WHEN IS RL BETTER THAN DPO IN RLHF? A REPRESENTATION AND OPTIMIZATION PERSPECTIVE

Ziniu Li*
The Chinese University of Hong Kong, Shenzhen
Shenzhen Research Institute of Big Data

Tian Xu* **& Yang Yu**†
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Polixir.ai

ABSTRACT

Aligning large language models with human preferences is important, and there are two kinds of alignment methods. The first class of algorithms is based on reinforcement learning (RL), which involves learning a reward function from a human preference dataset and improving performance via online reward maximization. Another class is characterized by direct preference optimization, exemplified by DPO (Rafailov et al., 2023), which learns an implicit reward and improves performance directly using a static offline dataset. Which algorithm performs well? We investigate this question using contextual bandits, which serve as mathematical models for alignment. We have two findings: First, we show that DPO may suffer from a reward quality issue when the feature representation is misspecified. Second, we present the error bounds for RL algorithms and show that they achieve the best improvement when the online updates are sufficient. The code to reproduce our results is available at `https://github.com/liziniu/policy_optimization`.

1 INTRODUCTION

Developing large language models (LLMs) requires alignment with human preferences. A standard practice involves providing a human preference dataset to guide LLMs. According to utility theory (Fishburn et al., 1979), preference is connected with a certain reward function. Currently, there are two kinds of alignment methods:

- The first class of methods investigated in (Christiano et al., 2017; Stiennon et al., 2020; Ouyang et al., 2022) is based on reinforcement learning (RL) (Sutton & Barto, 2018). These methods learn a separate reward function from the preference data and improve performance via online reward maximization (using algorithms like PPO (Schulman et al., 2017) or ReMax (Li et al., 2023)).
- The second class of methods is characterized by direct preference optimization, exemplified by DPO (Rafailov et al., 2023). Essentially, it parameterizes the reward function using the language model itself and thus avoids the explicit learning of the reward function.

While both approaches are able to improve performance by leveraging preference data, the superiority of one method over the other remains an open question, crucial for driving future advancements.

2 MAIN RESULTS

We investigate the above question within the framework of contextual bandits (Lattimore & Szepesvári, 2020). We study a linear bandit task and denote the prompt by s and response by a. The ground truth reward function is $r^\star(s,a) = \phi_r(s,a)^\top \theta_r^\star$, with $\phi_r(s,a) \in \mathbb{R}^d$ denoting a known feature mapping and $\theta_r^\star \in \mathbb{R}^d$ as the parameter to learn. In particular, a preference data $D_{\text{pref}} = \{(s_i, a_i, a_i')\}_{i=1}^n$ is

*Equal contribution. Author ordering is determined by coin flip. Emails: `ziniuli@link.cuhk.edu.cn` and `xut@lamda.nju.edu.cn`

†Corresponding author. Email: `yuy@nju.edu.cn`

collected by a reference policy π_{ref} with a_i is more likely preferred to a_i' if $r(s_i, a_i)$ is larger than $r(s_i, a_i')$. More details are provided in Appendix B.

DPO is inferior to RL when the representation is misspecified. The key to RLHF is the quality of the recovered reward function. The RL-based methods learn a separate reward function $r(s, a) = \phi_r(s, a)^\top \theta_r$ via maximum likelihood estimation; see Eq. (1), where σ denotes the sigmoid function.

$$\widehat{r} \leftarrow \operatorname*{argmax}_r \sum_{i=1}^{n} \log \sigma \left(r(s_i, a_i) - r(s_i, a_i') \right), \quad (s_i, a_i, a_i') \sim D_{\text{pref}}. \tag{1}$$

On the other hand, DPO reparameterizes the reward using the language model π, i.e., $r(s, a) \propto \beta \log(\pi(a|s)/\pi_{\text{ref}}(a|s))$; see Eq. (2), where β is a small positive number. Thus, the reward quality in DPO is restricted by the representation power of the language model.

