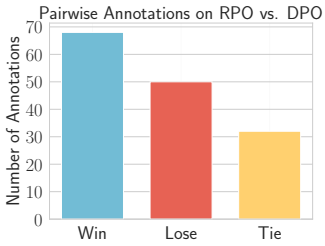

Figure 3: Pairwise annotations (by GPT-4) on RPO (beta) vs. DPO (beta) on the test split of the Ultrafeedback dataset.

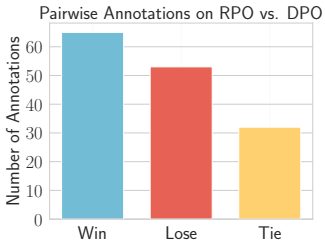

Figure 4: Pairwise annotations (by GPT-4) on RPO (gemma) vs. DPO (gemma) on the test split of the Argilla-DPO-Mix-7K dataset.

RPO consistently improves the benchmark performance. We further evaluate the reference model, RPO-trained model, DPO-trained model, and the officially released DPO-trained model for both beta and gemma series in two standard LLM chat benchmarks: MT-Bench and AlpacaEval 2.0. MT-Bench is a multi-turn benchmark that contains 160 questions across eight different domains of knowledge. The score for MT-Bench is evaluated by GPT-4 on a scale from 1 to 10. AlpacaEval 2.0 is a single-turn benchmark including 805 questions on different topics, mostly focused on helpfulness. The metrics of AlpacaEval 2.0 are the win rate and Length-Control (LC) win rate compared with GPT-4 Preview (11/06), where the annotator is also GPT-4 Preview (11/06) and LC win rate is proposed to mitigate the length bias of GPT-4. The results are summarized in Table 2, which shows that RPO consistently exceeds the performance of all the competitors (DPO, Reference model, and the officially released model trained by DPO) on MT-Bench and AlpacaEval 2.0. We also provide additional results on the pairwise win rate for these two benchmarks in Appendix E.3 to illustrate the performance improvement. Finally, we remark that RPO is a flexible plug-in module and can steadily improve the benchmark performance without changing the original training configuration or accessing extra preference data. This also sheds light on the potential of mitigating overoptimization for better alignment and generalization performance.

Table 2: Results on MT-Bench scores and AlpacaEval 2.0. `zephyr-beta-7b` and `zephyr-gemma-7b` are the officially released models. win rates and Length-Control (LC) win rates in AlpacaEval 2.0 are evaluated by GPT-4 compared with GPT-4.

Model Name	MT-Bench Score	AlpacaEval 2.0 LC win rate (%)	AlpacaEval 2.0 win rate (%)	Model Name	MT-Bench Score	AlpacaEval 2.0 LC win rate (%)	AlpacaEval 2.0 win rate (%)
RPO (beta)	**7.381**	**23.28**	**21.01**	RPO (gemma)	**7.916**	**15.51**	**13.85**
Ref. (beta)	5.088	7.19	4.69	Ref. (gemma)	7.266	8.35	4.61
DPO (beta)	7.278	21.15	17.27	DPO (gemma)	7.688	15.36	13.69
`zephyr-beta-7b`	7.200	13.20	10.99	`zephyr-gemma-7b`	7.719	14.78	12.14

7 Conclusions

This work proposes a new algorithm that provably mitigates reward overoptimization in RLHF. We establish its finite-sample convergence under a partial coverage style data condition, and provide an equivalent practical implementation, RPO. As a flexible plug-in module, RPO exhibits consistent improvement over the DPO baseline and effectively mitigates overoptimization. Future work includes extending our idea of algorithm design to online (iterative) RLHF where preference data are collected and updated iteratively during LLM fine-tuning. We give more detailed discussions in Appendix B.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 14

[2] Anthropic. Introducing claude. *https://www.anthropic.com/news/introducing-claude*, 2023. 14

[3] argill. argilla-dpo-mix-7k. https://huggingface.co/datasets/argilla/dpo-mix-7k. 3, 8

[4] Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*, 2023. 14

[5] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022. 14

[6] Viktor Bengs, Róbert Busa-Fekete, Adil El Mesaoudi-Paul, and Eyke Hüllermeier. Preference-based online learning with dueling bandits: A survey. *Journal of Machine Learning Research*, 22(7):1–108, 2021. 14

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

[8] Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023. 1, 14

[9] Xiaoyu Chen, Han Zhong, Zhuoran Yang, Zhaoran Wang, and Liwei Wang. Human-in-the-loop: Provably efficient preference-based reinforcement learning with general function approximation. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 3773–3793. PMLR, 17–23 Jul 2022. 14

[10] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017. 1, 14

[11] Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. *arXiv preprint arXiv:2310.02743*, 2023. 15

[12] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023. 3, 8

[13] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023. 8

[14] Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023. 14

[15] Yihan Du, Anna Winnicki, Gal Dalal, Shie Mannor, and R Srikant. Exploration-driven policy optimization in rlhf: Theoretical insights on efficient data utilization. *arXiv preprint arXiv:2402.10342*, 2024. 14

[16] Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D'Amour, DJ Dvijotham, Adam Fisch, Katherine Heller, Stephen Pfohl, Deepak Ramachandran, et al. Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking. *arXiv preprint arXiv:2312.09244*, 2023. 15

[17] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*, 2020. 14

[18] Ky Fan. Minimax theorems. *Proceedings of the National Academy of Sciences*, 39(1):42–47, 1953. 21

[19] Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR, 2023. 1, 14

[20] Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*, 2024. 1, 8, 14

[21] Haozhe Ji, Cheng Lu, Yilin Niu, Pei Ke, Hongning Wang, Jun Zhu, Jie Tang, and Minlie Huang. Towards efficient and exact optimization of language model alignment. *arXiv preprint arXiv:2402.00856*, 2024. 14

[22] Chenliang Li, Siliang Zeng, Zeyi Liao, Jiaxiang Li, Dongyeop Kang, Alfredo Garcia, and Mingyi Hong. Joint demonstration and preference learning improves policy alignment with human feedback. *arXiv preprint arXiv:2406.06874*, 2024. 14

[23] Zihao Li, Zhuoran Yang, and Mengdi Wang. Reinforcement learning with human feedback: Learning dynamic choices via pessimism. *arXiv preprint arXiv:2305.18438*, 2023. 14

