DPO Meets PPO: Reinforced Token Optimization for RLHF

Han Zhong [*1] **Guhao Feng** [*1] **Wei Xiong** [*2] **Xinle Cheng** [1] **Li Zhao** [3] **Di He** [1] **Jiang Bian** [3] **Liwei Wang** [1]

Abstract

In the classical Reinforcement Learning from Human Feedback (RLHF) framework, Proximal Policy Optimization (PPO) is employed to learn from sparse, sentence-level rewards—a challenging scenario in traditional deep reinforcement learning. Despite the great successes of PPO in the alignment of state-of-the-art closed-source large language models (LLMs), its open-source implementation is still largely sub-optimal, as widely reported by numerous research studies. To address these issues, we introduce a framework that models RLHF problems as a Markov decision process (MDP), enabling the capture of fine-grained token-wise information. Furthermore, we provide theoretical insights that demonstrate the advantage of our MDP framework over the previous sentence-level bandit formulation. Under this framework, we introduce an algorithm, dubbed as Reinforced Token Optimization (`RTO`), which learns the token-wise reward function from preference data and performs policy optimization based on this learned token-wise reward signal. Theoretically, `RTO` is proven to have the capability of finding the near-optimal policy sample-efficiently. For its practical implementation, `RTO` integrates Direct Preference Optimization (DPO) and PPO. DPO, originally derived from sparse sentence rewards, surprisingly provides us with a token-wise characterization of response quality, which is seamlessly incorporated into our subsequent PPO training stage. Extensive real-world alignment experiments verify the effectiveness of the proposed approach.

[*]Equal contribution [1]Peking University [2]University of Illinois Urbana-Champaign [3]Microsoft Research Asia. Correspondence to: Han Zhong <hanzhong@stu.pku.edu.cn>.

Proceedings of the 41st ICML 2024 Workshop on Models of Human Feedback for AI Alignment, Vienna, Austria. Copyright 2024 by the author(s).

1. Introduction

Reinforcement Learning from Human Feedback (RLHF) has emerged as a key technique for aligning foundation models with human values and preferences (Christiano et al., 2017; Ziegler et al., 2019). It has been pivotal in enabling Large Language Models (LLMs) to produce more helpful, harmless, and honest responses (Bai et al., 2022), as demonstrated in significant applications such as ChatGPT (OpenAI, 2023), Claude (Anthropic, 2023), and Gemini (Team et al., 2023). The classical RLHF pipeline (Ziegler et al., 2019; Ouyang et al., 2022) consists of two steps: (i) Reward training from human feedback, where the learner learns the reward function based on preference data, typically through Maximum Likelihood Estimation (MLE). (ii) Reward-based RL training, where the learner employs the seminal deep RL algorithm Proximal Policy Optimization (PPO; Schulman et al., 2017) to optimize the reward learned in the previous step.

Despite the success of this framework in the aforementioned powerful closed-source LLMs, the training of PPO is known to be unstable and sample-inefficient (Choshen et al., 2019) compared to supervised learning. PPO frequently fails to maintain a consistent average response length or experiences sudden drops in reward value. Moreover, the superior performance of PPO also relies on the code-level optimization and an appropriate configuration of the hyper-parameters (Engstrom et al., 2020), while the training stability issue further prohibits us from achieving the best performance of the PPO. So far, the success of PPO has not been widely reproduced, especially in the open-source community with rather limited resources. While researchers have made efforts to propose alternative approaches to the PPO algorithm, with notable examples like rejection sampling fine-tuning (Dong et al., 2023; Gulcehre et al., 2023), direct preference learning algorithms (Rafailov et al., 2023; Zhao et al., 2023; Azar et al., 2023), there is little evidence that these newly proposed approaches alone can make the state-of-the-art LLMs. Therefore, improving the performance of the PPO algorithm in the context of RLHF is still an important research direction that is largely under-explored.

After examining the open-source implementation of PPO, we identify that one potential reason for the sub-optimal performance of PPO is the mismatch between the formulation

of RLHF and the nature of PPO. Specifically, in the existing framework (Ouyang et al., 2022; Bai et al., 2022), RLHF is formulated as a *bandit*, where the entire response sentence is considered to be an action, and the reward is sentence-level, evaluating only the overall quality of the response. However, PPO is designed for multi-step RL problems modeled as *Markov decision processes* (MDPs), requiring a token-wise reward assignment to each step. In typical implementations of PPO (e.g., the TRL package from huggingface[1]), besides the regularization reward function assigned to each token to ensure the fine-tuned LLM stays close to the supervised fine-tuning (SFT) model, the learned sentence-level reward is only distributed to the last token, while other tokens receive zero learned reward. See (2.3) for the formal mathematical description. Clearly, there is a separation in terms of the assignment strategies of the regularization reward and the learned reward. Meanwhile, while it is generally believed that a fine-grained characterization with token-wise feedback can provide more information, in practice, it is also challenging to collect effective token-wise feedback for human conversations and use it in the MLE process. Consequently, the construction of token-wise reward signals also remains largely under-explored in the literature of RLHF.

Our Contributions. We aim to address the aforementioned issues by developing an RLHF framework with a fine-grained token-wise reward characterization, establishing the mathematical foundation, and advancing practical algorithmic designs. Our contributions are summarized as follows.

- We propose a framework that models RLHF as an MDP, offering a more precise token-wise characterization of the LLM's generation process. Furthermore, we provide theoretical insights into why the token-wise MDP formulation is superior to the previous sentence-level bandit formulation.
- Under the MDP formulation of RLHF, we introduce Reinforced Token Optimization (`RTO`), which extracts token-wise reward signals from offline preference data and subsequently performs RL training with respect to the learned token-wise rewards. Using MLE as the token-wise reward learning oracle, we prove that `RTO` can learn a near-optimal policy in a sample-efficient manner.
- Moving toward the practical implementation of `RTO`, we adopt a novel token-wise reward extraction approach from direct preference optimization (DPO; Rafailov et al., 2023). By assigning this DPO-based token-wise reward function to each token and then optimizing with PPO, `RTO` outperforms existing baselines such as PPO and DPO in the task of dialogue.

In summary, under the MDP formulation of RLHF, we develop a new principled RLHF algorithm, `RTO`, that leverages token-wise reward signals derived from offline preference data using DPO, and subsequently performs PPO training to optimize the token-wise rewards. The pipeline of `RTO` is visualized in Figure 1.

Related Works. We have discussed the mostly related work PPO (Schulman et al., 2017) and DPO (Rafailov et al., 2023) in the introduction. More related works on RLHF is deferred to Appendix A.

Notation. Given a set $\mathcal{X}$, we denote the collection of distributions over $\mathcal{X}$ by $\Delta(\mathcal{X})$. We use $\mathbb{1}\{\cdot\}$ to denote the indicator function. For any positive integer h, we use the notation $y_{1:h}$ to denote the sequence $\{y_1, y_2, \ldots, y_h\}$.

2. Preliminaries

In this section, we introduce the standard RLHF paradigm. Let $x \in \mathcal{X}$ denote the prompt sampled from a distribution $\rho \in \Delta(\mathcal{X})$, and $y = (y_1, y_2, \ldots, y_h, \ldots)$ be the corresponding response, which is a sequence of tokens generated by LLMs, where y_i represents the i-th token. In practice, it is widely assumed (Christiano et al., 2017; Ziegler et al., 2019; Bai et al., 2022; Ouyang et al., 2022; Touvron et al., 2023) that the preference signal is generated according to the Bradley-Terry (BT) model (Bradley and Terry, 1952):

$$\begin{aligned}&\mathbb{P}(y^1 \succ y^2 | x, y^1, y^2)\\ =&\frac{\exp(r(x, y^1))}{\exp(r(x, y^1)) + \exp(r(x, y^2))}\\ =&\sigma\big(r(x, y^1) - r(x, y^2)\big),\end{aligned} \tag{2.1}$$

where $\sigma(z) = 1/(1+\exp(-z))$ is the sigmoid function, and r is a ground-truth reward function defined at the **sentence level**. In other words, the reward function r only evaluates the overall performance of the entire response. The classical RLHF pipeline (Ziegler et al., 2019; Ouyang et al., 2022) typically consists of two steps: reward training from human feedback and reward-based RL training. In the first step, the learner is given a dataset $\mathcal{D} = \{(x, y^w, y^l)\}$, where y^w denotes the preferred response over the y^l. The reward function is learned through Maximal Likelihood Estimation (MLE) on this dataset $\mathcal{D}$:

$$r_{\mathrm{MLE}} = \operatorname*{argmax}_r \mathbb{E}_{(x,y^w,y^l)\sim\mathcal{D}}\big[\log\big(\sigma(r(x, y^w) - r(x, y^l))\big)\big]. \tag{2.2}$$

In the second step, the learned reward r_{MLE} from the previous step is optimized while ensuring that the updated language model (LLM) does not deviate significantly from the reference model π_{ref}, usually selected as a supervised fine-tuned (SFT) LLM. This is because reward optimization along usually leads to reward hacking (Casper et al., 2023), meaning that the LLM will utilize the imperfection of the reward model and chase for a high reward but with a poor performance at the same time. Formally, the LLM is optimized with respect to the

[1] https://github.com/huggingface/trl

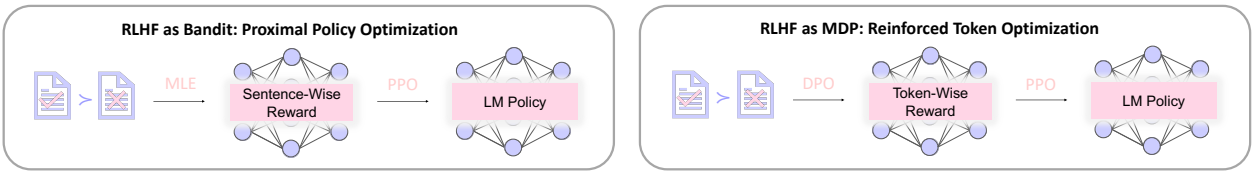


Figure 1. In the MDP framework of RLHF, RTO uses DPO to derive a token-level reward function and then applies PPO to enhance it. This approach is significantly different from the traditional RLHF process, which employs PPO to improve sentence-level rewards under the bandit framework.

learned reward r_{MLE} with a KL-regularized term: $\widehat{\pi} = \operatorname{argmax}_{\pi} \mathbb{E}_{x\sim\rho, y\sim\pi(\cdot|x)}[r_{\mathrm{MLE}}(x,y) - \beta \log \frac{\pi(y\,|\,x)}{\pi_{\mathrm{ref}}(y\,|\,x)}]$, where $\beta > 0$ is an appropriate KL penalty coefficient. This KL-regularized target is widely adopted in practice (Christiano et al., 2017; Ziegler et al., 2019; Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2023) to balance reward optimization and the goal of staying close to the reference policy. Another primary technical reason is that this regularization ensures that the framework admits a stochastic optimal policy, as compared to the deterministic greedy reward maximizer. The policy optimization step is typically achieved by PPO (Schulman et al., 2017), a seminal deep RL algorithm for solving multi-step decision-making problems and its implementation requires a reward signal at each step (corresponding to each token in the context of LLMs). To this end, given a prompt x and a response $y = y_{1:H}$ containing H tokens, existing open-source implementations of PPO assign the sentence-level reward $r_{\mathrm{MLE}}(x,y)$ to the last token and optimize the following reward:

$$r_{\mathrm{ppo}}(x, y_{1:h}) = \begin{cases} 0 - \beta \log \frac{\pi(y_h\,|\,x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h\,|\,x, y_{1:h-1})} \\ \text{if } h \le H-1, \\ r_{\mathrm{MLE}}(x,y) - \beta \log \frac{\pi(y_h\,|\,x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h\,|\,x, y_{1:h-1})} \\ \text{if } h = H, \end{cases} \tag{2.3}$$

where π is the current policy to be improved. However, it is well known that sparse rewards can make learning more difficult compared to dense rewards (Andrychowicz et al., 2017). One natural solution is to design dense token-wise rewards used for PPO training, but this is beyond the scope of the current bandit formulation for RLHF and motivates us to provide a framework with more fine-grained token-wise characterization that enables the use of token-wise rewards.

3. Formulation for RLHF: From Bandit to MDP

In this section, we introduce our MDP formulation for RLHF and demonstrate its advantages.