$$\widehat{\pi}_{\text{DPO}} \leftarrow \operatorname*{argmax}_\pi \sum_{i=1}^{n} \log \sigma \left(\beta \log \frac{\pi(a_i|s_i)}{\pi_{\text{ref}}(a_i|s_i)} - \beta \log \frac{\pi(a_i'|s_i)}{\pi_{\text{ref}}(a_i'|s_i)} \right), \quad (s_i, a_i, a_i') \sim D_{\text{pref}}. \tag{2}$$

We argue that DPO could face a reward quality issue when the representation feature is misspecified, i.e., $\phi_\pi \neq \phi_r$. This scenario often occurs in practice when networks have different architectures or are trained on different datasets. We evaluated the reward functions from RL and DPO using preference classification accuracy on a test dataset, as shown in Figure 1. When $\phi_\pi = \phi_r$, DPO matches RL in preference classification accuracy. However, when $\phi_\pi \neq \phi_r$, DPO's accuracy significantly lags behind RL. We also report the optimality gaps of the learned policies in Figure 2, where DPO is comparable to RL only when $\phi_\pi = \phi_r$ and underperforms otherwise.

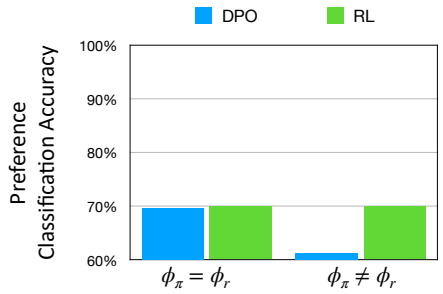

Figure 1: Preference classification accuracy (the larger, the better).

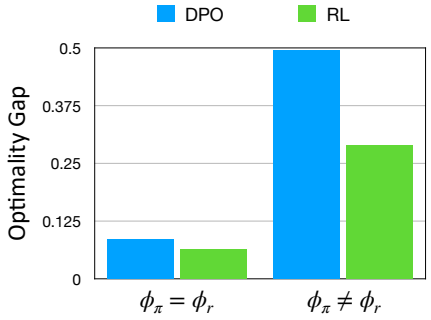

Figure 2: Alignment preference in the optimality gap (the smaller, the better).

Sufficient Online Update is Crucial for RL. Once the reward function is learned, the quality of policy optimization becomes crucial for RL-based algorithms. Unlike DPO, RL-based algorithms utilize online optimization, meaning that responses to be improved are generated from the language model itself, which changes over iterations. The goal of reward maximization, as described in Eq. (3), involves two expectations: one over the prompt distribution and the other over the response distribution. Since these two distributions are not directly accessible, we typically use Monte Carlo approximations to estimate them or their gradients.

$$\widehat{\pi}_{\text{RL}} \leftarrow \operatorname*{argmax}_\pi \mathbb{E}_{s \sim \rho(\cdot)} \left\{ \mathbb{E}_{a \sim \pi(\cdot|s)} \left[\widehat{r}(s, a) \right] - \beta D_{\text{KL}}(\pi(\cdot|s), \pi_{\text{ref}}(\cdot|s)) \right\}. \tag{3}$$

We present the error bounds for this stochastic approximation; refer to the detailed analysis in Appendix A. Focusing on the representation mismatch scenario to explore key factors, we conduct the ablation study below. We show that when prompts or responses generated from the language model are insufficient, the optimality gap becomes large, indicating that the potential of RL-based algorithms is not fully realized. Note that using a large number of prompts and responses does not require preference annotations, as the reward model can provide weak supervision signals.

	RL	RL (prompts are not sufficient)	RL (responses are not sufficient)
Optimality gap	0.2874	0.3078	0.2999

Table 1: Alginment performance in the optimality gap of RL-based algorithms.

URM STATEMENT

The authors acknowledge that at least one key author of this work meets the URM criteria of ICLR 2024 Tiny Papers Track. For example, the author Ziniu Li is outside the age range of 30-50 years.

REFERENCES

Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *The Journal of Machine Learning Research*, 22(1):4431–4506, 2021.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems 30*, pp. 4299–4307, 2017.

Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

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

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Peter C Fishburn, Peter C Fishburn, et al. *Utility theory for decision making*. Krieger NY, 1979.