[24] Xize Liang, Chao Chen, Jie Wang, Yue Wu, Zhihang Fu, Zhihao Shi, Feng Wu, and Jieping Ye. Robust preference optimization with provable noise tolerance for llms. *arXiv preprint arXiv:2404.04102*, 2024. 14

[25] Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*, 2023. 14

[26] Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*, 2023. 8

[27] Zhihan Liu, Miao Lu, Wei Xiong, Han Zhong, Hao Hu, Shenao Zhang, Sirui Zheng, Zhuoran Yang, and Zhaoran Wang. Maximize to explore: One objective function fusing estimation, planning, and exploration. *Advances in Neural Information Processing Systems*, 36, 2024. 18

[28] Eric J Michaud, Adam Gleave, and Stuart Russell. Understanding learned reward functions. *arXiv preprint arXiv:2012.05862*, 2020. 1, 14

[29] Ted Moskovitz, Aaditya K Singh, DJ Strouse, Tuomas Sandholm, Ruslan Salakhutdinov, Anca D Dragan, and Stephen McAleer. Confronting reward model overoptimization with constrained rlhf. *arXiv preprint arXiv:2310.04373*, 2023. 15

[30] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022. 1, 2, 3, 4, 14

[31] Aldo Pacchiano, Aadirupa Saha, and Jonathan Lee. Dueling rl: reinforcement learning with trajectory preferences. *arXiv preprint arXiv:2111.04850*, 2021. 14

[32] Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024. 14

[33] Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to $q^\star$: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024. 1, 8

[34] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2023. 2, 3, 5, 6, 14

[35] Mathieu Rita, Florian Strub, Rahma Chaabouni, Paul Michel, Emmanuel Dupoux, and Olivier Pietquin. Countering reward over-optimization in llm with demonstration-guided reinforcement learning. *arXiv preprint arXiv:2404.19409*, 2024. 15

[36] Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*, 2024. 14

[37] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 14

[38] Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of llms should leverage suboptimal, on-policy data. *arXiv preprint arXiv:2404.14367*, 2024. 14, 15

[39] Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*, 2024. 14

[40] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 14

[41] Jeremy Tien, Jerry Zhi-Yang He, Zackory Erickson, Anca D Dragan, and Daniel S Brown. Causal confusion and reward misidentification in preference-based reward learning. *arXiv preprint arXiv:2204.06601*, 2022. 1, 14

[42] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alexander M. Rush, and Thomas Wolf. The alignment handbook. https://github.com/huggingface/alignment-handbook, 2023. 22

[43] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023. 3, 8

[44] Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*, 2023. 23

[45] Yuanhao Wang, Qinghua Liu, and Chi Jin. Is rlhf more difficult than standard rl? a theoretical perspective. *Advances in Neural Information Processing Systems*, 36, 2023. 14

[46] Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. *arXiv preprint arXiv:2405.00675*, 2024. 14

[47] Wei Xiong, Hanze Dong, Chenlu Ye, Han Zhong, Nan Jiang, and Tong Zhang. Gibbs sampling from human feedback: A provable kl-constrained framework for rlhf. *arXiv preprint arXiv:2312.11456*, 2023. 4, 7, 14, 15

[48] Chenlu Ye, Wei Xiong, Yuheng Zhang, Nan Jiang, and Tong Zhang. A theoretical analysis of nash learning from human feedback under general kl-regularized preference. *arXiv preprint arXiv:2402.07314*, 2024. 7, 14

[49] Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012. 14

[50] Wenhao Zhan, Masatoshi Uehara, Nathan Kallus, Jason D Lee, and Wen Sun. Provable offline preference-based reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2023. 4, 6, 7, 14, 15

[51] Wenhao Zhan, Masatoshi Uehara, Wen Sun, and Jason D Lee. How to query human feedback efficiently in rl? *arXiv preprint arXiv:2305.18505*, 2023. 14

[52] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024. 14

[53] Xiaoying Zhang, Jean-Francois Ton, Wei Shen, Hongning Wang, and Yang Liu. Overcoming reward overoptimization via adversarial policy optimization with lightweight uncertainty estimation. *arXiv preprint arXiv:2403.05171*, 2024. 15

[54] Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023. 14

[55] Han Zhong, Guhao Feng, Wei Xiong, Li Zhao, Di He, Jiang Bian, and Liwei Wang. Dpo meets ppo: Reinforced token optimization for rlhf. *arXiv preprint arXiv:2404.18922*, 2024. 14

[56] Banghua Zhu, Jiantao Jiao, and Michael I Jordan. Principled reinforcement learning with human feedback from pairwise or k-wise comparisons. *arXiv preprint arXiv:2301.11270*, 2023. 7, 14

[57] Banghua Zhu, Michael I Jordan, and Jiantao Jiao. Iterative data smoothing: Mitigating reward overfitting and overoptimization in rlhf. *arXiv preprint arXiv:2401.16335*, 2024. 2, 7, 14

[58] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. 1, 14

A Related Works

In the following, we relate our work to recent lines of RLHF research on both theory and practice sides. We also review related works on reward hacking and overoptimization in RLHF.

RLHF: algorithm design. The technique of RLHF [10, 58, 30, 5, 14] has recently demonstrated its great importance in building the state-of-the-art LLMs, including ChatGPT [1], Gemini [40], Claude [2]. In the RLHF pipeline therein, the LLM is fine-tuned towards maximizing a learned reward model using the RL algorithm Proximal Policy Optimization (PPO; [37]). Meanwhile, PPO-style algorithm is also known for its instability, sample-inefficiency, and especially, a high demand for proper hyperparameter tuning [17]. This thus casts prohibitive computational cost to make the most effectiveness of PPO-based RLHF methods to align LLMs, especially for the open-source community.

Given that, further research on RLHF has explored various alternatives to PPO-based methods, with the most popular approach being the direct preference optimization method [54, 34], which skips the reward learning phase and directly optimize the LLM to align it with the human preference. Our practical implementation (RPO) also harnesses the wisdom of reward-LLM equivalence to avoid explicit reward learning followed by PPO training.

Besides the original DPO algorithm [34], ever since it popularizing the direct preference learning style method, variants of the direct preference learning approach are proposed, including but not limited to [25, 4, 47, 39, 21, 48, 32, 20, 36, 24, 52, 38, 46, 22]. Each of them aims to address further challenges of direct preference learning from varying perspectives. Specifically, the algorithm proposed by [32, 20] share similar algorithmic components as RPO proposed in this work. Both work consider SFT style regularization during preference optimization. However, theoretical understanding of how SFT loss can help alignment remains unknown. In contrast, we provide theoretical justifications to the SFT loss as an implicit adversarial regularizer that provably mitigates overoptimization in preference learning.