3.1. MDP Formulation for RLHF

We model the RLHF problem as a Markov decision process (MDP), which is denoted as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho, H)$. Here $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S}\times\mathcal{A} \to \Delta(\mathcal{S})$ is the transition kernel, r denotes the reward function, ρ signifies the initial state distribution and H is the maximal number of interaction steps. A (Markov) policy in MDPs $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ is a mapping from state to a distribution over actions. The interaction between the environment $\mathcal{M}$ and the agent can be described as follows. Initially, the starting state s_1 is sampled from the initial distribution ρ. At the h-th step, the agent observes the state s_h and selects an action a_h based on its policy. The environment then transits to the next state $s_{h+1} \sim \mathcal{P}(\cdot\,|\,s_h, a_h)$. This interaction continues until a certain ending condition is satisfied, which will be triggered within H steps.

In the standard text generation process of LLMs, each state $s_h = (x, y_{1:h-1})$ includes the prompt x and all response tokens produced up to that point. Each action $a_h = y_h$ represents a token from the vocabulary. The transition kernel $\mathcal{P}$ is usually known and deterministic, meaning that given tokens $s_h = (x, y_{1:h-1})$ and $a_h = y_h$, the environment will transition to $s_{h+1} = (x, y_{1:h})$. The policy π maps all the observed tokens so far to a distribution over the vocabulary. It is important to note that the policy captures the autoregressive nature of LLMs, i.e., $\pi(y_{1:h}\,|\,x) = \prod_{i=1}^{h} \pi(y_i\,|\,x, y_{1:i-1})$ for any h. Moreover, $r : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ represents the token-wise reward. The maximum number of tokens that can be generated, H, characterizes the length limit for LLM outputs. Each generated text ends with a special end-of-sentence token EoS, which terminates the generation process.

In our MDP formulation for RLHF, we also model the preference signal using BT model (Bradley and Terry, 1952). In specific, for any trajectory pair $\tau^1 = \{(s_h^1, a_h^1)\}_{h=1}^H$ and $\tau^2 = \{(s_h^2, a_h^2)\}_{h=1}^H$[2], the probability $\mathbb{P}(\tau^1 \succ \tau^2)$ of τ^1 is preferred over τ^2 is specified by

$$\begin{aligned} &\frac{\exp(\sum_{h=1}^H r(s_h^1, a_h^1))}{\exp(\sum_{h=1}^H r(s_h^1, a_h^1)) + \exp(\sum_{h=1}^H r(s_h^2, a_h^2))} \\ =&\sigma\Big(\sum_{h=1}^H r(s_h^1, a_h^1) - \sum_{h=1}^H r(s_h^2, a_h^2)\Big). \end{aligned} \tag{3.1}$$

[2]In fact, these two trajectories can have different lengths less than H. These trajectories can be extended to length H by assuming that the state ending with EoS is absorbing and yields zero reward. For simplicity, the following theoretical discussion may focus on length-H trajectories.

Compared to literature that formulates the RLHF problem as a contextual dueling bandit, a subtle difference is that the policy in the contextual dueling bandit maps a prompt to a distribution over sentences, which does not capture the autoregressive nature of LLMs. In contrast, our MDP formulation precisely captures this nature. We defer the discussion of these two types of policies in Section E.2. More importantly, the main difference is that the reward function in the MDP formulation is defined on a token level, which contrasts significantly with the sentence-level reward in the contextual dueling bandit. We discuss the advantages of token-level rewards in Section 3.2.

Learning Objective. Different from classical RL literature, where the sole goal is to maximize the reward function, the objective of RLHF is to maximize the reward function while ensuring that the learned policy does not deviate too much from the reference model (e.g., SFT model) too much. Inspired by this and the formulation of entropy-regularized MDPs (Williams and Peng, 1991; Ziebart, 2010), for any policy π, we define its corresponding regularized value-function by

$$V_\beta^\pi(s;r) = \mathbb{E}_\pi\left[\sum_{h=1}^{\infty}\left(r(s_h,a_h) - \beta\cdot\log\frac{\pi(a_h\,|\,s_h)}{\pi_{\mathrm{ref}}(a_h\,|\,s_h)}\right)\Bigg|\, s_1 = s\right], \tag{3.2}$$

where the expectation $\mathbb{E}_\pi$ is taken with respect to the randomness incurred by the policy π. Here the summation ends when a certain condition is met. In particular, since we assume that the maximal length of the generated responses of LLMs is at most H, the summation in (3.2) is taken at most H steps. In the remaining part of this paper, we may use $\sum_{h=1}^{\infty}$ and $\sum_{h=1}^{H}$ interchangeably, as they mostly have the same meaning. The regularized Q-function Q_β^π of a policy π is related to the regularized value function V_β^π as

$$\begin{aligned} Q_\beta^\pi(s,a;r) &= r_\beta(s,a) + \mathbb{E}_{s'\sim\mathcal{P}(\cdot\,|\,s,a)}[V_\beta^\pi(s';r)],\\ V_\beta^\pi(s;r) &= \mathbb{E}_{a\sim\pi(\cdot\,|\,s)}[-\beta\log\pi(a\,|\,s) + Q_\beta^\pi(s,a;r)], \end{aligned} \tag{3.3}$$

where we denote $r_\beta(s,a) = r(s,a) + \beta\log\pi_{\mathrm{ref}}(a\,|\,s)$. Moreover, when it is clear from the context, we may omit the dependency of the ground-truth reward function r in $Q_\beta^\pi(s,a;r), V_\beta^\pi(s;r)$ and use the shorthand $Q_\beta^\pi(s,a), V_\beta^\pi(s)$. The regularized optimal policy π_β^* is the policy that maximizes the regularized value function defined in (3.2), and its corresponding optimal Q-function and value function are denoted as Q_β^* and V_β^*, respectively. By (3.3), it can be shown that

$$\pi_\beta^*(a\,|\,s) = \exp\{(Q_\beta^*(s,a) - V_\beta^*(s))/\beta\}. \tag{3.4}$$

Our learning objective is to find a near-optimal policy $\widehat{\pi}$ minimizing the following suboptimality gap:

$$\mathrm{SubOpt}(\widehat{\pi}) = \mathbb{E}_{s\sim\rho}[V_\beta^*(s) - V_\beta^{\widehat{\pi}}(s)] = V_\beta^*(\rho) - V_\beta^{\widehat{\pi}}(\rho), \tag{3.5}$$

where we use the shorthand $V_\beta^\pi(\rho) = \mathbb{E}_{s\sim\rho}[V_\beta^\pi(s)]$ for any policy π. We define the state and state-action visitation measure as $d^\pi(s) = \mathbb{E}_{s_1\sim\rho}[\sum_{h=1}^{\infty}\mathbb{P}(s_t = s\,|\,s_1)]$ and $d^\pi(s,a) = \mathbb{E}_{s_1\sim\rho}[\sum_{h=1}^{\infty}\mathbb{P}(s_h = s, a_h = a\,|\,s_1)]$. We also use the shorthand $d^* = d^{\pi_\beta^*}$ to for simplicity.

3.2. Advantages of Token-Wise MDP over Sentence-Wise Bandit

Intuitively, the distinction between token-based and trajectory-based rewards reflects the difference between sparse and dense reward settings. In the sparse reward scenario, exploration proves to be more challenging. To illustrate this, we focus on the deterministic MDP with an action set size of $A = |\mathcal{A}|$. We employ an autoregressive policy π^* to represent the policy of a powerful LLM, such as GPT-4. Fixing a prompt x, given responses $(y^1 = y_{1:H}^1, y^2 = y_{1:H}^2)$, the evaluation provided by π^* is $\mathbb{P}(y^1 \succ y^2\,|\,x, y_1, y_2) = \frac{\pi^*(y^1\,|\,x)}{\pi^*(y^1\,|\,x)+\pi^*(y^2\,|\,x)}$. By comparing this with the BT models of bandit in (2.1) and of our MDP formulation in (3.1), we observe that the sentence-wise reward r_s and token-wise as r_t can be specified by

$$\begin{aligned} r_s(x,y) &= \log\pi^*(y\,|\,x),\\ r_t((x,y_{1:h-1}),y_h) &= \log\pi^*(y_h\,|\,x,y_{1:h-1}). \end{aligned} \tag{3.6}$$

Intuitively, the responses that powerful LLMs tend to choose have higher rewards. In addition, we have $r_s(x,y) = \sum_{h=1}^{H} r_t((x,y_{1:h-1}),y_h)$. We also make the following natural assumption.

Assumption 3.1. There exists a response $y = y_{1:H}$ satisfying $\pi^*(y\,|\,x) \geq A^{-\xi}$.

By the pigeon-hole principle, there must be a response y such that $\pi^*(y\,|\,x) \geq A^{-H}$, implying that $\xi \leq H$. In practice, ξ is usually much smaller than H because the language model tends to choose the optimal response rather than making a random guess. Now, we define the interaction protocol and the sample complexity. The learner can determine a response $y = y_{1:H}$ and receive either $r_s(x,y)$ or $\{r_t((x,y_{1:h-1}),y_h)\}_{h=1}^H$, depending on whether the sentence-level reward or the token-wise reward is used. The sample complexity is defined as the number of responses and corresponding reward signals that need to be gathered to find the optimal response $y^* = y_{1:H}^*$ with length H.

Theorem 3.2. Suppose Assumption 3.1 holds. In the setting where only the sentence-wise reward r_s in (3.6) is accessible, finding the optimal response y^* requires a sample complexity of A^H. However, if token-reward signals r_t in (3.6) are available, there exists an algorithm that can find the optimal policy with sample complexity $A^{\min\{\xi+1,H\}}$. Our algorithm is elimination-based and the detailed proof is deferred to Appendix B. A simplified example is given in Figure 2. Since $\xi \ll H$ typically holds in practice, the

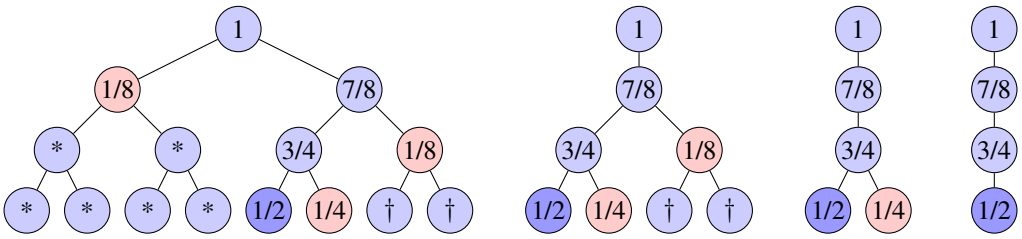


Figure 2. An illustration of our efficient learning algorithm for the token-wise reward setting with $A = 2$, $H = 3$, and $\xi = 1$. Here $*$ and $\dagger$ represent real numbers between 0 and 1/8. We do not specify their exact values as they do not influence the optimal path. All special nodes need to be eliminated are colored red, while other nodes are blue, with the optimal leaf node $1/2$ emphasized in dark blue. Each node $y_{1:h}$ is labelled with $\pi^*(y_{1:h} \mid x)$. If a non-optimal path (response) is selected, one red node in $\mathcal{N}$ will be identified, and all paths containing this node will be deleted. Here we visualize the process of choosing a path ending with $*$, $\dagger$, and $1/4$, respectively. At most $A^{\min\{\xi+1,H\}} = 4$ samples are needed to identify the optimal response.

gap between A^H and $A^{\min\{\xi+1,H\}}$ is deemed large. Hence, Theorem 3.2 reveals the significant separation of sample complexity between two types of reward signals, providing theoretical insights into the superiority of the token-wise MDP formulation over the sentence-wise bandit formulation of RLHF.

4. Reinforced Token Optimization

Under the MDP framework in Section 3, we develop the Reinforced Token Optimization (`RTO`) algorithmic framework. At a high level, `RTO` consists of two main steps: (i) token-wise reward learning, where `RTO` learns a token-wise reward based on the preference data; and (ii) optimizing token-wise reward through RL training methods such as PPO.

4.1. Theoretical Version with Sample Complexity Guarantee

We focus on the offline setting and assume the access to an offline dataset $\mathcal{D} = \{(\tau^w, \tau^l)\}$ that contains several trajectory pairs, where $\tau^w = \{(s_h^w, a_h^w)\}_{h=1}^H$ is preferred over $\tau^l = \{(s_h^l, a_h^l)\}_{h=1}^H$. Each pair of trajectories shares the same initial state/prompt (i.e., $s_1^w = s_1^l$), but differs in the subsequent tokens. We also assume that the reward function is linear, and our following results are ready to be extended to general function approximation (Chen et al., 2022; Zhan et al., 2023a).