Seyed Kamyar Seyed Ghasemipour, Richard S. Zemel, and Shixiang Gu. A divergence minimization perspective on imitation learning methods. In *Proceedings of the 3rd Conference on Robot Learning*, pp. 1259–1277, 2019.

Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems 29*, pp. 4565–4573, 2016.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in neural information processing systems 32*, pp. 12498–12509, 2019.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. *Advances in Neural Information Processing Systems 20*, 2007.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.

Ziniu Li, Tian Xu, Yushun Zhang, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient method for aligning large language models. *arXiv preprint arXiv:2310.10505*, 2023.

Tyler Lu, Dávid Pál, and Martin Pál. Contextual multi-armed bandits. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pp. 485–492, 2010.

R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2005.

Fan-Ming Luo, Tian Xu, Xingchen Cao, and Yang Yu. Reward-consistent dynamics models are strongly generalizable for offline reinforcement learning. *arXiv preprint arXiv:2310.05422*, 2023.

Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J. Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *Foundations and Trends in Robotic*, 7(1-2): 1–179, 2018.

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

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

Dean Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv*, 1707.06347, 2017.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pp. 1057–1063, 1999.

Tian Xu, Ziniu Li, Yang Yu, and Zhi-Quan Luo. Understanding adversarial imitation learning in small sample regime: A stage-coupled analysis. *arXiv preprint arXiv:2208.01899*, 2022.

Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, et al. Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales. *arXiv preprint arXiv:2308.01320*, 2023.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems 33*, pp. 14129–14142, 2020.

A Analysis

A.1 Problem Formulation

We consider the so-called contextual bandits (Langford & Zhang, 2007; Lu et al., 2010) formulation, which serves mathematical models for alignment. Let s and a be the state and action, respectively. We aim to obtain a decision policy π that acts optimally in terms of reward maximization:

$$\pi_r^\star \leftarrow \operatorname*{argmax}_\pi \mathbb{E}_{s\sim\rho(\cdot)}\mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a)\right], \tag{4}$$

where the symbol ρ denotes the state distribution, and r is the ground truth reward function. We omit the subscript r in $\pi_r^\star$ when the context is clear. For language models, the term "states" refers to prompts, while "actions" denote responses. The language model functions as the decision-making policy. It is worth noting that terminologies may be used interchangeably.

In the context of RLHF, it is usually to connect the human preference with reward by the Bradley-Terry-Luce model (Plackett, 1975; Luce, 2005):

$$\mathbb{P}\left(a > a' \mid s\right) = \frac{\exp(r(s,a))}{\exp(r(s,a)) + \exp(r(s,a'))}.$$

Here $a > a'$ means that the event that a is more preferred to a'. Then, by maximum likelihood estimation, we get the reward learning objective in (1).

A.2 Theoretical Analysis of RL-based Methods

In this section, we present a preliminary analysis of errors in RL-based methods. At a high level, we identify three types of errors for RL-based methods:

1) the reward evaluation error $|\widehat{r}(s,a) - r(s,a)|$;
2) the estimation error when using finite samples to calculate the expectation $\mathbb{E}_{a\sim\pi(\cdot|s)}[\cdot]$;
3) the estimation error when using finite samples to calculate the expectation $\mathbb{E}_{s\sim\rho(\cdot)}[\cdot]$.

In the following part, we will provide the theoretical analysis of the above three types of errors in RL-based methods.

We first formulate the optimization procedure of RL-based methods. For simplicity, we consider $\beta = 0$. We define $V_r(\pi) := \mathbb{E}_{s\sim\rho(\cdot),a\sim\pi(\cdot|s)}[r(s,a)]$ as the expected reward of π under the reward function r. Since the true reward function r and state distribution ρ are unknown, RL turns to maximize the following empirical objective:

$$\max_{\pi} \widehat{V}_{\widehat{r}}(\pi) := \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}[\widehat{r}(s,a)]. \tag{5}$$

Here $\widehat{r}$ is the reward function recovered by maximum likelihood estimation from the preference dataset and $\widehat{\rho}$ is the state distribution estimated from finite samples.