RLHF: theoretical investigation. Initiated from the literature of dueling bandits and dueling RL [49, 6, 31], recent success of RLHF in fine-tuning LLMs also motivates a long line of research to investigate the theoretical foundations of RLHF under different settings [9, 56, 50, 51, 45, 23, 47, 48, 15, 55], aiming to propose provably sample-efficient algorithms to learn a human-reward-maximizing policy from human preference signals. Our theoretical study of RLHF falls into the paradigm of offline learning from a pre-collected preference dataset, and is mostly related to the work of [56, 50, 23, 47, 48]. In this setup, the main challenge is to address the overoptimization issues due to human reward uncertainty and distributional shifts when only a fixed dataset is available. In the sequel, we compare our work with them in more detail.

Existing theoretical work on provably sample-efficient offline RLHF typically suffers from two drawbacks: they are either restricted to the linear function approximations setting [56, 47] which is far from the practical situations, or are generally unable to be implemented in the LLM experiments. Typically, to encompass the pessimistic principle in the face of uncertainty, the existing literature proposes to return the optimal policy against either an estimated reward model plus a structure-aware reward uncertainty penalty [47] or the most pessimistic reward model inside a confidence region [56, 50]. Both of these two types of method involve intractable components for implementation and needs for additional algorithmic design to approximate the theoretical algorithm in practice. In contrast, our theory works in the context of general function approximations while being friendly to be implemented. Finally, we remark that, while our study focuses on the standard Bradley-Terry model of human preference with general reward function approximations, the work of [48] further considers a general human preference model. But it remains unknown how their algorithms can be efficiently implemented in practice. It serves as an interesting direction to extend our technique to RLHF with general reward model and device new practical algorithms.

Reward hacking and overoptimization in RLHF for LLM. As is discussed in the introduction, the challenge of reward hacking or overoptimization may prevent the successful alignment of LLMs, degenerating the performance of an LLM because of maximizing an imperfect, overfitted, and misgeneralized proxy reward learned from the finite data [28, 41, 19, 8]. Efforts have been made to mitigate this fundamental issue through the perspective of theory, e.g., [56, 47, 57], and practice, e.g.,

[11, 16, 29, 53, 35]. Our approach starts from the theoretical insights of handling inherent uncertainty in learning human preference from finite data, while being suprisingly easy to implement in practice.

B Limitations and Future Works

One limitation of the current work is that we focus on the setting of offline RLHF where only a fixed preference dataset is available. Recent RLHF research has shown great potential of using iterative methods for LLM alignment with multiple rounds of preference data collection and tuning [47, 38].

Future works include extending our idea of theoretical algorithm design and analysis to the iterative RLHF setup where further preference data can be collected. Also, since our practical algorithm RPO is a plug-in module that effectively mitigates overoptimization and improves alignment performance, it serves as an exciting direction to combine it with explorative preference data collecting mechanism in iterative RLHF to further boost the performance of LLM alignment.

C Proofs for Sample Complexity Analysis

C.1 Further Discussions on Algorithm 1 and Theorem 5.3

We further compare our theory with two related works [47] and [50].

Remark C.1 (Comparison with [47])**.** *Another theoretical work on RLHF [47] explicitly models the KL-regularization between the target policy and the reference policy in the learning objective, referred to as the KL-regularized contextual bandit. This means that their metric becomes the KL-regularized expected reward. In contrast, here we put the KL-regularization as a component of our algorithm design, but we still keep the metric as the expected reward (2.2). Therefore our theory in Section 5.1 directly reveals how the learned policy performs in terms of the expected reward compared to any given target policy (which can be a stochastic policy).*

Remark C.2 (Comparison with [50])**.** *We remark that in the work of [50], they also mentioned a maximin object similar to (3.2) for offline preference-based RL as a complementary to their theoretical algorithm. However, the sample complexity of the maximin-style algorithm they presented is unknown, while we provide finite sample convergence result for Algorithm 1 in Section 5. Furthermore, our objective (3.2) features another KL-regularization term, which is essential for the proposal of our new practical algorithm design for aligning LLM in Section 4.*

C.2 Proof of Theorem 5.3

Proof of Theorem 5.3. By definition, the suboptimality gap of $\widehat{\pi}$ w.r.t. π is decomposed as following,

$$
\begin{aligned}
&\mathrm{Gap}^{\pi}(\widehat{\pi}) \\
&\quad = \mathbb{E}_{x\sim d_0, a\sim\pi(\cdot|x)}\big[r^\star(x,a)\big] - \mathbb{E}_{x\sim d_0, a\sim\widehat{\pi}(\cdot|x)}\big[r^\star(x,a)\big] \\
&\quad = \mathbb{E}_{x\sim d_0, a^1\sim\pi(\cdot|x), a^0\sim\pi^{\mathrm{ref}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] \\
&\qquad - \eta^{-1}\cdot\min_{r\in\mathcal{R}}\left\{\eta\cdot\mathbb{E}_{\substack{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot\mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\} \\
&\qquad + \eta^{-1}\cdot\min_{r\in\mathcal{R}}\left\{\eta\cdot\mathbb{E}_{\substack{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot\mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\} \\
&\qquad - \mathbb{E}_{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot\mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] \\
&\qquad + \beta\cdot\mathbb{E}_{x\sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big) - \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] \\
&\quad := \text{Term (A)} + \text{Term (B)} + \text{Term (C)},
\end{aligned}
\tag{C.1}
$$

where in the above Term (A), Term (B), and Term (C) are abbreviations for
Term (A)

$$= \mathbb{E}_{x\sim d_0, a^1\sim\pi(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]$$
$$- \eta^{-1}\cdot \min_{r\in\mathcal{R}}\left\{\eta\cdot \mathbb{E}_{\substack{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\},$$

and
Term (B)

$$= \eta^{-1}\cdot \min_{r\in\mathcal{R}}\left\{\eta\cdot \mathbb{E}_{\substack{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\}$$
$$- \mathbb{E}_{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x), a^0\sim\pi^{\mathrm{base}(\cdot|x)}}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big],$$

and

$$\text{Term (C)} = \beta\cdot \mathbb{E}_{x\sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big) - \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big].$$

In the following, we analyze Term (A) and Term (B) respectively.