Assumption 4.1 (Linear Reward)**.** We assume that the reward function r is linear, i.e., $r(s,a) = \phi(s,a)^\top \theta^*$ for some known feature $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ and unknown vector $\theta^* \in \mathbb{R}^d$. We also assume that $\|\phi(\cdot,\cdot)\|_2 \le L$ and $\|\theta^*\|_2 \le B$.

Following the standard reward learning pipeline (Ouyang et al., 2022), we learn the reward function via maximum likelihood estimation (MLE). Specifically, if we parametrize the reward function by θ, then the MLE is given by

$$\begin{aligned} \theta_{\mathrm{MLE}} &= \operatorname*{argmax}_{\|\theta\|_2 \le B} \mathcal{L}_{\mathcal{D}}(\theta), \\ \text{where} \quad \mathcal{L}_{\mathcal{D}}(\theta) &= \sum_{(\tau^w,\tau^l)\in\mathcal{D}} \\ &\Big[\log\Big(\sigma\big(\sum_{h=1}^H r_\theta(s_h^w, a_h^w) - \sum_{h=1}^H r_\theta(s_h^l, a_h^l)\big)\Big)\Big]. \end{aligned} \tag{4.1}$$

Inspired by previous literature in offline RL (Jin et al., 2021; Rashidinejad et al., 2021; Xiong et al., 2022; Zhu et al., 2023; Zhan et al., 2023a), given the MLE θ_{MLE}, we construct the pessimistic token-wise reward estimation as

$$\widehat{r}(s,a) = \phi(s,a)^\top \theta_{\mathrm{MLE}} - \varrho \cdot \|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}, \tag{4.2}$$

where $\Sigma_{\mathcal{D}} = \sum_{(\tau^1,\tau^2)\in\mathcal{D}} [\sum_{h=1}^H (\phi(s_h^1, a_h^1) - \phi(s_h^2, a_h^2))(\sum_{h=1}^H (\phi(s_h^1, a_h^1) - \phi(s_h^2, a_h^2)))^\top] + \lambda I_d$, $\lambda > 0$ is a tuning parameter, and ϱ is a problem-dependent coefficient will be specified in Theorem 4.2 and (C.2). Finally, `RTO` outputs the optimal policy $\widehat{\pi}$ with respect to $\widehat{r}$, i.e., $\widehat{\pi} = \operatorname{argmax}_\pi V_\beta^\pi(s; \widehat{r})$ for any $s \in \mathcal{S}$. The pseudocode of `RTO` is given in Algorithm 1. The following theorem provides theoretical guarantee for `RTO`, and the proof is deferred to Appendix C.

Theorem 4.2. Suppose Assumption 4.1 holds. For $\beta > 0$, $\lambda > 0$, $\delta \in (0,1)$, if we choose $\varrho = \widetilde{\mathcal{O}}(\sqrt{d})$ (see (C.2)), then the output policy $\widehat{\pi}$ of Algorithm 1 satisfies

$$\begin{aligned} \mathrm{SubOpt}(\widehat{\pi}) \le\ & 2\varrho \cdot \mathbb{E}_{(s,a)\sim d^*}\big[\|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big] \\ & - \beta \cdot \mathbb{E}_{s\sim d^*}\big[\mathrm{KL}\big(\pi_\beta^*(\cdot \mid s) \| \widehat{\pi}(\cdot \mid s)\big)\big]. \end{aligned} \tag{4.3}$$

The first term in Theorem 4.2 measures how well the offline dataset covers the trajectory generated by the policy π_β^*. Typically, this term decreases at a rate of $|\mathcal{D}|^{-1/2}$ under the mild partial coverage assumption (Jin et al., 2021; Uehara and Sun, 2021; Xiong et al., 2022; Zhu et al., 2023; Zhan et al., 2023a), where $|\mathcal{D}|$ is the size of the offline dataset. The second KL term is always negative, and it arises from the goal of learning a regularized value. We also remark that our algorithm relies on the known transition kernel to

Algorithm 1 Reinforced Token Optimization (Theoretical Version)

1: **Input:** Offline dataset $\mathcal{D}$, $\lambda > 0$, $\beta > 0$, and problem dependent coefficient ϱ.
2: Compute θ_{MLE} based on $\mathcal{D}$ by maximizing the loglikelihood given in (4.1).
3: Calculate the pessimistic reward $\widehat{r}$ via (4.2). ▷ token-wise reward learning
4: Compute the corresponding optimal policy $\widehat{\pi}$ with respect to $\widehat{r}$. ▷ optimizing token-wise reward
5: **Output:** policy $\widehat{\pi}$.

compute the exact optimal policy with respect to $\widehat{r}$. While this is natural in the context of large language models, we provide insights on how to extend our findings to stochastic regularized MDPs and the variant of our RTO algorithm in Appendix D.

There have also been previous works (Pacchiano et al., 2021; Chen et al., 2022; Wang et al., 2023; Li et al., 2023b; Zhan et al., 2023a) studying RLHF under the MDP framework, also known as dueling RL and preference-based RL. However, these works do not consider the KL constraint, which is an essential component of RLHF. Furthermore, they do not explicitly emphasize the superiority of the MDP framework over the contextual dueling bandit problem in the context of LLMs, and their proposed algorithms lack practical implementation. In contrast, we will provide a practical implementation of our algorithm, demonstrating the practicality of our approach.

4.2. Practical Implementation

In this subsection, we shift our focus to developing a practical version of RTO. The key challenge in implementing RTO in Algorithm 1 lies in learning the token-wise reward to be optimized from the offline data. In the most popular frameworks outlined in Instruct-GPT (Ouyang et al., 2022), Claude (Bai et al., 2022), and LLaMA2 (Touvron et al., 2023) projects replace the last layer of the LLM with a linear layer for a scalar output and maximize the log-likelihood as in (2.2). However, this approach gives only a sentence-level reward. To bridge the gap in the literature, we present our practical version of RTO in Algorithm 2, which features a novel calculation of token-wise reward. Our key observation is that, given a trajectory $\tau = \{(s_h, a_h)\}_{h=1}^H$, we have

$$\begin{aligned}
&\sum_{h=1}^H \beta \log \frac{\pi_\beta^*(a_h \mid s_h)}{\pi_{\mathrm{ref}}(a_h \mid s_h)} \\
=&\sum_{h=1}^H \big(Q_\beta^*(s_h, a_h) - V_\beta^*(s_h) - \log \pi_{\mathrm{ref}}(a_h \mid s_h)\big) \\
=&\sum_{h=1}^H r(s_h, a_h) - V_\beta^*(s_1) \\
&+ \underbrace{\sum_{h=1}^{H-1} \big(\mathbb{E}_{s' \sim \mathcal{P}(\cdot \mid s_h, a_h)}[V_\beta^*(s')] - V_\beta^*(s_{h+1})\big)}_{(\star)},
\end{aligned}$$

where the first equality uses $\pi_\beta^*(a \mid s) = \exp\{(Q_\beta^*(s,a) - V_\beta^*(s))/\beta\}$ in (3.4), and the second equality follows from the fact that $Q_\beta^\pi(s,a) = r_\beta(s,a) + \mathbb{E}_{s' \sim \mathcal{P}(\cdot \mid s,a)}[V_\beta^\pi(s')]$ in (3.3) with $r_\beta(s,a) = r(s,a) + \beta \log \pi_{\mathrm{ref}}(a \mid s)$. We focus on the typical LLM generation scenario where the transition kernel is deterministic. Then we have $(\star) = 0$ in (4.4), yielding that

$$\sum_{h=1}^H r(s_h, a_h) = \sum_{h=1}^H \beta \log \frac{\pi_\beta^*(a_h \mid s_h)}{\pi_{\mathrm{ref}}(a_h \mid s_h)} + V_\beta^*(s_1).$$

Building upon this result and combining it with the definition of the BT model in (3.1), for any trajectory pair $\{\tau^j = \{(s_h^j, a_h^j)\}_{h=1}^H\}_{j=1}^2$ satisfying $s_1^1 = s_1^2$, we have

$$\begin{aligned}
\mathbb{P}(\tau^1 \succ \tau^2) =& \sigma\bigg(\sum_{h=1}^H r(s_h^1, a_h^1) - \sum_{h=1}^H r(s_h^2, a_h^2)\bigg) \\
=& \sigma\bigg(\sum_{h=1}^H \beta \log \frac{\pi_\beta^*(a_h^1 \mid s_h^1)}{\pi_{\mathrm{ref}}(a_h^1 \mid s_h^1)} - \sum_{h=1}^H \beta \log \frac{\pi_\beta^*(a_h^2 \mid s_h^2)}{\pi_{\mathrm{ref}}(a_h^2 \mid s_h^2)}\bigg).
\end{aligned} \tag{4.4}$$

An interesting observation is that, based on the autoregressive nature of policies, (4.4) aligns with the learning objective of DPO proposed by Rafailov et al. (2023), but under the token-level MDP instead of the sentence-level bandit setup. Similar to the bandit setting where the learning objective is equivalent to a BT model with sentence-wise reward $r^*(x,y) = \beta \log \frac{\pi_\beta^*(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}$ (Rafailov et al., 2023), (4.4) shows that the learning objective in token-wise MDP equivalents to a BT model with a token-wise reward function

$$\begin{aligned}
&r^*(s_h = (x, y_{1:h-1}), a_h = y_h) \\
=&\beta \log \frac{\pi_\beta^*(a_h \mid s_h)}{\pi_{\mathrm{ref}}(a_h \mid s_h)} = \beta \log \frac{\pi_\beta^*(y_h \mid x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h \mid x, y_{1:h-1})},
\end{aligned} \tag{4.5}$$

where x is the prompt, $y_{1:h-1}$ is the tokens generated so far, and y_h is the token chosen at the current step. In contrast to the previous PPO implementation with sparse reward in (2.3), we will assign the token-wise reward function defined in (4.5) to each step. Formally, for any h, we define

$$\begin{aligned}
&\beta_1 \log \frac{\pi_\beta^*(y_h \mid x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h \mid x, y_{1:h-1})} - \beta_2 \log \frac{\pi(y_h \mid x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h \mid x, y_{1:h-1})} \\
\approx\ &\beta_1 \log \frac{\pi_{\mathrm{dpo}}(y_h \mid x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h \mid x, y_{1:h-1})} - \beta_2 \log \frac{\pi(y_h \mid x, y_{1:h-1})}{\pi_{\mathrm{ref}}(y_h \mid x, y_{1:h-1})} \\
:=\ &r_{\mathrm{rto}}((x, y_{1:h-1}), y_h)
\end{aligned} \tag{4.6}$$

as the token-wise reward used by RTO, where β_1 and β_2 are tuning parameters, and π is the current policy to be updated. In the last step of (4.6), we use π_{dpo}, the policy learned by DPO, as a proxy for the unknown π_β^*. Finally, we employ PPO to optimize the token-wise reward r_{rto} in (4.6).

Algorithm 2 Reinforced Token Optimization (Practical Version)

Input: Dataset $\mathcal{D}$, parameters $\beta_1, \beta_2 > 0$, DPO algorithm `DPO`, and PPO trainer `PPO-Update`.
Compute $\pi_{\text{dpo}} \leftarrow \texttt{DPO}(\mathcal{D})$ and let $\pi_0 = \pi_{\text{ref}}$ as the reference model.
for $t = 1, \ldots, T$ **do**
  Get a batch of samples $\mathcal{D}_t$ from the dataset $\mathcal{D}$ but we only keep the prompts.
  For each prompt $x \in \mathcal{D}_t$, generate a response $y \sim \pi_{t-1}(\cdot \mid x)$.
  Calculate the token-wise reward r_{rto} for each pair (x, y) by ▷ token-wise reward learning

$$r_{\text{rto}}((x, y_{1:h-1}), y_h) = \beta_1 \log \frac{\pi_{\text{dpo}}(y_h \mid x, y_{1:h-1})}{\pi_{\text{ref}}(y_h \mid x, y_{1:h-1})} - \beta_2 \log \frac{\pi_{t-1}(y_h \mid x, y_{1:h-1})}{\pi_{\text{ref}}(y_h \mid x, y_{1:h-1})}.$$

  $\pi_t \leftarrow \texttt{PPO-Update}(\pi_{t-1}, r_{\text{rto}}, \{(x, y)\}_{x \in \mathcal{D}_t})$. ▷ optimizing token-wise reward
end for
Output: policy π_T.

5. Experiments

In this section, we conduct real-world alignment experiments to verify the effectiveness of `RTO`. We provide the experimental setups and experimental results in Sections 5.1 and 5.2, respectively.