RL often applies stochastic gradient ascent to solve the optimization problem in (5). Here we consider the direct parameterization for policies, i.e., $\Pi = \{\pi \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} : \forall(s,a) \in \mathcal{S}\times\mathcal{A}, \pi(a|s) > 0, \sum_{a\in\mathcal{A}} \pi(a|s) = 1\}$. According to the policy gradient theorem (Sutton et al., 1999), we have that

$$\nabla \widehat{V}_{\widehat{r}}(\pi) = \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}[\nabla \log \pi(a|s)\widehat{r}(s,a)].$$

In this policy gradient formula, there is an expectation over the policy $\mathbb{E}_{a\sim\pi(\cdot|s)}[\cdot]$, which is difficult to calculate in practice. Consequently, RL-based methods often collect finite actions from the current policy to estimate this expectation, which results in the following stochastic gradient.

$$\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi) = \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\widehat{\pi}(\cdot|s)}[\nabla \log \pi(a|s)\widehat{r}(s,a)].$$

Then a projected stochastic gradient ascent step is performed repeatedly for the policy update.

$$\pi_{t+1} = \mathrm{Proj}_{\Pi}\left(\pi_t + \eta_t \widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right), \forall t = 1, 2, \cdots, T. \tag{6}$$

The following proposition presents the error bounds of RL-based methods. The proof is based on some analysis for stochastic-gradient-based methods (Bottou et al., 2018).

Proposition 1. *Consider the RL-based method in (6) and the output policy $\overline{\pi} = \sum_{t=1}^{T} \pi_t/T$. We define the reward evaluation error $\varepsilon_r := \max_{(s,a)\in\mathcal{S}\times\mathcal{A}} |r(s,a) - \widehat{r}(s,a)|$, the state distribution estimation error $\varepsilon_s := \sup_{\pi\in\Pi} |\mathbb{E}_{s\sim\rho(\cdot),a\sim\pi(\cdot|s)}[r(s,a)] - \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}[r(s,a)]|$ and the action distribution estimation error $\varepsilon_a := \sup_{\pi\in\Pi} \|\nabla\widehat{V}_{\widehat{r}}(\pi) - \widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi)\|^2$. Then we have that*

$$\max_{\pi\in\Pi} V_r(\pi) - \mathbb{E}\left[V_r(\overline{\pi})\right] \le 2\left(\varepsilon_r + \varepsilon_s\right) + \sqrt{\frac{2\left(\varepsilon_a + |\mathcal{S}||\mathcal{A}|^2 R_{\max}^2\right)}{T}},$$

where $R_{\max} = \max_{(s,a)\in\mathcal{S}\times\mathcal{A}} |\widehat{r}(s,a)|$ and the expectation is taken with respect to the randomness of $\overline{\pi}$.

Proposition 1 implies that there are three types of errors in the optimality gap bound of RL-based methods. The reward evaluation error ε_r is due to the limited preference data. The state estimation error ε_s arises from using finite state samples to estimate the state distribution ρ. We can effectively reduce ε_s by additionally collecting sufficient state samples. Finally, the policy estimation error ε_a is caused by using finite actions sampled from the policy to estimate the policy gradient. With more computation power, we can employ the policy model to sample sufficient actions and thereby estimate the policy distribution accurately.

Proof. We use $\pi^\star := \operatorname{argmax}_\pi V_r(\pi)$ to denote the optimal policy. By the standard error decomposition analysis, we have that

$$\begin{aligned}
&V_r(\pi^\star) - \mathbb{E}\left[V_r(\overline{\pi})\right] \\
&= V_r(\pi^\star) - \widehat{V}_{\widehat{r}}(\pi^\star) + \widehat{V}_{\widehat{r}}(\pi^\star) - \max_\pi \widehat{V}_{\widehat{r}}(\pi) + \max_\pi \widehat{V}_{\widehat{r}}(\pi) - \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right] + \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right] - \mathbb{E}\left[V_r(\overline{\pi})\right] \\
&\leq V_r(\pi^\star) - \widehat{V}_{\widehat{r}}(\pi^\star) + \max_\pi \widehat{V}_{\widehat{r}}(\pi) - \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right] + \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right] - \mathbb{E}\left[V_r(\overline{\pi})\right].
\end{aligned}$$