Upper bound Term (A). Notice that by the optimality of our choice of policy $\widehat{\pi}$ in (3.2), we have
Term (A)

$$= \mathbb{E}_{x\sim d_0, a^1\sim\pi(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] \tag{C.2}$$
$$- \eta^{-1}\cdot \min_{r\in\mathcal{R}}\left\{\eta\cdot \mathbb{E}_{\substack{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\}$$

$$\le \mathbb{E}_{x\sim d_0, a^1\sim\pi(\cdot|x), a^0\sim\pi^{\mathrm{ref}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]$$
$$- \eta^{-1}\cdot \min_{r\in\mathcal{R}}\left\{\eta\cdot \mathbb{E}_{\substack{x\sim d_0, a^1\sim\pi(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot \mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\}$$

$$= \max_{r\in\mathcal{R}}\left\{\mathbb{E}_{x\sim d_0, a^1\sim\pi(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[\big(r^\star(x,a^1) - r^\star(x,a^0)\big) - \big(r(x,a^1) - r(x,a^0)\big)\Big] - \eta^{-1}\cdot\mathcal{L}_{\mathcal{D}}(r)\right\},$$

where in the inequality we apply the optimality of the choice of policy $\widehat{\pi}$ in (3.2).

Upper bound Term (B). For this term, we directly consider the following bound,
Term (B)

$$= \eta^{-1}\cdot \min_{r\in\mathcal{R}}\left\{\eta\cdot \mathbb{E}_{\substack{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x),\\ a^0\sim\pi^{\mathrm{ref}}(\cdot|x)}}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r)\right\}$$
$$- \mathbb{E}_{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]$$
$$\le \mathbb{E}_{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \eta^{-1}\cdot\mathcal{L}_{\mathcal{D}}(r^\star)$$
$$- \mathbb{E}_{x\sim d_0, a^1\sim\widehat{\pi}(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r^\star(x,a^1) - r^\star(x,a^0) - \beta\cdot \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]$$
$$= \eta^{-1}\cdot\mathcal{L}_{\mathcal{D}}(r^\star), \tag{C.3}$$

where in the inequality we apply the fact that $r^\star \in \mathcal{R}$ by Assumption 5.1.

Combining Term (A), Term (B), and Term (C). Now by (C.1), (C.2), and (C.3), we have that

$$\begin{aligned}
&\mathrm{Gap}_\beta^\pi(\widehat{\pi}) = \text{Term (A)} + \text{Term (B)} + \text{Term (C)} \qquad \text{(C.4)}\\
&\leq \max_{r\in\mathcal{R}} \left\{ \mathbb{E}_{\substack{x\sim d_0, a^1\sim\pi(\cdot|x),\\ a^0\sim\pi^{\mathrm{base}}(\cdot|x)}} \Big[\big(r^\star(x,a^1) - r^\star(x,a^0)\big) - \big(r(x,a^1) - r(x,a^0)\big) \Big] + \eta^{-1}\cdot \Big(\mathcal{L}_\mathcal{D}(r^\star) - \mathcal{L}_\mathcal{D}(r)\Big) \right\}\\
&\quad + \beta\cdot \mathbb{E}_{x\sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big) - \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big].
\end{aligned}$$

In the following, we upper bound the right hand side of (C.4) via relating the MLE loss difference term to the reward difference term through a careful analysis of the preference model. On the one hand, we invoke Lemma C.3 to give an upper bound of the difference of the MLE loss as following, with probability at least $1-\delta$ over random samples and $\varepsilon = (6\cdot(1+e^R)\cdot N)^{-1}$, for any reward model $r\in\mathcal{R}$, it holds that

$$\begin{aligned}
&\mathcal{L}_\mathcal{D}(r^\star) - \mathcal{L}_\mathcal{D}(r)\\
&\quad \leq -2\cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_\mathcal{D}(\cdot,\cdot,\cdot)}\Big[D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big)\Big]\\
&\qquad + \frac{3}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right),
\end{aligned}$$

where we recall that we use the subscript r in $\mathbb{P}_r$ to emphasize the dependence of the probabilistic model on the reward model. Here $\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)$ denotes the ε-covering number of the reward model class and R is the upper bound on the reward functionss (Assumption 5.1). Now to facilitate the calculation, we lower bound the Hellinger distance by total variation (TV) distance as

$$D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big) \geq D^2_{\mathrm{TV}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big),$$

By the expression of the probability model $\mathbb{P}_r$, we can further write the TV distance above as

$$\begin{aligned}
&D_{\mathrm{TV}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big)\\
&\quad = \frac{1}{2}\cdot\Big|\sigma\big(r^\star(x,a^1)-r^\star(x,a^0)\big) - \sigma\big(r(x,a^1)-r(x,a^0)\big)\Big|\\
&\qquad + \frac{1}{2}\cdot\Big|\sigma\big(r^\star(x,a^0)-r^\star(x,a^1)\big) - \sigma\big(r(x,a^0)-r(x,a^1)\big)\Big|\\
&\quad = \Big|\sigma\big(r^\star(x,a^1)-r^\star(x,a^0)\big) - \sigma\big(r(x,a^1)-r(x,a^0)\big)\Big|, \qquad \text{(C.5)}
\end{aligned}$$

where in the second equality we use the fact that $\sigma(-z) = 1-\sigma(z)$. Now by Lemma C.4 and the condition that $r(x,a)\in[0,R]$ for any $(x,a,r)\in\mathcal{X}\times\mathcal{A}\times\mathcal{R}$ (Assumption 5.1), we know that

$$\begin{aligned}
&\Big|\sigma\big(r^\star(x,a^1)-r^\star(x,a^0)\big) - \sigma\big(r(x,a^1)-r(x,a^0)\big)\Big|\\
&\qquad \geq \kappa\cdot\Big|\big(r^\star(x,a^1)-r^\star(x,a^0)\big) - \big(r(x,a^1)-r(x,a^0)\big)\Big|,
\end{aligned}$$

where $\kappa = 1/(1+\exp(R))^2$. As a result, the difference of the MLE loss is upper bounded by

$$\begin{aligned}
&\mathcal{L}_\mathcal{D}(r^\star) - \mathcal{L}_\mathcal{D}(r)\\
&\quad \leq -2\kappa^2\cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_\mathcal{D}(\cdot,\cdot,\cdot)}\left[\Big|\big(r^\star(x,a^1)-r^\star(x,a^0)\big) - \big(r(x,a^1)-r(x,a^0)\big)\Big|^2\right]\\
&\qquad + \frac{3}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right). \qquad \text{(C.6)}
\end{aligned}$$