5.1. Experimental Setups

▷ *Tasks and Data.* We consider two text generation tasks, including the **Dialogue Generation** (Bai et al., 2022) and **Summarization** task (Völske et al., 2017). Given a text sequence representing dialogue history between the user and the assistant, the goal of the **Dialogue Generation** task is to generate a helpful response as the answer. For this purpose, we utilize the helpful subset of the Anthropic Helpful and Harmless (HH-RLHF) dialogue dataset[3] (Bai et al., 2022). For the **Summarization** task, we employ the Reddit TL;DR summarization dataset[4] (Völske et al., 2017), and the model is required to generate a concise summary for a given post from the Reddit forum. To facilitate readers, we provide examples of the HH-RLHF and TL;DR datasets in Appendix G.1.

▷ *Model and Baselines.* We employ the open-sourced Pythia-2.8B model (Biderman et al., 2023) as the backbone for all experiments. We select four methods as baselines. The first baseline, **SFT** (i.e., π_{ref}), fine-tunes the language model using the human-preferred responses only. Building upon this SFT model, we further train a **DPO** model, which fine-tunes the SFT model using the positive/negative preference data. Besides these two RL-free algorithms, we compare two RLHF algorithms relying on RL training. The first one is the standard **PPO** algorithm, which directly optimizes sentence-level reward in (2.3) from the SFT model. For an ablation study, we also consider a PPO variant as a baseline where the sentence-level reward is provided by the DPO objective, defined as follows:

$$r_{\text{dppo}}(x, y_{1:h}) = \begin{cases} 0 - \beta_2 \log \frac{\pi(y_h \mid x, y_{1:h-1})}{\pi_{\text{ref}}(y_h \mid x, y_{1:h-1})} \\ \text{if } h \le H-1, \\ \beta_1 \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)} - \beta_2 \log \frac{\pi(y_h \mid x, y_{1:h-1})}{\pi_{\text{ref}}(y_h \mid x, y_{1:h-1})} \\ \text{if } h = H, \end{cases} \tag{5.1}$$

where $(x, y = y_{1:H})$ is the prompt-response pair, π is the current policy, and (β_1, β_2) are tuning hyperparameters. In other words, we use the DPO to extract a sentence-level reward from the preference data, assign it to the last token, and fine-tune the model using the PPO algorithm. We refer to this baseline as **DPPO**. For our proposed **RTO**, we use the DPO model to derive a token-wise reward model (4.6), and train the policy to align human preference using PPO, as detailed in Algorithm 2. The training configurations of all aforementioned models are given in Appendix G.2.

▷ *Evaluation.* We primarily assess the alignment performance of various methods using GPT-4. The GPT-4 evaluation harnesses the capabilities of GPT-4 itself and has been shown to align well with human evaluations (Rafailov et al., 2023). For the same prompt, we provide GPT-4 with two responses generated by two different models and ask it to determine which one is superior. We then calculate the win rates, following (Rafailov et al., 2023). The prompts for GPT-4 evaluation of both HH-RLHF and TL;DR are presented in Appendix G.3. For each GPT-4 evaluation, we use 100 samples. In addition to GPT-4 evaluation, we also employ oracle reward evaluation for HH-RLHF. For this, we employ an open-sourced reward model [5] as the oracle, which is trained from Mistral-7B (Jiang et al., 2023) and achieves one of the highest accuracy on the Anthropic Helpful and Harmless dialogue task (Bai et al., 2022). Given 400 prompts, we generate the responses using all baselines and calculate the average rewards of the responses by the oracle reward model.

[3] https://huggingface.co/datasets/Anthropic/hh-rlhf
[4] https://huggingface.co/datasets/openai/summarize_from_feedback
[5] https://huggingface.co/weqweasdas/RM-Mistral-7B

Win Rate	RTO	DPO	SFT	PPO	DPPO
RTO	0.50	**0.51**	**0.56**	**0.52**	**0.56**
DPO	**0.49**	0.50	0.52	0.54	0.51
SFT	**0.44**	0.48	0.50	0.41	0.46
PPO	**0.48**	0.46	0.59	0.50	0.52
DPPO	**0.44**	0.49	0.54	0.48	0.50

Table 1(a): HH-RLHF win rates evaluated by GPT-4.

Win Rate	RTO	DPO	SFT	PPO	DPPO
RTO	0.50	**0.61**	**0.67**	**0.67**	**0.67**
DPO	**0.39**	0.50	0.58	0.59	0.50
SFT	**0.33**	0.42	0.50	0.59	0.49
PPO	**0.33**	0.41	0.41	0.50	0.40
DPPO	**0.33**	0.50	0.51	0.60	0.50

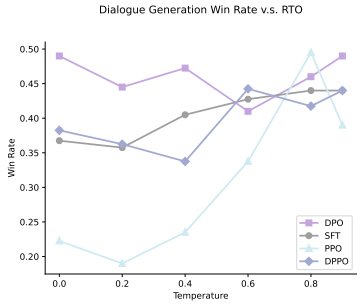
Table 1(b): TL;DR win rates evaluated by GPT-4

Table 1. Win rates between each pair of models evaluated by GPT-4. The value in line i column j represents the win rate of the model in row i against the model in column j.

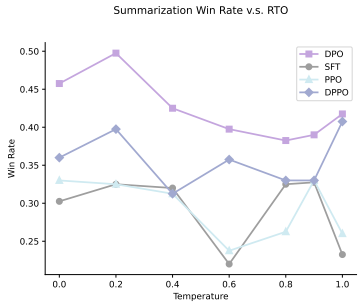
5.2. Experimental Results

The experiment results of our proposed method and baselines on HH-RLHF are detailed in Table 1(a) and Figure 3(c). This table meticulously presents the win rates between different models, assessed through both the oracle model and the GPT-4. From these results, we can see that the model trained by `RTO` achieves win rates over 50% against all other baselines, especially compared to DPO, evaluated by both the oracle reward model and GPT-4. This highlights the effectiveness of the `RTO` algorithm in alignment tasks. Furthermore, the model trained by `RTO` gets a win rate of 56% evaluated by GPT-4 over the DPPO algorithm. This implies that the token-wise reward mechanism significantly improves the performance of the RL algorithm in training models. Table 1(b) presents the performance of our method on the TL;DR dataset. We can see that the model trained by `RTO` outperforms all other baselines. Specifically, we achieve a win rate of 61% over the DPO algorithm evaluated by GPT-4. This illustrates the effectiveness of the `RTO` algorithm in a real-world text summarization task. The optimizing process curves of `RTO` and DPPO are given in Appendix F. All these empirical findings demonstrate the token-wise reward mechanism's advantage in improving model performance.

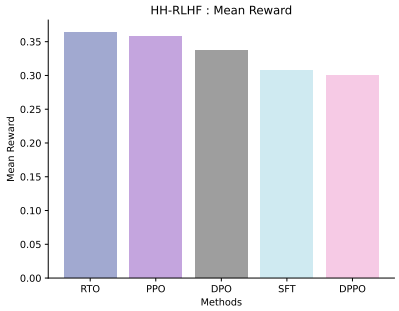
▷ *Ablation Studies on Temperatures.* We compare the model trained by `RTO` to other baselines on both datasets across different temperatures. The win rates of these comparisons, as the temperature varies, are presented in Figures 3(a) and 3(b). We can observe that the `RTO` model consistently outperforms the other baselines, highlighting its robustness across different temperatures.



(a) HH-RLHF Evaluated by GPT-4



(b) TL;DR Evaluated by GPT-4



(c) HH-RLHF Evaluated by oracle model

Figure 3. Subfigures (a) and (b) show the win rates of RTO across different sampling temperatures, while Subfigure (c) visualizes the mean reward of different baselines evaluated by the oracle model.

6. Conclusion

In this work, we suggest that the suboptimal performance of open-source implementations of PPO may be attributed to their reliance on sentence-level rewards, which neglect valuable token-wise information. To tackle this problem caused by the limitations of the previous bandit framework for RLHF, we propose an MDP formulation for RLHF that better characterizes token-wise information, along with theoretical insights demonstrating its superiority. Building upon this formulation, we introduce a novel algorithm called Reinforced Token Optimization (`RTO`), which leverages token-wise rewards to improve the policy. `RTO` is shown to be both provably sample-efficient and practical. Our practical implementation involves a novel token-wise reward learning approach via DPO, followed by optimization using PPO. This innovative combination of DPO and PPO allows `RTO` to effectively utilize token-level information and significantly improve the performance of baselines. Furthermore, our research opens up several intriguing future research directions, such as designing alternative methods for learning token-wise rewards beyond DPO and exploring other effective algorithms for optimizing token-level rewards besides PPO.

References

Agarwal, A., Kakade, S. M., Lee, J. D. and Mahajan, G. (2021). On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *The Journal of Machine Learning Research*, **22** 4431–4506.

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

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O. and Zaremba, W. (2017). Hindsight experience replay. *Advances in neural information processing systems*, **30**.

Anthropic (2023). Introducing claude. https://www.anthropic.com/index/introducing-claude

Azar, M. G., Rowland, M., Piot, B., Guo, D., Calandriello, D., Valko, M. and Munos, R. (2023). A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*.

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

Bengs, V., Busa-Fekete, R., El Mesaoudi-Paul, A. and Hüllermeier, E. (2021). Preference-based online learning with dueling bandits: A survey. *The Journal of Machine Learning Research*, **22** 278–385.

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E. et al. (2023). Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*. PMLR.

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

Cai, Q., Yang, Z., Jin, C. and Wang, Z. (2020). Provably efficient exploration in policy optimization. In *International Conference on Machine Learning*. PMLR.

Casper, S., Davies, X., Shi, C., Gilbert, T. K., Scheurer, J., Rando, J., Freedman, R., Korbak, T., Lindner, D., Freire, P. et al. (2023). Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*.

Cen, S., Cheng, C., Chen, Y., Wei, Y. and Chi, Y. (2022). Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, **70** 2563–2578.

Chan, A. J., Sun, H., Holt, S. and van der Schaar, M. (2024). Dense reward for free in reinforcement learning from human feedback. *arXiv preprint arXiv:2402.00782*.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A. and Mordatch, I. (2021). Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, **34** 15084–15097.

Chen, X., Zhong, H., Yang, Z., Wang, Z. and Wang, L. (2022). Human-in-the-loop: Provably efficient preference-based reinforcement learning with general function approximation. In *International Conference on Machine Learning*. PMLR.

Choshen, L., Fox, L., Aizenbud, Z. and Abend, O. (2019). On the weaknesses of reinforcement learning for neural machine translation. *arXiv preprint arXiv:1907.01752*.

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

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., SHUM, K. and Zhang, T. (2023). RAFT: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*.

Engstrom, L., Ilyas, A., Santurkar, S., Tsipras, D., Janoos, F., Rudolph, L. and Madry, A. (2020). Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*.

Faury, L., Abeille, M., Calauzènes, C. and Fercoq, O. (2020). Improved optimistic algorithms for logistic bandits. In *International Conference on Machine Learning*. PMLR.

Gulcehre, C., Paine, T. L., Srinivasan, S., Konyushkova, K., Weerts, L., Sharma, A., Siddhant, A., Ahern, A., Wang, M., Gu, C. et al. (2023). Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*.

Hoang Tran, B. H., Chris Glaze (2024). Snorkel-mistral-pairrm-dpo. https://huggingface.co/snorkelai/Snorkel-Mistral-PairRM-DPO

Hu, J., Tao, L., Yang, J. and Zhou, C. (2023). Aligning language models with offline reinforcement learning from human feedback. *arXiv preprint arXiv:2308.12050*.

Huang, J., Yardim, B. and He, N. (2024). On the statistical efficiency of mean-field reinforcement learning with general function approximation. In *International Conference on Artificial Intelligence and Statistics*. PMLR.

Jang, J., Kim, S., Lin, B. Y., Wang, Y., Hessel, J., Zettlemoyer, L., Hajishirzi, H., Choi, Y. and Ammanabrolu, P. (2023). Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*.

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

Jin, Y., Yang, Z. and Wang, Z. (2021). Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*. PMLR.

Kakade, S. and Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*.

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

Li, Z., Xu, T., Zhang, Y., Yu, Y., Sun, R. and Luo, Z.-Q. (2023a). Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv e-prints* arXiv–2310.

Li, Z., Yang, Z. and Wang, M. (2023b). Reinforcement learning with human feedback: Learning dynamic choices via pessimism. *arXiv preprint arXiv:2305.18438*.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I. and Cobbe, K. (2023). Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Liu, Q., Chung, A., Szepesvári, C. and Jin, C. (2022). When is partially observable reinforcement learning not scary? In *Conference on Learning Theory*. PMLR.