By Lemma 1, we have that

$$V_r(\pi^\star) - \mathbb{E}\left[V_r(\overline{\pi})\right] \leq V_r(\pi^\star) - \widehat{V}_{\widehat{r}}(\pi^\star) + \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right] - \mathbb{E}\left[V_r(\overline{\pi})\right] + \sqrt{\frac{2\left(\varepsilon_a + |\mathcal{S}||\mathcal{A}|^2 R_{\max}^2\right)}{T}}.$$

For any policy π, we have that

$$\begin{aligned}
\left|V_r(\pi) - \widehat{V}_{\widehat{r}}(\pi)\right| &= \left|\mathbb{E}_{s\sim\rho(\cdot),a\sim\pi(\cdot|s)}\left[r(s,a)\right] - \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}\left[\widehat{r}(s,a)\right]\right| \\
&\leq \left|\mathbb{E}_{s\sim\rho(\cdot),a\sim\pi(\cdot|s)}\left[r(s,a)\right] - \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}\left[r(s,a)\right]\right| \\
&\quad + \left|\mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}\left[r(s,a)\right] - \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\pi(\cdot|s)}\left[\widehat{r}(s,a)\right]\right| \\
&\leq \varepsilon_s + \varepsilon_r.
\end{aligned}$$

Then we obtain that

$$V_r(\pi^\star) - \mathbb{E}\left[V_r(\overline{\pi})\right] \leq 2\left(\varepsilon_s + \varepsilon_r\right) + \sqrt{\frac{2\left(\varepsilon_a + |\mathcal{S}||\mathcal{A}|^2 R_{\max}^2\right)}{T}}.$$

We complete the proof. □

The following Lemma provides the optimization error bound of applying stochastic gradient ascent to solve the optimization problem in Eq. (5).

Lemma 1. *Consider the RL-based method in (6) with the step size of* $\eta = \sqrt{2/(T(\varepsilon_a + |\mathcal{S}||\mathcal{A}|^2 R_{\max}^2))}$*, and the output policy* $\widehat{\pi} = \sum_{t=1}^T \pi_t/T$*. We define the action distribution estimation error* $\varepsilon_a := \sup_{\pi\in\Pi} \|\nabla\widehat{V}_{\widehat{r}}(\pi) - \widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi)\|^2$*. Then we have that*

$$\max_\pi \widehat{V}_{\widehat{r}}(\pi) - \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right] \leq \sqrt{\frac{2\left(\varepsilon_a + |\mathcal{S}||\mathcal{A}|^2 R_{\max}^2\right)}{T}},$$

where $R_{\max} = \max_{(s,a)\in\mathcal{S}\times\mathcal{A}} |\widehat{r}(s,a)|$ *and the expectation is taken with respect to the randomness of* $\overline{\pi}$*.*

Lemma 1 indicates that the optimization error of RL depends on the policy estimation error ε_a. With more computation power, we can employ the policy model to sample more actions and thereby estimate the policy distribution accurately, reducing the optimization error effectively.

Proof. For any policy $\pi \in \Pi$, we have that

$$\begin{aligned}
\widehat{V}_{\widehat{r}}(\pi) - \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\pi_t)\right] &= \mathbb{E}\left[\widehat{V}_{\widehat{r}}(\pi) - \widehat{V}_{\widehat{r}}(\pi_t)\right] \\
&\overset{(a)}{\leq} \mathbb{E}\left[(\pi - \pi_t)^\top \nabla\widehat{V}_{\widehat{r}}(\pi_t)\right] \\
&\overset{(b)}{=} \mathbb{E}\left[(\pi - \pi_t)^\top \mathbb{E}\left[\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)|\pi_t\right]\right] \\
&\overset{(c)}{=} \mathbb{E}\left[(\pi - \pi_t)^\top \widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right].
\end{aligned}$$