On the other hand, the reward difference term in (C.4), which is evaluated on actions from π and π^{base}, can be related to the reward difference evaluated on the data distribution $\mu_\mathcal{D}$ via Assumption 5.2, i.e.,

$$\mathbb{E}_{x\sim d_0, a^1\sim\pi(\cdot|x), a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[\big(r^\star(x,a^1)-r^\star(x,a^0)\big) - \big(r(x,a^1)-r(x,a^0)\big)\Big] \qquad \text{(C.7)}$$

$$\leq C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\mathrm{base}})\sqrt{\mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}}\left[\left|\big(r^\star(x,a^1)-r^\star(x,a^0)\big)-\big(r(x,a^1)-r(x,a^0)\big)\right|^2\right]}.$$

Finally, combining (C.6), (C.7), and (C.4), denoting

$$\Delta_r := \sqrt{\mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}}\left[\left|\big(r^\star(x,a^1)-r^\star(x,a^0)\big)-\big(r(x,a^1)-r(x,a^0)\big)\right|^2\right]},$$

we have that

$$\begin{aligned}
\mathrm{Gap}^{\pi}(\widehat{\pi}) &\leq \max_{r\in\mathcal{R}}\left\{C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\mathrm{base}})\cdot\Delta_r - 2\eta^{-1}\kappa^2\cdot\Delta_r^2\right\} + \frac{3}{\eta N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right)\\
&\quad + \beta\cdot\mathbb{E}_{x\sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big) - \mathrm{KL}\big(\widehat{\pi}(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]\\
&\leq \frac{\big(C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\mathrm{base}})\big)^2\eta}{8\kappa^2} + \frac{3}{\eta N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right)\\
&\quad + \beta\cdot\mathbb{E}_{x\sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big],
\end{aligned}$$

where in the second inequality we use that fact that $az - bz^2 \leq a^2/(4b)$ for any $z\in\mathbb{R}$ and that KL-divergence is non-negative. Consequently, with the choice of

$$\eta = 2\sqrt{6}\cdot\sqrt{\frac{\log\left(\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)/\delta\right)}{N}},\quad \beta = \frac{1}{\sqrt{N}},\quad \kappa = \frac{1}{(1+\exp(R))^2},$$

we conclude that with probability at least $1-\delta$ and $\varepsilon = (6\cdot(1+e^R)\cdot N)^{-1}$,

$$\begin{aligned}
&\mathrm{Gap}^{\pi}(\widehat{\pi})\\
&\quad\leq \frac{\sqrt{6}\big(1+\exp(R)\big)^2\left(\big(C_{\mu_{\mathcal{D}}}(\mathcal{R};\pi,\pi^{\mathrm{base}})\big)^2+1\right)\iota + 4\mathbb{E}_{x\sim d_0}\Big[\mathrm{KL}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big]}{4\sqrt{N}},
\end{aligned}$$

where we denote $\iota = \sqrt{\log\left(\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)/\delta\right)}$ with $\varepsilon = (6\cdot(1+e^R)\cdot N)^{-1}$. This finishes the proof of Theorem 5.3. □

C.3 Technical Lemmas

Lemma C.3 (Uniform concentration). *Consider the MLE loss (3.1) and define the approximation error as $\varepsilon = (6\cdot(1+e^R)\cdot N)^{-1}$ where R is the upper bound on the reward functions (Assumption 5.2). Suppose that the reward model class $\mathcal{R}$ has a finite ε-covering number $\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)<\infty$. Then for any $\delta<1/e$ it holds with probability at least $1-\delta$ that*

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(r^\star) - \mathcal{L}_{\mathcal{D}}(r)\\
&\quad\leq -2\cdot\mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big)\Big]\\
&\qquad + \frac{3}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right).
\end{aligned}$$

Proof of Lemma C.3. For notational simplicity, we use $\mathcal{C}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)$ to denote an ε-cover of the reward model class $\mathcal{R}$ under the $\|\cdot\|_\infty$-norm. It holds that $\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty) = |\mathcal{C}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)|$. First we invoke Proposition 5.3 of [27] to obtain a uniform concentration over the finite set of ε-cover $\mathcal{C}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)$. Specifically, with probability at least $1-\delta$, for any $r\in\mathcal{C}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)$,

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(r^\star) - \mathcal{L}_{\mathcal{D}}(r)\\
&\quad\leq -2\cdot\mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big)\Big]\\
&\qquad + \frac{2}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right).
\end{aligned}\tag{C.8}$$

Now for any reward model $r \in \mathcal{R}$, we take a $r^\dagger \in \mathcal{C}_\varepsilon(\mathcal{R}, \|\cdot\|_\infty)$ satisfying $\|r - r^\dagger\|_\infty \le \varepsilon$. We have

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(r^\star) - \mathcal{L}_{\mathcal{D}}(r) \\
&\quad = \mathcal{L}_{\mathcal{D}}(r^\star) - \mathcal{L}_{\mathcal{D}}(r^\dagger) + \mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r) \\
&\quad \le -2 \cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_{r^\dagger}(\cdot|x,a^1,a^0)\big)\Big] \\
&\qquad + \frac{2}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right) + \mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r) \\
&\quad \le -2 \cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_{r}(\cdot|x,a^1,a^0)\big)\Big] \\
&\qquad + \frac{2}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right) + \mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r) \\
&\qquad\quad + 4 \cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\dagger}(\cdot|x,a^1,a^0)\|\mathbb{P}_{r}(\cdot|x,a^1,a^0)\big)\Big], \qquad \text{(C.9)}
\end{aligned}$$

where in the fir inequality we use (C.8) for $r^\dagger$ and in the second inequality we utilize the triangular inequality for Hellinger distance. Therefore, it remains to upper bound the approximation error induced by $r^\dagger$. On the one hand, by the definition of $\mathcal{L}_{\mathcal{D}}$ in (3.1), we have that

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r) \\
&\quad = \frac{1}{N}\sum_{i=1}^N y_i \cdot \log\left(\frac{\sigma\big(r(x_i,a_i^1) - r(x_i,a_i^0)\big)}{\sigma\big(r^\dagger(x_i,a_i^1) - r^\dagger(x_i,a_i^0)\big)}\right) \\
&\qquad + \frac{1}{N}\sum_{i=1}^N (1-y_i)\cdot\log\left(\frac{\sigma\big(r(x_i,a_i^0) - r(x_i,a_i^1)\big)}{\sigma\big(r^\dagger(x_i,a_i^0) - r^\dagger(x_i,a_i^1)\big)}\right).
\end{aligned}$$

