Getting More Juice Out of the SFT Data: Reward Learning from Human Demonstration Improves SFT for LLM Alignment

Jiaxiang Li [1] Siliang Zeng [1] Hoi-To Wai [2] Chenliang Li [3] Alfredo Garcia [3] Mingyi Hong [1]

Abstract

Aligning human preference and value is important for contemporary foundation models. State-of-the-art techniques such as Reinforcement Learning from Human Feedback (RLHF) consist of two stages: 1) supervised fine-tuning (SFT), where the model is fine-tuned to imitate human demonstration data; 2) Preference learning, where preference data is used to learn a reward model, which is then used by a reinforcement learning (RL) step to fine-tune the model. In this work, we argue that the SFT stage benefits from learning a reward model as well. Instead of using the human demonstration data directly via supervised learning, we propose to leverage an Inverse RL (IRL) technique to build an reward model, while learning the policy model. This approach leads to new SFT algorithms that are not only efficient to implement, but also promote the ability to distinguish between preferred and non-preferred continuations. Our results indicate that it is beneficial to explicitly or implicitly leverage reward learning throughout the entire alignment process.

1. Introduction

Large Language Models (LLMs) have become the cornerstone of modern artificial intelligence applications. They are believed to lead the way towards artificial general intelligence (Bubeck et al., 2023), also have shown great capabilities towards specialized domains such as math problem solving (Cobbe et al., 2021; Trinh et al., 2024; Wei et al., 2022; Lewkowycz et al., 2022), code generation (Chen et al., 2021; Austin et al., 2021; Li et al., 2022), text generation (Anil et al., 2023; Touvron et al., 2023; Thoppilan et al., 2022), etc. Usually, one needs to align pre-trained LLMs with human-labeled data to achieve desired performance over certain tasks, a process known as alignment or fine-tuning. The alignment datasets can be categorized into two classes: (i) demonstration data with input prompt and human response; (ii) preference data with input prompt and two responses, where human labeler will pick a chosen one and a rejected one. With the alignment datasets, one could employ methods like supervised fine-tune (SFT, (Ouyang et al., 2022; Tunstall et al., 2023; Chung et al., 2024)) for demonstration datasets, and reinforcement learning from human feedback (RLHF, (Christiano et al., 2017; Ouyang et al., 2022)) and direct preference optimization (DPO, (Rafailov et al., 2024)) for preference datasets. Specifically, RLHF *explicitly* trains a reward model and uses RL (policy optimization) to obtain a fine-tuned LLM; on the other hand, DPO and its extensions simplifies the RLHF by training the LLM policy model directly, while *implicitly* learns the reward model via the log likelihood ratio between the learned and reference models. Both methods exhibit better performance over SFT on demonstration datasets and are adopted by state-of-the-art LLMs, for example ChatGPT by RLHF (Ouyang et al., 2022), zephyr by DPO (Tunstall et al., 2023).

When dealing with preference data, state-of-the-art methods usually build an (explicit or implicit) reward model to evaluate the quality of responses for a given prompt. On the contrary, reward modeling is not done for demonstration datasets. However, as *human preferences* are also implicit in the demonstration data, one can argue that training a reward model that encodes human value distilled from these datasets may help boost the alignment capability of the LLM. In RL literature, for a Markov decision process (MDP), it is likely that supervised learning methods which naively fit the demonstration data will suffer from distribution shift – the fine-tuned policy from supervised learning produce unsatisfactory generations in certain states unseen in the training dataset (Ross et al., 2011). Through formulating the learning from demonstration problem, inverse RL methods (Ziebart et al., 2008; Ross et al., 2011; Zeng et al., 2022b) can alleviate such distribution shift issues. Witnessing the success in ChatGPT, one would expect the LLM alignment with demonstration datasets can be improved through in-

[1]Department of Electrical and Computer Engineering, University of Minnesota [2]Department of Systems Engineering and Engineering Management, Chinese University of Hong Kong [3]Department of Industrial and Systems Engineering, Texas A&M University. Correspondence to: Jiaxiang Li <li003755@umn.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

verse RL. In this work, we pose the following question:

Does building a reward model using the demonstration data benefit the alignment process?