Here inequality (a) holds since $\widehat{V}_{\widehat{r}}(\pi)$ is a concave function with π, equation (b) holds due to the unbiasedness of $\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)$ and equation (c) follows the Tower property. Furthermore, we have that

$$\begin{aligned}\mathbb{E}\left[(\pi-\pi_t)^\top \widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right] &= \frac{1}{\eta}\mathbb{E}\left[(\pi-\pi_t)^\top \eta\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right] \\ &= \frac{1}{2\eta}\mathbb{E}\left[\left(\|\pi-\pi_t\|_2^2+\eta^2\left\|\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2-\left\|\pi-\pi_t-\eta\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2\right)\right] \\ &\le \frac{1}{2\eta}\mathbb{E}\left[\left(\|\pi-\pi_t\|_2^2+\eta^2\left\|\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2-\|\pi-\pi_{t+1}\|_2^2\right)\right].\end{aligned}$$

Here the last inequality follows the non-expansivity of the projection operator, see Lemma 3.1 in (Bubeck, 2015) for details. Then we continue to upper bound the second moment of the stochastic gradient.

$$\begin{aligned}\mathbb{E}\left[\left\|\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2\right] &= \mathbb{E}\left[\left\|\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)-\nabla\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2\right]+\left\|\mathbb{E}\left[\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right]\right\|_2^2 \\ &\le \varepsilon_a+\left\|\nabla\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2.\end{aligned}$$

By simple calculation, we have that $\nabla\widehat{V}_{\widehat{r}}(\pi_t)=\{\rho(s)\widehat{r}(s,a)\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}\in\mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$. Then we obtain that

$$\left\|\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right\|_2^2\le\varepsilon_a+\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}}\rho^2(s)\widehat{r}^2(s,a)\le\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2.$$

With the above bound on the second moment of the stochastic gradient, we have that

$$\begin{aligned}\widehat{V}_{\widehat{r}}(\pi)-\mathbb{E}\left[\widehat{V}_{\widehat{r}}(\pi_t)\right] &\le \mathbb{E}\left[(\pi-\pi_t)^\top\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi_t)\right] \\ &\le \frac{1}{2\eta}\mathbb{E}\left[\left(\|\pi-\pi_t\|_2^2-\|\pi-\pi_{t+1}\|_2^2\right)\right]+\frac{\eta}{2}\left(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2\right).\end{aligned}$$

Taking a summation from $t=1$ to $t=T$ yields that

$$\begin{aligned}\sum_{t=1}^T\widehat{V}_{\widehat{r}}(\pi)-\mathbb{E}\left[\widehat{V}_{\widehat{r}}(\pi_t)\right] &\le \frac{1}{2\eta}\mathbb{E}\left[\left(\|\pi-\pi_1\|_2^2-\|\pi-\pi_{T+1}\|_2^2\right)\right]+\frac{\eta T}{2}\left(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2\right) \\ &\le \frac{1}{2\eta}\|\pi-\pi_1\|_2^2+\frac{\eta T}{2}\left(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2\right) \\ &\le \frac{1}{\eta}+\frac{\eta T}{2}\left(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2\right) \\ &\overset{(a)}{=} \sqrt{2T\left(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2\right)}.\end{aligned}$$

Equation (a) is obtained by using the step size of $\eta=\sqrt{2/(T(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2))}$. Finally, we obtain at

$$\max_\pi\widehat{V}_{\widehat{r}}(\pi)-\mathbb{E}\left[\widehat{V}_{\widehat{r}}(\overline{\pi})\right]=\max_\pi\frac{1}{T}\sum_{t=1}^T\widehat{V}_{\widehat{r}}(\pi)-\mathbb{E}\left[\widehat{V}_{\widehat{r}}(\pi_t)\right]\le\sqrt{\frac{2\left(\varepsilon_a+|\mathcal{S}||\mathcal{A}|^2R_{\max}^2\right)}{T}},$$

which completes the proof. □

A.3 Reward Modeling in DPO

In this section, we elaborate on the reward modeling in DPO (Rafailov et al., 2023). In general, we may define a reward class $\mathcal{R}=\{r:\mathcal{S}\times\mathcal{A}\to\mathbb{R}\}$ with the hope that the ground truth reward function in this class. RL-based methods typically learn a separate reward function. In contrast, DPO parameterizes the reward function by

$$\mathcal{R}_{\mathrm{DPO}}=\{r:\mathcal{S}\times\mathcal{A}\to\mathbb{R}\mid r(s,a)\propto[\log\pi(s,a)-\log\pi_{\mathrm{ref}}(s,a)]\}.$$