Use the inequality that $\log(x) \le x - 1$, we can further upper bound $\mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r)$ by

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r) \\
&\quad \le \frac{1}{N}\sum_{i=1}^N y_i \cdot \frac{\sigma\big(r(x_i,a_i^1) - r(x_i,a_i^0)\big) - \sigma\big(r^\dagger(x_i,a_i^1) - r^\dagger(x_i,a_i^0)\big)}{\sigma\big(r^\dagger(x_i,a_i^1) - r^\dagger(x_i,a_i^0)\big)} \\
&\qquad + \frac{1}{N}\sum_{i=1}^N (1-y_i)\cdot \frac{\sigma\big(r(x_i,a_i^0) - r(x_i,a_i^1)\big) - \sigma\big(r^\dagger(x_i,a_i^0) - r^\dagger(x_i,a_i^1)\big)}{\sigma\big(r^\dagger(x_i,a_i^0) - r^\dagger(x_i,a_i^1)\big)}.
\end{aligned}$$

Now since $\|r^\dagger - r\|_\infty \le \varepsilon$ and $r^\dagger \in [0, R]$, invoking Lemma C.4, we can derive that

$$\begin{aligned}
\mathcal{L}_{\mathcal{D}}(r^\dagger) - \mathcal{L}_{\mathcal{D}}(r) &\le \frac{1}{N}\sum_{i=1}^N \frac{\big|\big(r(x_i,a_i^1) - r(x_i,a_i^0)\big) - \big(r^\dagger(x_i,a_i^1) - r^\dagger(x_i,a_i^0)\big)\big|}{(1+e^R)^{-1}} \\
&\qquad + \frac{1}{N}\sum_{i=1}^N \frac{\big|\big(r(x_i,a_i^0) - r(x_i,a_i^1)\big) - \big(r^\dagger(x_i,a_i^0) - r^\dagger(x_i,a_i^1)\big)\big|}{(1+e^R)^{-1}} \\
&\le 4\cdot\|r^\dagger - r\|_\infty \cdot (1+e^R) \le 4\varepsilon\cdot(1+e^R). \qquad \text{(C.10)}
\end{aligned}$$

On the other hand, we upper bound the hellinger distance between $\mathbb{P}_r$ and $\mathbb{P}_{r^\dagger}$, for any $(x, a^1, a^0) \in \mathcal{X}\times\mathcal{A}\times\mathcal{A}$,

$$\begin{aligned}
&D^2_{\mathrm{Hellinger}}\big(\mathbb{P}_{r^\dagger}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big) \\
&\quad \le D_{\mathrm{TV}}\big(\mathbb{P}_{r^\dagger}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big) \\
&\quad = \Big|\sigma\big(r^\dagger(x,a^1) - r^\dagger(x,a^0)\big) - \sigma\big(r(x,a^1) - r(x,a^0)\big)\Big| \\
&\quad \le \Big|\big(r^\dagger(x,a^1) - r^\dagger(x,a^0)\big) - \big(r(x,a^1) - r(x,a^0)\big)\Big| \\
&\quad \le 2\cdot\|r^\dagger - r\|_\infty \le 2\varepsilon, \qquad \text{(C.11)}
\end{aligned}$$

where the first inequality uses the fact that $D^2_{\text{Hellinger}} \le D_{\text{TV}}$, the equality uses the same argument as (C.5), and the second inequality applies Lemma C.4. Finally, combining (C.9), (C.10), and (C.11), we conclude that

$$\mathcal{L}_{\mathcal{D}}(r^\star) - \mathcal{L}_{\mathcal{D}}(r) \le -2 \cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\text{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big)\Big] + \frac{2}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right) + 6\varepsilon\cdot(1+e^R).$$

By taking the approximation error $\varepsilon = (6\cdot(1+e^R)\cdot N)^{-1}$, we conclude that for $\delta < e^{-1}$, with probability at least $1-\delta$, for any $r \in \mathcal{R}$, it holds that

$$\begin{aligned}&\mathcal{L}_{\mathcal{D}}(r^\star) - \mathcal{L}_{\mathcal{D}}(r)\\ &\quad\le -2 \cdot \mathbb{E}_{(x,a^1,a^0)\sim\mu_{\mathcal{D}}(\cdot,\cdot,\cdot)}\Big[D^2_{\text{Hellinger}}\big(\mathbb{P}_{r^\star}(\cdot|x,a^1,a^0)\|\mathbb{P}_r(\cdot|x,a^1,a^0)\big)\Big]\\ &\qquad+\frac{3}{N}\cdot\log\left(\frac{\mathcal{N}_\varepsilon(\mathcal{R},\|\cdot\|_\infty)}{\delta}\right).\end{aligned}$$

This completes the proof of Lemma C.3. □

Lemma C.4 (Sigmoid function)**.** *For any real numbers $z_1, z_2 \in [-R, R]$, it holds that*

$$\kappa\cdot|z_1 - z_2| \le |\sigma(z_1) - \sigma(z_2)| \le |z_1 - z_2|,$$

where the constant $\kappa = 1/(1+\exp(R))^2$.

Proof of Lemma C.4. Since the sigmoid function $\sigma(\cdot)$ is differentiable, we know that for any $z_1, z_2 \in [-R, R]$, there exists some $\xi(z_1, z_2) \in [-R, R]$ such that

$$\sigma(z_1) - \sigma(z_2) = \sigma'\big(\xi(z_1,z_2)\big)\cdot(z_1 - z_2).$$

Notice that $\sigma'(z) = \sigma(z)\cdot(1-\sigma(z))$, we can obtain that

$$\begin{aligned}1 \ge \sigma'\big(\xi(z_1,z_2)\big) &= \sigma\big(\xi(z_1,z_2)\big)\cdot\Big(1-\sigma\big(\xi(z_1,z_2)\big)\Big)\\ &= \frac{1}{1+\exp(\xi(z_1,z_2))}\cdot\left(1-\frac{1}{1+\exp(\xi(z_1,z_2))}\right)\\ &\ge \frac{1}{1+\exp(R)}\cdot\left(1-\frac{1}{1+\exp(-R)}\right)\\ &= \frac{1}{(1+\exp(R))^2}.\end{aligned}$$