By developing an IRL framework, we give a positive answer to above question. In particular our main contributions are:

- We develop a new reward-based SFT approach, which takes the form of a *bilevel* optimization, where in the *lower-level*, LLM policy is learned via policy optimization for a given reward, while in the *upper-level*, the reward model is optimized so to maximize the likelihood for observing the demonstration data.
- We propose two alignment algorithms, one learns the reward model explicitly, and the other implicitly. For the first algorithm, we show that the reward learned from only demonstration data already possesses strong capabilities in distinguishing between chosen and rejected responses; see Figure 1 and our experiment for details. For the second algorithm, we observe that implicitly learning a reward is equivalent to improving the model by comparing the demonstration data with *synthetic* data generated by past models. The resulting algorithm covers SPIN (Chen et al., 2024) as a special case, where the latter algorithm has been recently proposed from a different viewpoint.
- We prove that both proposed algorithms converge to a stationary point of our proposed formulation. We show that the proposed algorithms outperform vanilla SFT in almost all cases we have tested, for example the model performance on HuggingFace Open LLM Leaderboard increases from 59.47% to 61.03%.

Notations. $\pi(y|x)$ denotes the LLM output probability, and we refer to π as the policy; we use $\pi(y|x;\boldsymbol{\theta})$ if π is directly parameterized by $\boldsymbol{\theta}$. When π is indirectly determined by $\boldsymbol{\theta}$, we use $\pi_{\boldsymbol{\theta}}(y|x)$. $\mathcal{D}=\{(x,y)\}$ denotes the demonstration dataset and $\mathcal{P}=\{(x,y_w,y_l)\}$ for the preference dataset, where y_w is preferred over y_l. We also denote $(x,y)\sim\mathcal{D}$ as $x\sim\rho, y\sim\pi^{E}(\cdot|x)$, where ρ is the input prompts distribution. We similarly denote $x\sim\rho,(y_l\prec y_w)\sim\pi^{P}(\cdot|x)$ for the preference dataset.

2. Preliminaries

Consider a Large Language Model (LLM) $\pi(y|x;\boldsymbol{\theta})$ where $x=[x_1,...,x_n]$ is the sequence of input prompts and $y=[y_1,...,y_m]$ is the sequence of continuation. We review two procedures for fine-tuning $\boldsymbol{\theta}$: (1) SFT over demonstration dataset, (2) RLHF over preference dataset.

SFT. Given a *demonstration dataset* $\mathcal{D}:=\{(x,y)\}$, the SFT optimizes the following problem:

$$\max_{\boldsymbol{\theta}}\ \ell_{\mathrm{SFT}}(\boldsymbol{\theta}):=\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\log\pi\left(y|x;\boldsymbol{\theta}\right)\right]. \quad (1)$$

It is easy to see that the above problem shares the same optimal solutions as $\min_{\boldsymbol{\theta}}\ \mathbb{E}_{x\sim\rho}[D_{\mathrm{KL}}(\pi^{\mathrm{E}}(\cdot|x)\,\|\pi(\cdot|x;\boldsymbol{\theta}))]$, which shows that SFT imitates the demonstration dataset via minimizing the KL divergence.

RLHF. Suppose that we have a reward model $r(x,y;\boldsymbol{\phi})$ (parameterized by $\boldsymbol{\phi}$) for any given input and output pair (x,y), the LLM can be fine tuned by the RL problem:

$$\begin{aligned}\max_{\boldsymbol{\theta}}\ \ell_{\mathrm{RL}}(\boldsymbol{\theta}):=\ &\mathbb{E}_{x\sim\rho,y\sim\pi(\cdot|x;\boldsymbol{\theta})}\left[r(x,y;\boldsymbol{\phi})\right]\\ &-\mathbb{E}_{x\sim\rho}[D_{\mathrm{KL}}(\pi\left(\cdot|x;\boldsymbol{\theta}\right)\|\pi_{\mathrm{ref}}\left(\cdot|x\right))],\end{aligned} \quad (2)$$

where π_{ref} is a fixed reference model, usually the pre-trained model or the model after SFT. (2) is usually solved by standard policy optimization techniques such as REINFORCE (Ahmadian et al., 2024) or PPO (Schulman et al., 2017).