That is, DPO uses the log probability of policy to model the reward function. Thus, the reward quality in DPO is limited to the representation power of the policy model. If the representation is misspecified, i.e., $\mathcal{R}\neq\mathcal{R}_{\mathrm{DPO}}$, the reward and associated optimal policy may be of poor quality. We

note that this issue can arise in practical scenarios where the reward model and policy model take different architectures; see Figure 3.

(a) Reward neural network.

(b) Policy neural network.

Figure 3: Architectures of the reward and policy models used in practice differ.The reward neural network learns the joint representation of state and action, while the policy model first learns the state representation. As a result, the representations for these two models are different.

B Experiment Setup

In this section, we introduce the experiment setup of this paper. All of our experiments are run with 10 different random seeds (2021-2030), and the averaged results are reported. Note that we set π_{REF} to be a policy with a uniform action distribution in all experiments and $\beta = 0.01$ for all methods. Besides, we use a policy with a uniform action distribution to collect the preference data.

We study a linear bandit task, where we have $r(s,a) = \phi_r(s,a)^\top \theta_r^\star$, with $\phi_r(s,a) \in \mathbb{R}^d$ denoting the feature representation and $\theta_r^\star \in \mathbb{R}^d$ as the parameter. In this case, the reward learning optimization problem is convex, so we use CVXPY (Diamond & Boyd, 2016) to find the solution $\widehat{r}$. In particular, we use the feature map $\phi_r(s,a)$ and the parameter $\theta_r^\star$ as

$$\phi_r(s,a) = \left((a+1)\cdot\cos(s\cdot\pi), \frac{1}{a+1}\cdot\sin(s\cdot\pi)\right)^\top, \quad \theta_r^\star = (1,2)^\top,$$

where $s \in \mathcal{S} = [0,1]$ and $a \in \mathcal{A} = \{0,1,2,3\}$. A uniform distribution over $\mathcal{S}$ is studied. For the policy, we consider the parameterization

$$\pi(a|s) = \frac{\exp(\phi_\pi(s,a)^\top \theta_\pi)}{\sum_{a'} \exp(\phi_\pi(s,a')^\top \theta_\pi)},$$

with $\phi_\pi(s,a)$ and θ_π both in $\mathbb{R}^2$. In this case, the policy optimization problem is a non-convex problem, but the gradient domination condition holds (Agarwal et al., 2021). We use the gradient ascent method with the AdaGrad optimizer (Duchi et al., 2011) (a step size of 0.1 is used).

We examine two scenarios. In the first scenario, there is no representation misspecification, i.e., $\phi_\pi = \phi_r$. In the second, we consider the representation misspecification case and use a different feature map for policy:

$$\phi_\pi(s,a) = \left((a+1)\cdot\sin(s\cdot\pi), \frac{1}{a+1}\cdot\cos(s\cdot\pi)\right)^\top.$$

In our experiments, we set the size of training preference data to be $n = 20$. In Figure 1, we use 100 test preference data to evaluate the reward functions. In Figure Figure 2, the RL method uses the states in the training preference data to estimate the state distribution $\rho(\cdot)$ in Eq. (3).