Liu, Z., Lu, M., Xiong, W., Zhong, H., Hu, H., Zhang, S., Zheng, S., Yang, Z. and Wang, Z. (2023). Maximize to explore: One objective function fusing estimation, planning, and exploration. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W. et al. (2021). Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

OpenAI (2023). Gpt-4 technical report. *ArXiv*, **abs/2303.08774**.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A. et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, **35** 27730–27744.

Pacchiano, A., Saha, A. and Lee, J. (2021). Dueling rl: reinforcement learning with trajectory preferences. *arXiv preprint arXiv:2111.04850*.

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

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D. and Finn, C. (2023). Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*.

Rashidinejad, P., Zhu, B., Ma, C., Jiao, J. and Russell, S. (2021). Bridging offline reinforcement learning and imitation learning: A tale of pessimism. *Advances in Neural Information Processing Systems*, **34** 11702–11716.

Rosset, C., Cheng, C.-A., Mitra, A., Santacroce, M., Awadallah, A. and Xie, T. (2024). Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*.

Saha, A. (2021). Optimal algorithms for stochastic contextual preference bandits. *Advances in Neural Information Processing Systems*, **34** 30050–30062.

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

Tang, Y., Guo, Z. D., Zheng, Z., Calandriello, D., Munos, R., Rowland, M., Richemond, P. H., Valko, M., Pires, B. Á. and Piot, B. (2024). Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A. et al. (2023). Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

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

Uehara, M. and Sun, W. (2021). Pessimistic model-based offline reinforcement learning under partial coverage. *arXiv preprint arXiv:2107.06226*.

Uesato, J., Kushman, N., Kumar, R., Song, F., Siegel, N., Wang, L., Creswell, A., Irving, G. and Higgins, I. (2022). Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, **30**.

Völske, M., Potthast, M., Syed, S. and Stein, B. (2017). Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*.

Wang, H., Lin, Y., Xiong, W., Yang, R., Diao, S., Qiu, S., Zhao, H. and Zhang, T. (2024). Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. *arXiv preprint arXiv:2402.18571*.

Wang, Y., Liu, Q. and Jin, C. (2023). Is rlhf more difficult than standard rl? *arXiv preprint arXiv:2306.14111*.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, **8** 229–256.

Williams, R. J. and Peng, J. (1991). Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, **3** 241–268.

Wu, T., Yang, Y., Zhong, H., Wang, L., Du, S. and Jiao, J. (2022). Nearly optimal policy optimization with stable at any time guarantee. In *International Conference on Machine Learning*. PMLR.

Wu, Z., Hu, Y., Shi, W., Dziri, N., Suhr, A., Ammanabrolu, P., Smith, N. A., Ostendorf, M. and Hajishirzi, H. (2024). Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, **36**.

Xiong, W., Dong, H., Ye, C., Wang, Z., Zhong, H., Ji, H., Jiang, N. and Zhang, T. (2023). Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Xiong, W., Zhong, H., Shi, C., Shen, C., Wang, L. and Zhang, T. (2022). Nearly minimax optimal offline reinforcement learning with linear function approximation: Single-agent mdp and markov game. *arXiv preprint arXiv:2205.15512*.

Yang, R., Pan, X., Luo, F., Qiu, S., Zhong, H., Yu, D. and Chen, J. (2024a). Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. *arXiv preprint arXiv:2402.10207*.

Yang, S., Zhang, S., Xia, C., Feng, Y., Xiong, C. and Zhou, M. (2024b). Preference-grounded token-level guidance for language model fine-tuning. *Advances in Neural Information Processing Systems*, **36**.

Ye, C., Xiong, W., Zhang, Y., Jiang, N. and Zhang, T. (2024). A theoretical analysis of nash learning from human feedback under general kl-regularized preference. *arXiv preprint arXiv:2402.07314*.

Yue, Y., Broder, J., Kleinberg, R. and Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, **78** 1538–1556.

Zhan, W., Uehara, M., Kallus, N., Lee, J. D. and Sun, W. (2023a). Provable offline reinforcement learning with human feedback. *arXiv preprint arXiv:2305.14816*.

Zhan, W., Uehara, M., Sun, W. and Lee, J. D. (2023b). How to query human feedback efficiently in rl? *arXiv preprint arXiv:2305.18505*.

Zhang, T. (2023). *Mathematical analysis of machine learning algorithms*. Cambridge University Press.

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

Zhong, H., Xiong, W., Zheng, S., Wang, L., Wang, Z., Yang, Z. and Zhang, T. (2022). Gec: A unified framework for interactive decision making in mdp, pomdp, and beyond. *arXiv preprint arXiv:2211.01962*.

Zhong, H. and Zhang, T. (2024). A theoretical analysis of optimistic proximal policy optimization in linear markov decision processes. *Advances in Neural Information Processing Systems*, **36**.

Zhu, B., Jiao, J. and Jordan, M. I. (2023). Principled reinforcement learning with human feedback from pairwise or k-wise comparisons. *arXiv preprint arXiv:2301.11270*.

Ziebart, B. D. (2010). *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University.

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

A. Related Works

We review the works that are mostly related to our project in this subsection. Due to the space constraint, we refer interested readers to the survey (Casper et al., 2023) for a more comprehensive overview of RLHF.

RLHF algorithm. The classic RLHF framework is established in Christiano et al. (2017); Ziegler et al. (2019) and further developed in Ouyang et al. (2022); Bai et al. (2022), where the latter can be viewed as the results of the preliminary versions of Chat-GPT and Claude. PPO (Schulman et al., 2017) is the default choice for all these projects and its effectiveness has been showcased in the resulting revolutionary foundation language models. However, as we mentioned in the introduction, tuning the PPO algorithm to its best performance requires extensive efforts and resources are often unavailable to the open-source community. Motivated by this, researchers have made efforts to develop alternative approaches to the PPO algorithm. As a direct extension of the best-of-n inference (Nakano et al., 2021), rejection sampling fine-tuning is proposed by Dong et al. (2023); Gulcehre et al. (2023); Wang et al. (2024), which prompts the LLM to generate n responses per prompt and uses a learned reward function to rank the responses and fine-tune the model on those with high rewards. Besides, inspired by the reward-conditioned training in RL literature (Chen et al., 2021), Hu et al. (2023); Yang et al. (2024a) develop conditional SFT to avoid the reward learning. Another line of work aims to skip the reward modeling step and may be referred to as the direct preference learning approach (Zhao et al., 2023; Rafailov et al., 2023; Azar et al., 2023; Tang et al., 2024). Among them, the direct preference optimization (DPO) algorithm is the most popular one, mostly due to its innovative idea: *your language model is secretly a reward model*. In particular, according to the reward benchmark (Lambert et al., 2024), the DPO-aligned algorithm often admits a competing ranking accuracy as a reward function. We will formally discuss the principle of DPO in Appendix E.1, which also partly motivates our methods. After these, there are also many tasks that consider the variants of this direct preference learning approach by increasing the training steps (Xiong et al., 2023; Hoang Tran, 2024) and consider the more general preference signal sources (Ye et al., 2024; Rosset et al., 2024). Although all these recently proposed algorithms achieve promising results, there is little evidence that these algorithms alone without PPO can make state-of-the-art LLMs. Therefore, understanding PPO and improving its performance in the context of foundation model alignment is still an important research direction.

Theoretical study of RLHF. The theoretical study of RLHF may date back to the dueling bandit and dueling RL (e.g., Yue et al., 2012; Saha, 2021; Faury et al., 2020; Bengs et al., 2021; Pacchiano et al., 2021; Chen et al., 2022; Zhu et al., 2023; Wang et al., 2023; Zhan et al., 2023a;b), where the reward maximization problem is considered in the face of preference signals, instead of the absolute reward signals. However, the reward maximization framework admits a greedy and deterministic optimal policy, which deviates from the principle of generative AI. Meanwhile, instead of the original reward function, the most widely used learning target is a Kullback-Leibler (KL)-regularized one. In recognition of the above issues, Xiong et al. (2023) first formally formulates the RLHF as the reverse-KL constrained contextual bandit in offline, online, and hybrid settings, and proposes sample-efficient algorithms in different settings accordingly. Beyond the reward-based framework under the Bradley-Terry model, Azar et al. (2023); Ye et al. (2024) consider the RLHF under a general preference oracle, and motivate the algorithmic design in a KL-regularized minimax game between two LLMs. In particular, Azar et al. (2023) proposes the first sample-efficient planning algorithm, and Ye et al. (2024) designs the sample-efficient learning algorithms in offline and online settings. Notably, as these studies of the KL-regularized framework align with the practical applications closely, the theoretical insights naturally motivate practically powerful algorithms like GSHF (Xiong et al., 2023), Nash-MD (Azar et al., 2023), and DNO (Rosset et al., 2024). However, we remark that Xiong et al. (2023); Azar et al. (2023); Ye et al. (2024) are still confined to the bandit setting, thus differing from the MDP formulation presented in this paper.

Improving PPO in the context of RLHF. Although some works (e.g., Uesato et al., 2022; Lightman et al., 2023; Yang et al., 2024b) use token-wise or step-wise information to enhance the performance of LLMs, such as their reasoning ability, we will not discuss them in detail here. Instead, we will focus on comparing our work with others that aim to improve the PPO in RLHF. In particular, Li et al. (2023a) and Ahmadian et al. (2024) state that the PPO is not the best fit for RLHF because of the sentence-level reward and deterministic transition, and argue that the reinforce-style (Williams, 1992) algorithms perform better. Wu et al. (2024) proposes to construct several separate reward functions for different goals and use the linear combination of them to guide the PPO training, but the separate models are still confined to the sentence level. Similarly, Jang et al. (2023) extends the PPO to the multi-objective optimization scenario, but still uses the sentence-level modeling. Chan et al. (2024) shares similar insights that aim to improve PPO via a dense reward. They still follow the two-staged RLHF framework to model the reward function via MLE of the Bradley-Terry model and assume that the learned

reward is based on the transformer (Vaswani et al., 2017). Then, they propose to use the attention value to redistribute the final scalar reward on a token level. In comparison, while sharing similar insights about using a token-wise reward, our techniques to obtain the dense signal and mathematical motivation are fundamentally different.

Concurrent work. During the preparation of this work, there is a concurrent and independent work (Rafailov et al., 2024) that also provides a token-wise MDP formulation for RLHF. Their work shares the same insight as ours, namely that "DPO implicitly optimizes the token-wise reward". Based on this insight, they improve the efficiency of search-based algorithms. In contrast, we propose a new algorithm `RTO` that leverages the token-wise reward functions to enhance the performance of PPO. In addition, our work provides a theoretical foundation for the unique advantages of token-wise MDP and its sample-efficient learning.

B. Proof of Theorem 3.2

Proof of Theorem 3.2. If only the sentence-level reward r_s is available, the learner must try every possible response and determine the optimal one by ranking the collected sentence-level reward signals, resulting in a sample complexity of A^H. Instead, we consider a binary tree with depth $H+1$, where each node is indexed by some token sequence $y_{1:h}$ and has A children $\{(y_{1:h}, y_{h+1})\}_{y_{h+1}\in\mathcal{A}}$. All A^H leaf nodes denote a unique prompt-response pair $(x, y_{1:H})$. We define two disjoint node sets:

$$\mathcal{N} = \big\{y_{1:h} : \pi^*(y_{1:h} \,|\, x) < A^{-\xi}, \pi^*(y_{1:h-1} \,|\, x) \geq A^{-\xi}\big\}, \qquad \mathcal{N}^*\big\{y_{1:H} : \pi^*(y_{1:H} \,|\, x) \geq A^{-\xi}\big\}. \tag{B.1}$$

Our key observations are that (i) each path must contain a node in $\mathcal{N}$ or $\mathcal{N}^*$, (ii) the path containing the node in $\mathcal{N}$ is suboptimal; and (iii) $|\mathcal{N}\cup\mathcal{N}^*| \leq A^{\xi+1}$. The exploration strategy is to query a new path that does not contain the nodes in $\mathcal{N}\cup\mathcal{N}'$ that have been visited. Since each query of a new path (response with length H) can identify a new additional node in $\mathcal{N}\cup\mathcal{N}^*$, after at least $A^{\xi+1}$ queries, we collect a set of paths where each node in $\mathcal{N}\cup\mathcal{N}^*$ belongs to one of the paths. Finally, ranking all gathered rewards of the node in $\mathcal{N}^*$ identifies the optimal $y^* = y^*_{1:H}$. Together with the fact that there exists as most A^H nodes, we finish the proof of Theorem 3.2. To facilitate understanding, we visualize a simplified learning process in Figure 2. □

C. Proof of Theorem 4.2

Recall that the visitation measure of policy π is

$$d^\pi(s) = \mathbb{E}_{s_1\sim\rho}\Big[\sum_{h=1}^{\infty}\mathbb{P}(s_t = s \,|\, s_1)\Big], \qquad d^\pi(s,a) = \mathbb{E}_{s_1\sim\rho}\Big[\sum_{h=1}^{\infty}\mathbb{P}(s_h = s, a_h = a \,|\, s_1)\Big]. \tag{C.1}$$

Under this notation, we can rewrite the value function in (3.2) as

$$V^\pi_\beta(\rho) = \mathbb{E}_{(s,a)\sim d^\pi}\big[r(s,a) - \mathrm{KL}\big(\pi(\cdot\,|\,s)\|\pi_{\mathrm{ref}}(\cdot\,|\,s)\big)\big].$$

For simplicity, we will use the shorthand $d^* = d^{\pi^*_\beta}$.