This completes the proof of Lemma C.4. □

D Proofs for Equivalence between Maximin and Minimax Objectives

D.1 Proof of Theorem 5.6

Proof of Theorem 5.6. Consider denoting an auxiliary policy $\widehat{\pi}$ as

$$\widehat{\pi} \in \operatorname*{argmax}_{\pi\in\Pi}\min_{r\in\mathcal{R}}\phi(\pi, r). \tag{D.1}$$

By the definition of $\widehat{r}$ and $\widehat{\pi}$, the duality gap of $(\widehat{r}, \widehat{\pi})$, defined as

$$\text{Dual}(\widehat{r},\widehat{\pi}) := \max_{\pi\in\Pi}\phi(\pi,\widehat{r}) - \min_{r\in\mathcal{R}}\phi(\widehat{\pi}, r)$$

is zero. This is because the following deduction,

$$\begin{aligned}\text{Dual}(\widehat{r},\widehat{\pi}) &= \left(\max_{\pi\in\Pi}\phi(\pi,\widehat{r}) - \min_{r\in\mathcal{R}}\max_{\pi\in\Pi}\phi(\pi,r)\right)\\ &\quad+\left(\max_{\pi\in\Pi}\min_{r\in\mathcal{R}}\phi(\pi,r) - \min_{r\in\mathcal{R}}\phi(\widehat{\pi},r)\right)\\ &= 0,\end{aligned} \tag{D.2}$$

where in the first equality we apply Lemma D.1 that the minimax objective and the maximin objective are equivalent, and the last equality applies the definition of $\widehat{r}$ and $\widehat{\pi}$ respectively. Note that we can rewrite the duality gap as following

$$\mathrm{Dual}(\widehat{r},\widehat{\pi}) = \left(\max_{\pi\in\Pi}\phi(\pi,\widehat{r}) + \phi(\widehat{\pi},\widehat{r})\right) - \left(\phi(\widehat{\pi},\widehat{r}) - \min_{r\in\mathcal{R}}\phi(\widehat{\pi},r)\right). \tag{D.3}$$

Combining (D.2) and (D.3), we can conclude that

$$\max_{\pi\in\Pi}\phi(\pi,\widehat{r}) = \phi(\widehat{\pi},\widehat{r}) \quad \Rightarrow \quad \widehat{\pi} \in \operatorname*{argmax}_{\pi\in\Pi}\phi(\widehat{r},\pi). \tag{D.4}$$

Now comparing what $\pi_{\widehat{r}}$ and $\widehat{\pi}$ satisfy in (5.4) and (D.4) respectively, invoking Lemma D.3 that the maximizer of $\phi(\cdot,r)$ given any $r\in\mathcal{R}$ is unique on the support of d_0, we can conclude that

$$\pi_{\widehat{r}}(\cdot|x) = \widehat{\pi}(\cdot|x), \quad \forall x \in \mathrm{Supp}(d_0). \tag{D.5}$$

Therefore, by (D.1) and (D.5), and the fact that $\phi(\pi,r)$ depends on π only through its value on the support of d_0, we can conclude that

$$\pi_{\widehat{r}} \in \operatorname*{argmax}_{\pi\in\Pi}\min_{r\in\mathcal{R}}\phi(\pi,r).$$

This finishes the proof of Theorem 5.6. □

D.2 Auxiliary Lemmas

Lemma D.1 (Equivalence of maximin and minimax objectives)**.** *For the policy class Π defined in (2.3) and the reward model class $\mathcal{R}$ satisfying Assumption 5.5, it holds that the maximin objective is equivalent to the minimax objective, i.e.,*

$$\max_{\pi\in\Pi}\min_{r\in\mathcal{R}}\phi(\pi,r) = \min_{r\in\mathcal{R}}\max_{\pi\in\Pi}\phi(\pi,r).$$

Proof of Lemma D.1. The foundation of this result is a minimax theorem given by [18] (Lemma D.2). In our setting, the policy class Π is a nonempty set, and the reward model class $\mathcal{R}$ is a nonempty compact Hausdorff space. Furthermore, by our choice of the policy class Π in (2.3), Π is a convex set. Meanwhile, the function ϕ is a concave function of $\pi\in\Pi$ since the dependence on π is linear terms plus a negative KL term (concave). Finally, by our assumption, the function ϕ is convex-like on the reward model class $\mathcal{R}$ and is also continuous on $\mathcal{R}$. As a result, all the conditions of Lemma D.2 are satisfied and the minimax theorem holds in our problem setup, finishing the proof of Lemma D.1. □

Lemma D.2 (Minimax theorem [18])**.** *Let $\mathcal{X}$ be a nonempty set (not necessarily topologized) and $\mathcal{Y}$ be a nonempty compact topological space. Let $f:\mathcal{X}\times\mathcal{Y}\mapsto\mathbb{R}$ be lower semicontinuous on $\mathcal{Y}$. Suppose that f is concave-like on $\mathcal{X}$ and convex-like on $\mathcal{Y}$, i.e., for any $x_1,x_2\in\mathcal{X}$, $\alpha\in[0,1]$, there exists $x_3\in\mathcal{X}$ such that*

$$f(x_3,\cdot) \geq \alpha\cdot f(x_1,\cdot) + (1-\alpha)\cdot f(x_2,\cdot) \text{ on } \mathcal{Y},$$

and for any $y_1,y_2\in\mathcal{Y}$, $\beta\in[0,1]$, there exists $y_3\in\mathcal{Y}$ such that

$$f(\cdot,y_3) \leq \beta\cdot f(\cdot,y_1) + (1-\beta)\cdot f(\cdot,y_2) \text{ on } \mathcal{Y}.$$

Then the following equation holds,

$$\max_{x\in\mathcal{X}}\min_{y\in\mathcal{Y}} f(x,y) = \min_{y\in\mathcal{Y}}\max_{x\in\mathcal{X}} f(x,y).$$

Lemma D.3 (Unique maximizer of ϕ)**.** *Consider the function ϕ defined as*

$$\phi(\pi,r) := \eta\cdot\mathbb{E}_{x\sim d_0,a^1\sim\pi(\cdot|x),a^0\sim\pi^{\mathrm{base}}(\cdot|x)}\Big[r(x,a^1) - r(x,a^0) - \beta\cdot D_{\mathrm{KL}}\big(\pi(\cdot|x)\|\pi^{\mathrm{ref}}(\cdot|x)\big)\Big] + \mathcal{L}_{\mathcal{D}}(r).$$

Then given any $r\in\mathcal{R}$, the maximimzer of $\phi(\cdot,r)$ is unique on the support of d_0.