In Table 1, we evaluate three RL-based methods: RL, RL (prompts are not sufficient), RL (responses are not sufficient). In particular, RL additionally collects a preference-free data $D_{\text{pref-free}} = \{s_j\}_{j=1}^m$ with $m = 100$, and apply it to estimate the state distribution ρ. We note that, unlike the preference dataset, it is relatively cheap to collect such a large preference-free dataset. Besides, RL utilizes the exact policy distribution to construct the policy gradient of $\nabla \widehat{V}_{\widehat{r}}(\pi) = \mathbb{E}_{s\sim\widehat{\rho}(\cdot), a\sim\pi(\cdot|s)}[\nabla \log \pi(a|s)\widehat{r}(s,a)]$. We use this to simulate the case that sufficient responses are collected to approximate the policy distribution accurately.

RL (prompts are not sufficient) differs from RL in the use of state samples for estimating the state distribution. Specifically, RL (prompts are not sufficient) only uses the states in the preference dataset.

Besides, unlike RL utilizes the exact policy distribution, RL (responses are not sufficient) uses 5 responses to estimate the policy distribution, i.e., $\widehat{\nabla}\widehat{V}_{\widehat{r}}(\pi) = \mathbb{E}_{s\sim\widehat{\rho}(\cdot),a\sim\widehat{\pi}(\cdot|s)}[\nabla \log \pi(a|s)\widehat{r}(s,a)]$.

C Connection with Other Research

Our research is related to imitation learning (Osa et al., 2018), which aims to learn a policy from expert demonstrations. A popular approach to achieve this goal is through behavioral cloning (BC) (Pomerleau, 1991), which trains a policy model by maximizing the likelihood of expert data. Note that the working mechanism of BC is quite similar to DPO: the likelihood of positively preferred actions is increased and that of negatively preferred actions is decreased:

$$\pi_{\mathrm{BC}} \leftarrow \operatorname*{argmax}_{\pi} \sum_{i=1}^{n} \log \pi(a_i|s_i), \quad (s_i, a_i) \sim D_{\mathrm{E}},$$

where D_{E} is the expert dataset.

Ghasemipour et al. (2019) showed that another class of imitation methods, known as adversarial imitation learning (AIL) methods, (such as GAIL (Ho & Ermon, 2016)), usually performs better than BC. In particular, AIL methods leverage a recovered reward function to perform online policy optimization, significantly improving performance. Following the formulation in (Xu et al., 2022), the training objective of reward-model-based AIL can be re-formulated as

$$\pi_{\mathrm{AIL}} \leftarrow \operatorname*{argmin}_{\pi} \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} \left| d_{\pi}(s,a) - \widehat{d}_{\mathrm{E}}(s,a) \right|,$$

where $\widehat{d}_{\mathrm{E}}$ is the empirical state-action distribution estimated from D_{E}, and $d_{\pi}(s,a)$ is obtained from online interaction. For the optimization objective of AIL, it utilizes states beyond those in the expert dataset (reflected in the summation over all state-action pairs). We notice that Xu et al. (2022) theoretically proved that AIL can outperform BC in terms of addressing the distribution shift issue with this online optimization. The idea of recovering a reward function and using it to perform extensive policy optimization is quite similar to the RL methods in RLHF.

Additionally, our research is related to transition-model-based reinforcement learning methods, where the goal is to find an optimal policy through interactions with environments. Many empirical successes suggest that transition-model-based approaches are superior in terms of sample complexity (Luo et al., 2019; Janner et al., 2019). We do not aim to present a detailed discussion since RL involves lots of concepts and notations. Instead, we would like to highlight that our findings align with the understanding that additional policy optimization on transition-model-generated data is helpful. We would like to refer readers to (Hafner et al., 2020; Schrittwieser et al., 2020; Yu et al., 2020; Luo et al., 2023) for the effect of data augmentation in transition-model-based RL methods.

Finally, we note that compared with DPO (Rafailov et al., 2023), RL-based methods do not require extra preference annotation. For applications such as language models, training and storing a reward model has been shown to be highly efficient, as demonstrated in (Yao et al., 2023). The primary challenge in RL-based methods lies in the huge action space during policy optimization. However, this issue can be effectively addressed by computationally efficient methods like those proposed by (Dong et al., 2023; Li et al., 2023). Notably, Li et al. (2023) showed that optimizing the language model with prompts-only data can improve performance, a setting that cannot achieved by DPO.