Proof of Theorem 4.2. Our proof relies on the following standard MLE analysis.

Lemma C.1 (MLE Analysis)**.** It holds with probability $1-\delta$ that

$$\|\theta_{\mathrm{MLE}} - \theta^*\|_{\Sigma_{\mathcal{D}}} \leq \varrho := C\cdot\sqrt{\frac{d\log(1/\delta)}{\Upsilon} + \lambda B^2}, \tag{C.2}$$

where C is an absolute constant and $\Upsilon = 1/(2+\exp(-2HLB)+\exp(2HLB))$.

Proof. See e.g., Faury et al. (2020); Pacchiano et al. (2021); Zhu et al. (2023) for a detailed proof. □

Back to the proof of Theorem 4.2, we first decompose the suboptimality gap defined in (3.5) as

$$\begin{aligned}\mathrm{SubOpt}(\widehat{\pi}) &= V_\beta^*(\rho; r) - V_\beta^{\widehat{\pi}}(\rho; r) \\ &= \mathbb{E}_{(s,a)\sim d^*}\big[r(s,a) - \beta \cdot \mathrm{KL}\big(\pi_\beta^*(\cdot \mid s) \| \pi_{\mathrm{ref}}(\cdot \mid s)\big)\big] \\ &\quad - \big(\mathbb{E}_{(s,a)\sim d^{\widehat{\pi}}}\big[r(s,a) - \beta \cdot \mathrm{KL}\big(\widehat{\pi}(\cdot \mid s) \| \pi_{\mathrm{ref}}(\cdot \mid s)\big)\big]\big) \\ &= \underbrace{\mathbb{E}_{(s,a)\sim d^*}[r(s,a) - \widehat{r}(s,a)]}_{\mathrm{Term(i)}} + \underbrace{\mathbb{E}_{(s,a)\sim d^{\widehat{\pi}}}[\widehat{r}(s,a) - r(s,a)]}_{\mathrm{Term(ii)}} + \underbrace{V_\beta^{\pi_\beta^*}(\rho; \widehat{r}) - V_\beta^{\widehat{\pi}}(\rho; \widehat{r})}_{\mathrm{Term(iii)}}. \end{aligned} \tag{C.3}$$

Then we analyze these three terms respectively.

Term (i). Recall that the pessimistic reward $\widehat{r}$ defined in (4.2) takes the form

$$\widehat{r}(s,a) = \phi(s,a)^\top \theta_{\mathrm{MLE}} - \varrho \cdot \|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}.$$

Then we can rewrite Term (i) in (C.3) as

$$\begin{aligned}\mathrm{Term(i)} &= \mathbb{E}_{(s,a)\sim d^*}\big[\phi(s,a)^\top(\theta^* - \theta_{\mathrm{MLE}}) + \varrho \cdot \|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big] \\ &\le \mathbb{E}_{(s,a)\sim d^*}\big[\|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}} \cdot \|\theta^* - \theta_{\mathrm{MLE}}\|_{\Sigma_{\mathcal{D}}} + \varrho \cdot \|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big] \\ &\le 2\varrho \cdot \mathbb{E}_{(s,a)\sim d^*}\big[\|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big], \end{aligned} \tag{C.4}$$

where the first inequality is obtained by Cauchy-Schwarz inequality, and the last inequality follows from Lemma C.1.

Term (ii). Similar to the derivation of (C.4), we have

$$\begin{aligned}\mathrm{Term(ii)} &= \mathbb{E}_{(s,a)\sim d^{\widehat{\pi}}}\big[\phi(s,a)^\top(\theta_{\mathrm{MLE}} - \theta^*) - \varrho \cdot \|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big] \\ &\le \mathbb{E}_{(s,a)\sim d^{\widehat{\pi}}}\big[\|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}} \cdot \|\theta_{\mathrm{MLE}} - \theta^*\|_{\Sigma_{\mathcal{D}}} - \varrho \cdot \|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big] \\ &\le 0, \end{aligned} \tag{C.5}$$

where the first inequality uses Cauchy-Schwarz inequality, and the last inequality is implied by Lemma C.1.

Term (iii). To handle this term, we introduce the following performance difference lemma for MDP with KL constraint.

Lemma C.2 (Performance Different Lemma)**.** For any reward function r and policy pair (π, π'), it holds that

$$V_\beta^\pi(\rho; r) - V_\beta^{\pi'}(\rho; r) = \mathbb{E}_{(s,a)\sim d^\pi}[Q_\beta^{\pi'}(s,a;r) - V_\beta^{\pi'}(s;r) - \beta \log \pi(a \mid s)].$$

Proof. See Appendix C.1 for a detailed proof. □

When $\beta = 0$, the regularized MDP becomes the standard MDP, and Lemma C.2 reduces to the standard performance difference lemma (Kakade and Langford, 2002). Applying Lemma C.2 to Term (iii) in (C.3), we have

$$\begin{aligned}\mathrm{Term(iii)} &= \mathbb{E}_{(s,a)\sim d^*}[Q_\beta^{\widehat{\pi}}(s,a;\widehat{r}) - V_\beta^{\widehat{\pi}}(s;\widehat{r}) - \beta \log \pi_\beta^*(a \mid s)] \\ &= \mathbb{E}_{(s,a)\sim d^*}[\beta \log \widehat{\pi}(a \mid s) - \beta \log \pi_\beta^*(a \mid s)] \\ &= -\beta \cdot \mathbb{E}_{s\sim d^*}\big[\mathrm{KL}\big(\pi_\beta^*(\cdot \mid s) \| \widehat{\pi}(\cdot \mid s)\big)\big], \end{aligned} \tag{C.6}$$

where the second equality follows from the fact that $\widehat{\pi}$ is the optimal policy with respect to $V_\beta^\pi(s;\widehat{r})$ and the expression of optimal policy $\widehat{\pi}(a \mid s) = \exp\{(Q_\beta^{\widehat{\pi}}(s,a;\widehat{r}) - V_\beta^{\widehat{\pi}}(s;\widehat{r}))/\beta\}$ in (3.4), and the last equality is obtained by the definition of KL divergence.

Finishing the Proof. Plugging (C.4), (C.5), and (C.6) into (C.3), we obtain that

$$\mathrm{SubOpt}(\widehat{\pi}) \leq 2\varrho \cdot \mathbb{E}_{(s,a)\sim d^*}\big[\|\phi(s,a)\|_{\Sigma_{\mathcal{D}}^{-1}}\big] - \beta \cdot \mathbb{E}_{s\sim d^*}\big[\mathrm{KL}\big(\pi_\beta^*(\cdot\,|\,s)\|\widehat{\pi}(\cdot\,|\,s)\big)\big],$$

which finishes the proof of Theorem 4.2. □

Remark C.3. If we do not have access to the exact optimal policy $\widehat{\pi}$ with respect to $\widehat{r}$, we can use the policy optimization algorithms to find a near-optimal optimal policy $\widetilde{\pi}$. In such case, Term (iii) in (C.5) becomes $V_\beta^{\pi_\beta^*}(\rho;\widehat{r}) - V_\beta^{\widetilde{\pi}}(\rho;\widehat{r}) = V_\beta^{\pi_\beta^*}(\rho;\widehat{r}) - V_\beta^{\widehat{\pi}}(\rho;\widehat{r}) + V_\beta^{\widehat{\pi}}(\rho;\widehat{r}) - V_\beta^{\widetilde{\pi}}(\rho;\widehat{r})$, and we need to handle the additional error term $V_\beta^{\widehat{\pi}}(\rho;\widehat{r}) - V_\beta^{\widetilde{\pi}}(\rho;\widehat{r})$. This type of error analysis has been established for NPG (Agarwal et al., 2021; Cen et al., 2022) and PPO (Cai et al., 2020; Wu et al., 2022; Zhong and Zhang, 2024).

C.1. Proof of Lemma C.2

Proof of Lemma C.2. Without loss of generality, we assume that the initial state is a fixed state $s_1 \in \mathcal{S}$. For simplicity, we also omit the dependency of r in the regularized Q-function and value function. First, we have

$$\begin{aligned} V_\beta^\pi(s_1) - V_\beta^{\pi'}(s_1) = &\underbrace{V_\beta^\pi(s_1) - \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}\big[r_\beta(s_1,a_1) + \mathbb{E}_{s_2\sim\mathcal{P}(\cdot\,|\,s_1,a_1)}[V_\beta^{\pi'}(s_2)]\big]}_{(\star)} \\ &+ \underbrace{\mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}[Q_\beta^{\pi'}(s_1,a_1)] - V_\beta^{\pi'}(s_1)}_{(\star\star)}, \end{aligned} \tag{C.7}$$

where we uses the equality $Q_\beta^{\pi'}(s_1,a_1) = r_\beta(s_1,a_1) + \mathbb{E}_{s_2\sim\mathcal{P}(\cdot\,|\,s_1,a_1)}[V_\beta^{\pi'}(s_2)]$ in (3.3) with $r_\beta(s,a) = r(s,a) + \beta\log\pi_{\mathrm{ref}}(a\,|\,s)$. By (3.3), we further have

$$\begin{aligned} V_\beta^\pi(s_1) &= \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}[-\beta\log\pi(a_1\,|\,s_1) + Q_\beta^\pi(s_1,a_1)] \\ &= \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}\big[-\beta\log\pi(a_1\,|\,s_1) + r_\beta(s_1,a_1) + \mathbb{E}_{s_2\sim\mathcal{P}(\cdot\,|\,s_1,a_1)}[V_\beta^\pi(s_2)]\big]. \end{aligned}$$

Plugging this into Term $(\star)$ of (C.7), we have

$$\begin{aligned} (\star) &= \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}\big[-\beta\log\pi(a_1\,|\,s_1) \\ &\qquad + \mathbb{E}_{s_2\sim\mathcal{P}(\cdot\,|\,s_1,a_1)}[V_\beta^\pi(s_2)]\big] - \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}\big[\mathbb{E}_{s_2\sim\mathcal{P}(\cdot\,|\,s_1,a_1)}[V_\beta^{\pi'}(s_2)]\big] \\ &= \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}[-\beta\log\pi(a_1\,|\,s_1)] + \mathbb{E}_{s_2\sim d_2^\pi}[V_\beta^\pi(s_2) - V_\beta^{\pi'}(s_2)], \end{aligned} \tag{C.8}$$

where we use $d_h^\pi(s)$ to denote the visitation measure at the $h-$th step. Meanwhile, we rewrite $(\star\star)$ in (C.7) as

$$(\star\star) = \mathbb{E}_{a_1\sim\pi(\cdot\,|\,s_1)}[Q_\beta^{\pi'}(s_1,a_1) - V_\beta^{\pi'}(s_1)]. \tag{C.9}$$

Plugging (C.8) and (C.9) into (C.7), we have

$$\begin{aligned} V_\beta^\pi(s_1) - V_\beta^{\pi'}(s_1) &= \mathbb{E}_{s_2\sim d_2^\pi}[V_\beta^\pi(s_2) - V_\beta^{\pi'}(s_2)] + \mathbb{E}_{(s_1,a_1)\sim d_1^\pi}[Q_\beta^{\pi'}(s_1,a_1) - V_\beta^{\pi'}(s_1) - \beta\log\pi(a_1\,|\,s_1)] \\ &= \cdots \\ &= \sum_{h=1}^\infty \mathbb{E}_{(s_h,a_h)\sim d_h^\pi}[Q_\beta^{\pi'}(s_h,a_h) - V_\beta^{\pi'}(s_h) - \beta\log\pi(a_h\,|\,s_h)] \\ &= \mathbb{E}_{(s,a)\sim d^\pi}[Q_\beta^{\pi'}(s,a) - V_\beta^{\pi'}(s) - \beta\log\pi(a\,|\,s)], \end{aligned}$$

where we use $\mathbb{E}_{(s_h,a_h)\sim d_h^\pi}$ to denote $\mathbb{E}_{s_h\sim d_h^\pi, a_h\sim\pi(\cdot\,|\,s_h)}$ and the definition of d^π in (C.1). Therefore, we conclude the proof of Lemma C.2. □