Proof of Lemma D.3. Given any $r \in \mathcal{R}$, consider that

$$\begin{aligned}
&\max_{\pi \in \Pi} \phi(\pi, r) \\
&\quad = \eta \cdot \max_{\pi \in \Pi} \left\{ \mathbb{E}_{x \sim d_0, a^1 \sim \pi(\cdot|x)} \left[r(x, a^1) - \beta \cdot D_{\mathrm{KL}}\big(\pi(\cdot|x) \| \pi^{\mathrm{ref}}(\cdot|x)\big) \right] \right\} \\
&\quad = \eta \cdot \max_{\pi \in \Pi} \left\{ C_r - \beta \cdot \mathbb{E}_{x \sim d_0} \left[D_{\mathrm{KL}} \left(\pi(\cdot|x) \middle\| \frac{\pi^{\mathrm{ref}}(\cdot|x) \cdot \exp(\beta^{-1} \cdot r(x, \cdot))}{\int_{a' \in \mathcal{A}} \mathrm{d}\pi^{\mathrm{ref}}(a'|x) \cdot \exp(\beta^{-1} \cdot r(x, a'))} \right) \right] \right\},
\end{aligned}$$

where

$$C_r = \mathbb{E}_{x \sim d_0} \left[\beta \cdot \log \left(\int_{a \in \mathcal{A}} \mathrm{d}\pi^{\mathrm{ref}}(a|x) \cdot \exp\left(\beta^{-1} \cdot r(x, a)\right) \right) \right]$$

is a constant independent of π. Therefore, the maximizer of $\phi(\cdot, r)$ on the support of d_0 must equal to

$$\pi_r(\cdot|x) = \frac{\pi^{\mathrm{ref}}(\cdot|x) \cdot \exp(\beta^{-1} \cdot r(x, \cdot))}{\int_{a' \in \mathcal{A}} \mathrm{d}\pi^{\mathrm{ref}}(a'|x) \cdot \exp(\beta^{-1} \cdot r(x, a'))},$$

which completes the proof of Lemma D.3. □

E Additional Details on Experiments

E.1 Training Details

We train the gemma series models with 8 NVIDIA A6000 GPUs and the beta series models with 8 NVIDIA A100 GPUs, where they are all GPT-like models with around 7 billion parameters. It takes around three hours to train a beta series model and five hours to train a gemma one. Our codebase is adapted from the Alignment Handbook [42]. By comparing the validation loss on the test split (not used for later evaluation), we select the hyperparameter η of both RPO (beta) and RPO (gemma) to be 0.005. We list the remaining training configurations in Table 3, which are recommended by the Alignment Handbook.

Configuration	Beta Series	Gemma Series
learning rate	5.0e-7	5.0e-7
learning scheduler type	cosine	cosine
warmup ratio	1.0	1.0
batch size	128	128
gradient accumulation	2	16
batch size per device	8	1
training epoch	1	2
β	0.01	0.05
optimizer	adamw torch	adamw torch
seed	42	42
precision	bfloat16	bfloat16

Table 3: Training configurations for beta series and gemma series models in this paper.

E.2 Evaluation Details

GPT-4 evaluation on the test split. We use the following prompts to guide GPT-4 to annotate the preferences among win, lose, and tie (we denote them by A, B, and C, respectively).

Prompts: Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's question better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. Begin your evaluation by comparing the two responses and provide a short explanation. Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: [[A]] if assistant A is better, [[B]] if assistant B is better, and [[C]] for a tie. [Instruction] instruction [The Start of Assistant A's Answer] {*answer A*} [The End of Assistant A's Answer] [The Start of Assistant B's Answer] {*answer B*} [The End of Assistant B's Answer]

Here, we replace {*answer A*} and {*answer B*} with the answers of two models. Since GPT annotation has shown to prefer the answer in the first position [44], we randomly exchange the positions between two answers during the evaluation to ensure a fair comparison.

Benchmark evaluation. We use the default configuration for the evaluations on MT-Bench[1] and AlpacaEval 2.0[2]. By default, the annotator of MT-Bench is the *latest version* of GPT-4. The default annotator and the competitor model are both GPT-4 (Preview 11/06). We only need to manually import the proper chat template that formats the training dataset, which are shown as follows.

Chat Template for Beta Series: <|system|></s><|user|>
{*instruction*}</s>
<|assistant|>

Chat Template for Gemma Series: <bos> <|im_start|>user
{*instruction*}<|im_end|>
<|im_start|>assistant

E.3 Additional Results on Experiments

In this section, we provide the additional results to show the performance gain for RPO (beta) in MT-Bench and RPO (gemma) in AlpacaEval 2.0. We report the pairwise win rates in Tables 4, 5, and 6 to analyze their performance gaps, where all the annotation configurations are the same in Table 2. Results show that RPO still exceeds DPO in the metric of the pairwise win rates on the benchmarks for both beta series and gemma series.

win rate (%)	RPO (beta)	Ref. (beta)	DPO (beta)
RPO (beta)	50.00	**83.75**	**57.81**
Ref. (beta)	16.25	50.00	21.25
DPO (beta)	78.75	42.19	50.00

Table 4: Pairwise win rates (left vs. right) for beta series models on MT-Benchmark.

[1] https://github.com/lm-sys/FastChat/tree/main/fastchat/llm_judge
[2] https://github.com/tatsu-lab/alpaca_eval/tree/main

win rate (%)	RPO (beta)	Ref. (beta)	DPO (beta)
RPO(beta)	50.00	**80.13**	**52.02**
Ref.(beta)	19.87	50.00	20.61
DPO (beta)	47.98	79.39	50.00

Table 5: Pairwise win rates (left vs. right) for gemma series models on AlpacaEval 2.0.

win rate (%)	RPO (beta)	Ref. (beta)	DPO (beta)
RPO (beta)	50.00	**64.93**	**51.33**
Ref. (beta)	35.07	50.00	36.44
DPO (beta)	48.67	64.56	50.00

Table 6: Pairwise Length-Control (LC) win rates (left vs. right) for gemma series models on AlpacaEval 2.0.