D. Variants of Reinforced Token Optimization

Different from Algorithm 1 where the learner constructs a pessimistic reward estimation and then outputs its corresponding optimal policy. Indeed, we can also perform pessimistic planning with respect to the value function to find the near-optimal policy:

$$\widehat{\pi} = \operatorname*{argmax}_{\pi} \min_{\theta\in\Theta} \left\{ (\mathbb{E}_{(s,a)\sim d^{\pi}}[\phi(s,a)])^{\top}\theta - \beta \cdot \mathbb{E}_{s\sim d^{\pi}}\left[\mathrm{KL}\big(\pi(\cdot\,|\,s)\|\pi_{\mathrm{ref}}(\cdot\,|\,s)\big)\right] \right\}, \tag{D.1}$$

where $\Theta = \{\|\theta\|_2 \le B : \|\theta - \theta_{\mathrm{MLE}}\|_{\Sigma_{\mathcal{D}}} \le \varrho\}$ and θ_{MLE} is given in (4.1). Here ϱ is the problem-dependent constant in (C.2) and $\Sigma_{\mathcal{D}} = \sum_{(\tau^1,\tau^2)\in\mathcal{D}}[\sum_{h=1}^{H}(\phi(s_h^1,a_h^1)-\phi(s_h^2,a_h^2))(\sum_{h=1}^{H}(\phi(s_h^1,a_h^1)-\phi(s_h^2,a_h^2)))^{\top}] + \lambda I_d$ is the covariance matrix. For policy $\widehat{\pi}$ in (D.1), we have the following theoretical guarantee.

Theorem D.1. Suppose Assumption 4.1 holds. For $\beta > 0$, $\lambda > 0$, $\delta \in (0,1)$, if we choose $\varrho = \widetilde{\mathcal{O}}(\sqrt{d})$ (see (C.2)), then the output policy $\widehat{\pi}$ of (D.1) satisfies

$$\mathrm{SubOpt}(\widehat{\pi}) \le 2\varrho \cdot \|\mathbb{E}_{(s,a)\sim d^{*}}[\phi(s,a)]\|_{\Sigma_{\mathcal{D}}^{-1}}.$$

Proof of Theorem D.1. For ease of presentation, we define

$$\widehat{V}_{\beta}^{\pi}(\rho) = \min_{\theta\in\Theta} \left\{ (\mathbb{E}_{(s,a)\sim d^{\pi}}[\phi(s,a)])^{\top}\theta - \beta \cdot \mathbb{E}_{s\sim d^{\pi}}\left[\mathrm{KL}\big(\pi(\cdot\,|\,s)\|\pi_{\mathrm{ref}}(\cdot\,|\,s)\big)\right] \right\}.$$

By Lemma C.1, we know that $\theta^* \in \Theta$ with probability $1-\delta$. This implies that

$$\widehat{V}_{\beta}^{\widehat{\pi}}(\rho) \le (\mathbb{E}_{(s,a)\sim d^{\widehat{\pi}}}[\phi(s,a)])^{\top}\theta^* - \beta \cdot \mathbb{E}_{s\sim d^{\widehat{\pi}}}\left[\mathrm{KL}\big(\widehat{\pi}(\cdot\,|\,s)\|\pi_{\mathrm{ref}}(\cdot\,|\,s)\big)\right] = V_{\beta}^{\widehat{\pi}}(\rho). \tag{D.2}$$

Meanwhile, by (D.1), we have

$$\widehat{V}_{\beta}^{\pi_{\beta}^{*}}(\rho) \le \widehat{V}_{\beta}^{\widehat{\pi}}(\rho). \tag{D.3}$$

Combining (D.2) and (D.3), we obtain

$$\widehat{V}_{\beta}^{\pi_{\beta}^{*}}(\rho) \le V_{\beta}^{\widehat{\pi}}(\rho).$$

Plugging this into the definition of the suboptimality gap in (3.5), we have

$$\mathrm{SubOpt}(\widehat{\pi}) = V_{\beta}^{*}(\rho) - V_{\beta}^{\widehat{\pi}}(\rho) \le V_{\beta}^{*}(\rho) - \widehat{V}_{\beta}^{\pi_{\beta}^{*}}(\rho)$$

Now we introduce the notation of $\widehat{\theta}$:

$$\widehat{\theta} = \operatorname*{argmin}_{\theta\in\Theta} \left\{ (\mathbb{E}_{(s,a)\sim d^{*}}[\phi(s,a)])^{\top}\theta - \beta \cdot \mathbb{E}_{s\sim d^{*}}\left[\mathrm{KL}\big(\pi_{\beta}^{*}(\cdot\,|\,s)\|\pi_{\mathrm{ref}}(\cdot\,|\,s)\big)\right] \right\}.$$

Under this notation, we further obtain that

$$\begin{aligned}
\mathrm{SubOpt}(\widehat{\pi}) &\le \mathbb{E}_{(s,a)\sim d^{*}}[(\theta^* - \widehat{\theta})^{\top}\phi(s,a)] \\
&= \mathbb{E}_{(s,a)\sim d^{*}}[(\theta^* - \theta_{\mathrm{MLE}})^{\top}\phi(s,a)] + \mathbb{E}_{(s,a)\sim d^{*}}[(\theta_{\mathrm{MLE}} - \widehat{\theta})^{\top}\phi(s,a)] \\
&\le \big(\|\theta_{\mathrm{MLE}} - \theta^*\|_{\Sigma_{\mathcal{D}}} + \|\theta_{\mathrm{MLE}} - \widehat{\theta}\|_{\Sigma_{\mathcal{D}}}\big) \cdot \|\mathbb{E}_{(s,a)\sim d^{*}}[\phi(s,a)]\|_{\Sigma_{\mathcal{D}}^{-1}} \\
&\le 2\varrho \cdot \|\mathbb{E}_{(s,a)\sim d^{*}}[\phi(s,a)]\|_{\Sigma_{\mathcal{D}}^{-1}},
\end{aligned}$$

where the second inequality uses Cauchy-Schwarz inequality, and the last inequality is obtained by Lemma C.1. Therefore, we conclude the proof of Theorem D.1. □

Remark D.2 (Extension to Unknown Transitions)**.** In (D.1), we assume that the transition kernel is known so that we can compute the state distribution d^{π} induced by the policy π. Although this is natural in LLMs, we briefly sketch the extension to the unknown transition setting. Following Zhan et al. (2023a), which is inspired by previous works on standard

reward-based RL theory (Uehara and Sun, 2021; Liu et al., 2022; Zhong et al., 2022; Liu et al., 2023; Huang et al., 2024), we can also construct a confidence set for the transition kernel

$$\Theta_{\mathcal{P}} = \left\{ P : \sum_{(\tau^1,\tau^2)\in\mathcal{D}} \sum_{i=1}^{2} \log P(\tau^i) \geq \max_{\widetilde{P}} \sum_{(\tau^1,\tau^2)\in\mathcal{D}} \sum_{i=1}^{2} \log \widetilde{P}(\tau^i) - \zeta \right\},$$

where $P(\tau)$ is the probability of observing the trajectory τ under the transition P and ζ is a tuning parameter. With a proper choice of ζ, one can also show that $\mathcal{P} \in \Theta_{\mathcal{P}}$ with high probability. Then we can perform the following pessimistic planning

$$\widehat{\pi} = \operatorname*{argmax}_{\pi} \min_{\theta\in\Theta, P\in\Theta_{\mathcal{P}}} \left\{ (\mathbb{E}_{(s,a)\sim d_P^{\pi}}[\phi(s,a)])^{\top}\theta - \beta \cdot \mathbb{E}_{s\sim d_P^{\pi}}\left[\mathrm{KL}\big(\pi(\cdot\,|\,s)\|\pi_{\mathrm{ref}}(\cdot\,|\,s)\big)\right] \right\},$$

where d_P^{π} denotes the state distribution induced by policy π under the environment P. Combining the analysis of Theorem D.1 and previous work on offline RL (Uehara and Sun, 2021; Zhan et al., 2023a), we can also establish a similar result to Theorem D.1, but with an additional estimation error for the transition kernel part. As this part is standard and not the focus of our work, we omit it for simplicity.

E. Additional Discussions

E.1. Direct Preference Optimization

Direct Preference Optimization (DPO) is a representative algorithm of the direct preference learning algorithm (Rafailov et al., 2023; Zhao et al., 2023; Azar et al., 2023; Tang et al., 2024). From a high level, these type of algorithms aim to skip the reward modeling and learn directly from the preference data, hence the name direct preference learning. In this section, we introduce the mathematical principle of DPO for completeness.

We first recall that in the original two-staged learning paradigm, we aim to optimize the following KL-regularized target:

$$\widehat{\pi} = \operatorname*{argmax}_{\pi} \mathbb{E}_{x\sim\rho, y\sim\pi(\cdot|x)}\left[r_{\mathrm{MLE}}(x,y) - \beta \log \frac{\pi(y\,|\,x)}{\pi_{\mathrm{ref}}(y\,|\,x)} \right], \tag{E.1}$$

where r_{MLE} is the MLE of the BT model on the offline preference dataset $\mathcal{D}$ obtained via

$$r_{\mathrm{MLE}} = \operatorname*{argmax}_{r} \sum_{(x,y^w,y^l)\in\mathcal{D}} \log \sigma\big(r(x,y^w) - r(x,y^l)\big). \tag{E.2}$$

One notable feature of this KL-constrained optimization problem is that it admits a closed-form solution, as summarized in the following lemma.

Lemma E.1 (Solution of KL-regularized Optimization (Proposition 7.16 and Theorem 15.3 of Zhang (2023))). Given a loss functional with respect to $\pi(\cdot\,|\,x)$, written as

$$\mathbb{E}_{y\sim\pi(\cdot\,|\,x)}\left[-r(x,y) - \beta \log \frac{\pi_{\mathrm{ref}}(y\,|\,x)}{\pi(y\,|\,x)} \right] = \beta \cdot \mathrm{KL}\Big(\pi(y\,|\,x)\Big\|\pi_{\mathrm{ref}}(y\,|\,x)\exp\Big(\frac{1}{\beta}r(x,y)\Big)\Big),$$

the minimizer of the loss functional is $\pi_r(y\,|\,x) \propto \pi_{\mathrm{ref}}(y\,|\,x)\exp\Big(\frac{1}{\beta}r(x,y)\Big)$, also known as Gibbs distribution.

Therefore, for any fixed reward function r, it leads to a closed-form policy:

$$\pi_r(y\,|\,x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y\,|\,x)\exp\Big(\frac{1}{\beta}r(x,y)\Big),$$

where $Z(x) = \sum_{y'} \pi_{\mathrm{ref}}(y'\,|\,x)\exp(\frac{1}{\beta}r(x,y'))$ is the normalization constant. Then, we can solve the reward as

$$r(x,y) = \beta \log \frac{\pi_r(y\,|\,x)}{\pi_{\mathrm{ref}}(y\,|\,x)} + \beta \log Z(x). \tag{E.3}$$

We can plug (E.3) into (E.2) to get

$$\widehat{\pi} = \operatorname*{argmax}_{\pi_r} \sum_{(x,y^w,y^l)\in\mathcal{D}} \log \sigma\left(\beta \log \frac{\pi_r(y^w\,|\,x)}{\pi_{\mathrm{ref}}(y^w\,|\,x)} - \beta \log \frac{\pi_r(y^l\,|\,x)}{\pi_{\mathrm{ref}}(y^l\,|\,x)} \right). \tag{E.4}$$

Clearly, if r is the solution of (E.2), the π_r is the solution of (E.4). On the other hand, if π is optimal for the DPO target in (E.4), then, the induced implicit reward $\beta \log \frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}$ is optimal for (E.2).

Interestingly, while the DPO is derived from the sentence-level reward function and BT model, the implicit reward naturally gives a token-wise characterization of the prompt-response pair and can be leveraged as a dense reward signal for the PPO training.

E.2. Autoregressive Policy

For the policy defined in a contextual dueling bandit setting, it maps from a prompt to a complete sentence. For ease of presentation, we call this type of policy the *predetermined policy* since it determines the entire sentence regardless of the generation process. In contrast, the Markov policy defined in the MDP formulation generates responses autoregressively: it considers not only the prompt but also the tokens generated so far. By definition, the Markov policy is at least as good as the policy that determines the whole sentence based solely on the prompt. In deterministic MDPs, the optimal action sequence is predetermined given the initial state, which demonstrates the equivalence of these two types of policies. However, for stochastic MDPs, the Markov policy is strictly more expressive than the predetermined policy. The transition can be stochastic for various reasons. For example, if the LLM uses an external search engine, the next state s_{h+1} depends not only on the current tokens $(x, y_{1:h})$ but also on the text generated by the external search engine $\pi'(\cdot \mid x, y_{1:h})$, making it stochastic. Moreover, RLHF may have applications in other scenarios, such as robotics (Christiano et al., 2017), where the transition kernel is stochastic. To clarify, we distinguish these two types of policies in the following proposition.

Proposition E.2. There exists an MDP such that the value of any predetermined policy is at least 0.5 less than that of optimal Markov/autoregressive policy.

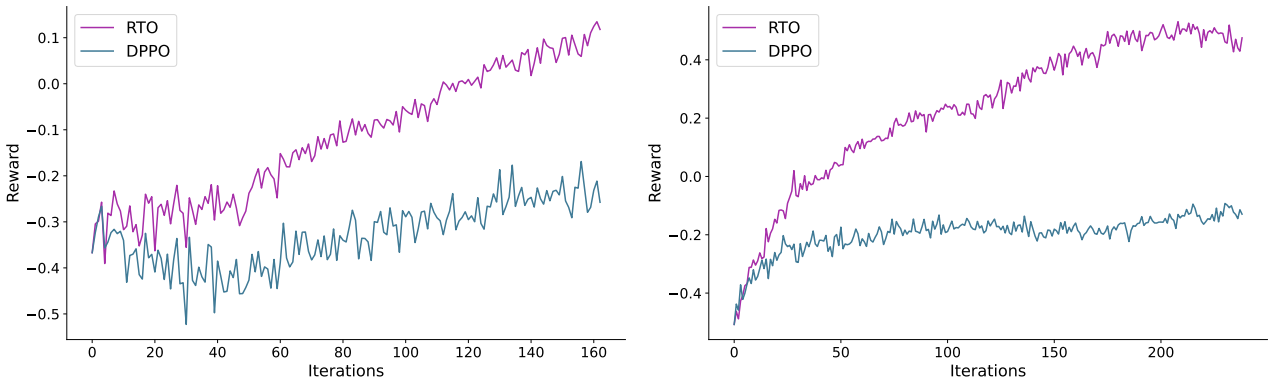
Proof. We construct an MDP $\mathcal{M}$ with state space $\mathcal{S} = \{s_0, s_1, s_2\}$, action space $\mathcal{A} = \{a_1, a_2\}$, horizon $H = 2$, fixed initial state s_0. The reward r and transition kernel $\mathcal{P}$ are given by

$$r(s_i, a_j) = \mathbb{1}\{i = j\}, \qquad \mathcal{P}(s_1 \mid s_0, a_j) = \mathcal{P}(s_2 \mid s_0, a_j) = 0.5, \qquad \forall (i, j) \in \{0, 1, 2\} \times \{1, 2\}.$$

It is straightforward to see that the optimal autoregressive policy achieves a value of 1. In contrast, any predetermined policy only achieves a value of 0.5. This completes the proof. □

F. Additional Experimental Results

Optimization Process Curves. To further investigate the benefits of the token-wise reward mechanism in the optimization process, we compare the estimated reward during the training period in Figure 4(a) and Figure 4(b). In both figures, the x-axis represents the training iterations. The y-axis represents the reward given by the implicit reward model derived from the DPO model (the reward model used in training) per batch. As we can see, in one epoch (roughly corresponds to 160 PPO training iterations in Figure 4(a) and 240 in Figure 4(b)), the reward of the model trained by `RTO` on HH-RLHF can achieve about 0.1, while the reward of the model trained by DPPO is roughly -0.25; similarly, the reward of the model trained by `RTO` on TL;DR can achieve about 0.4, while the reward of the model trained by DPPO is roughly -0.2. The results demonstrate that the token-wise reward mechanism significantly enhances the training process, leading to a remarkably higher reward.



(a) (b)

Figure 4. The reward curve of DPPO and RTO during training. The reward is given by the implicit reward model $\beta \log \frac{\pi_{\mathrm{dpo}}(y \mid x)}{\pi_{\mathrm{ref}}(x \mid y)}$ optimized by DPO. The x-axis represents the training steps, and the y-axis represents the reward values. Figure 4(a) shows the reward curve of HH-RLHF, and figure 4(b) shows the reward curve of TL;DR.

G. Additional Experimental Details

G.1. Examples of Datasets

G.1.1. A Sample of the HH-RLHF Dataset

Prompt:

Human: What does ugly Christmas sweater mean?

Assistant:

Chosen response: The ugly Christmas sweater is a popular meme that's become pretty big over the past few years. The concept is simple: you buy a cheap ugly Christmas sweater, you wear it on Christmas day, and you post a photo of yourself in the sweater on social media with some funny caption or message.

Rejected response: It means something that a person might wear as a Christmas sweater. And by "Christmas sweater", I assume you mean a sweater that people might wear around Christmas time, and not a regular sweater worn in the winter, and not a wool sweater.

G.1.2. A SAMPLE OF THE TL;DR DATASET

Prompt:

SUBREDDIT: r/AskReddit
TITLE: Reddit, what event drove you to cry in the bathroom at work?
POST: Yesterday, I finally became that girl who goes into the bathroom to cry while at work.

I work at a domestic violence shelter, and normally I'm pretty capable of brushing things off. I'm somewhat ashamed to say that it was not secondhand truama that led me to weep in the bathroom stall like a little girl, but my coworkers. It had been a rough day, which are pretty normal around here, but it was a tolerable level of rougness. My patience was wearing thin and I just wanted to go to the support group for advocates and take a breather.

Unfortunately, my coworker decided at that time to demand that I clean one of the recently vacated rooms. Not just clean it, but DEEP clean it. I'm not talking clean-it-like-your-parents-are-coming-home-after-a-weekend-away type clean. I mean, she wanted it hospital-grade clean. She wanted to be able to perform surgery on any surface of that room. The checklist she gave me- handwritten of course- had at least thirty tasks on it. For a dorm-sized room.

I lost it, guys. I just completely lost my shit. I told her that I would be happy to help clean that room, but she was absolutely off her rocker if she thought I was going to spend the next four hours cleaning by myself. She was incensed at my apparent refusal, and though I tried to reiterate that I would do it, but not alone, she started screaming for the lead advocate to put me in my place.

Well, the lead advocate just didn't want to deal with the situation and told me to just do it. I was absolutely frustrated, appalled, and overwhelmed. And so...I went into the bathroom and cried. Then I went and cleaned the stupid room.
TL;DR:

Chosen response: I was stressed, my lazy coworker demanded I clean every speck of dust from a room alone, I lost my shit, my supervisor sided with my coworker.

Rejected response: Coworker thinks it's okay to ask me to clean a room she thinks is a dumpster, so I cried. Then I cleaned it.

G.2. Training Configurations

We provide the training configuration of SFT, DPO, PPO, DPPO, and RTO below. All training can be done on 4 NVIDIA Tesla A40 48G GPUs. In the table of the training configuration of the standard PPO algorithm, we also present the configuration of training the reward model used in the PPO algorithm.

G.2.1. TRAINING CONFIGURATIONS OF HH-RLHF

SFT	
Optimizer	AdamW
Learning Rate	1e-5
Batch Size	32
Epochs	1

Table 2. Configurations for supervise fine-tuning.

DPO	
Optimizer	AdamW
Learning Rate	5e-6
KL Coefficient (β)	0.1
Batch Size	32
Epochs	1

Table 3. Configurations for DPO.

PPO	
Optimizer (PPO)	Adam
Optimizer (Reward Model)	AdamW
Mini Batch Size in PPO	16
Init KL Coefficient (β)	0.1
Learning Rate (PPO)	5e-6
Learning Rate (Reward Model)	5e-5
Batch Size Per PPO Iteration	256
Epochs of PPO Update Per Iteration	2
Batch Size (Reward Model)	128
Training Epochs (PPO and Reward Model)	1
Maximum Sequence Length	512

Table 4. Configurations for standard PPO. We also present the configuration of training the reward model used in the PPO algorithm in this table.

RTO and DPPO	
Optimizer	Adam
Learning Rate	5e-6
Training Epochs	1
Mini Batch Size in PPO	16
DPO KL Coefficient β_1	0.1
Init KL Coefficient β_2 (RTO)	0.03
Init KL Coefficient β_2 (DPPO)	0.05
Batch Size Per PPO Iteration	256
Maximum Sequence Length	512
Epochs of PPO Update Per Iteration	2

Table 5. Configurations for RTO and DPPO.

G.2.2. Training Configurations of TL;DR

SFT	
Optimizer	AdamW
Learning Rate	1e-5
Batch Size	32
Epochs	1

Table 6. Configurations for supervise fine-tuning.

DPO	
Optimizer	AdamW
Learning Rate	5e-6
KL Coefficient (β)	0.1
Batch Size	32
Epochs	1

Table 7. Configurations for DPO.

G.3. Evaluation Details

Evaluation via Oracle Reward Model: We employ the implicit reward model, derived from the DPO model, as the Oracle reward model. This evaluation is conducted for both dialogue generation and summarization tasks. To assess the performance across models, we use top-p sampling and set $p = 0.99$ and temperature $\tau = 0.9$ to generate completions for 400 prompts from the test set. These samples are then compared based on their rewards to calculate the win rate of one model over another.

PPO	
Optimizer (PPO)	Adam
Optimizer (Reward Model)	AdamW
Mini Batch Size in PPO	16
Init KL Coefficient (β)	0.05
Learning Rate (PPO)	3e-6
Learning Rate (Reward Model)	3e-6
Batch Size Per PPO Iteration	256
Epochs of PPO Update Per Iteration	2
Batch Size (Reward Model)	128
Training Epochs (PPO and Reward Model)	1
Maximum Sequence Length	512

Table 8. Configurations for standard PPO. We also present the configuration of training the reward model used in the PPO algorithm in this table.

RTO and DPPO	
Optimizer	Adam
Learning Rate	3e-6
Training Epochs	1
Mini Batch Size in PPO	16
DPO KL Coefficient β_1	0.1
Init KL Coefficient β_2 (RTO)	0.05
Init KL Coefficient β_2 (DPPO)	0.05
Batch Size Per PPO Iteration	256
Maximum Sequence Length	512
Epochs of PPO Update Per Iteration	2

Table 9. Configurations for RTO and DPPO.

Evaluation via GPT-4: Following the previous work (Rafailov et al., 2023), for evaluations utilizing GPT-4, completions are sampled by top-p sampling method with temperature of $\tau = 0.9$ and $p = 0.99$ for 100 prompts. To mitigate any positional bias inherent in GPT-4's responses, we ensure that the order of completions within each pair is randomized. The version of the GPT-4 we used is GPT-4-0613, and the specific prompt utilized for GPT-4 evaluation is detailed as follows.

Prompt for GPT-4 evaluation in dialogue generation task.
For the following query to a chatbot, which response is more helpful? Query: `<the user query>` Response A: `<either the test method or baseline>` Response B: `<the other response>` FIRST, provide a one-sentence comparison of the two responses and explain which you feel is more helpful. SECOND, on a new line, state only ”A” or ”B” to indicate which response is more helpful. Your response should use the format: Comparison: `<one-sentence comparison and explanation>` More helpful: `<"A" or "B">`

Table 10. Prompt for GPT-4 evaluation in dialogue generation task.

Prompt for GPT-4 evaluation in summarization task.
Which of the following summaries does a better job of summarizing the most important points in the given forum post, without including unimportant or irrelevant details? A good summary is both precise and concise. Post: `<the forum post>` Summary A: `<either the test method or baseline>` Summary B: `<the other summarization>` FIRST provide a one-sentence comparison of the two summaries, explaining which you prefer and why. SECOND, on a new line, state only ”A” or ”B” to indicate your choice. Your response should use the format: Comparison: `<one-sentence comparison and explanation>` Comparison: Preferred: `<"A" or "B">`

Table 11. Prompt for GPT-4 evaluation in summarization task.