To find an appropriate reward model $r(x,y;\boldsymbol{\phi})$, RLHF (see e.g., (Christiano et al., 2017)) leverages a *preference dataset* $\mathcal{P}:=\{(x,y_w,y_l)\}$, where each data contains a pair of output y_w,y_l, and y_w is preferred over y_l by human labeler (denoted as $y_w\succ y_l$). The Bradley-Terry model (Bradley and Terry, 1952) assumes that the probability of choosing y_w over y_l is (where $\sigma(\cdot)$ is the sigmoid function):

$$\mathbb{P}\left(y_w\succ y_l\mid x;\boldsymbol{\phi}\right)\ =\sigma\left(r(y_w;x;\boldsymbol{\phi})-r\left(y_l;x;\boldsymbol{\phi}\right)\right).$$

The following problem finds the reward model:

$$\max_{\boldsymbol{\phi}}\ \ell_{\mathrm{RM}}(\boldsymbol{\phi}):=\mathbb{E}_{x\sim\rho,(y_l\prec y_w)\sim\pi^{P}(\cdot|x)}\Big[\log\Big(\mathbb{P}\left(y_w\succ y_l\mid x;\boldsymbol{\phi}\right)\Big)\Big] \quad (3)$$

It is observed that models trained via learning the policy (2) and then learning the reward (3) outperforms those that are only trained using SFT (Ouyang et al., 2022). The reward model allows a better generalization ability via the consistent input of the preference data from human labeler.

3. Reward Learning and Policy Fine Tuning from Demonstration Data

In this section, we argue that reward learning from the demonstration dataset can benefit the LLM alignment by a joint reward learning and policy fine tuning formulation.

3.1. Joint Reward-learning and Policy Fine-tuning

We consider the *joint* reward and policy learning problem via maximum likelihood inverse RL (ML-IRL) (Ziebart et al., 2008; 2013; Zeng et al., 2022b;a):

$$\begin{aligned}\max_{\boldsymbol{\theta}}\ \ell(\boldsymbol{\theta}):=&\mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x)}\left[\log\pi_{\boldsymbol{\theta}}\left(y\mid x\right)\right]\\ \text{s.t. }\pi_{\boldsymbol{\theta}}:=&\arg\max_{\pi}\mathbb{E}_{x\sim\rho,y\sim\pi(\cdot|x)}[r\left(x,y;\boldsymbol{\theta}\right)\\ &-\beta D_{\mathrm{KL}}\Big(\pi(\cdot|x)\|\pi_{\mathrm{ref}}(\cdot|x)\Big)].\end{aligned} \quad (4)$$

The above problem has a *bilevel* structure. The upper level is similar to (1), but is evaluated on the policy $\pi_{\boldsymbol{\theta}}$ induced by the reward model $r\left(x,y;\boldsymbol{\theta}\right)$; meanwhile, this policy $\pi_{\boldsymbol{\theta}}$ is found in the lower level using the RL objective (2).

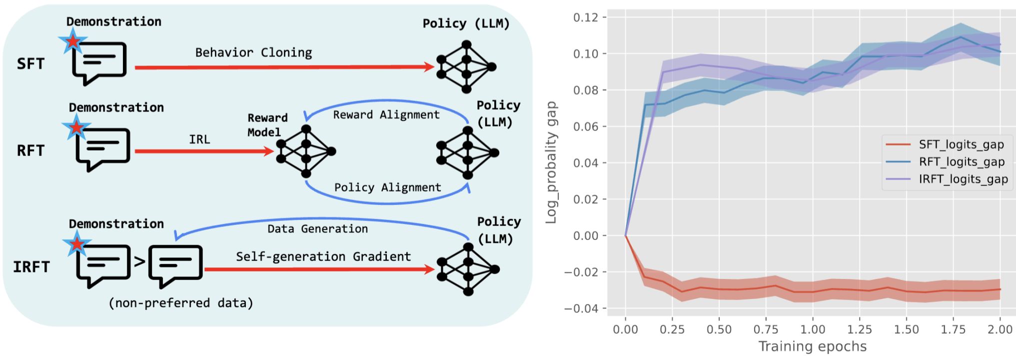


Figure 1. **Left:** Difference between SFT and the proposed methods: RFT (Algorithm 1) and IRFT (Algorithm 2); **Right:** Log probability gap between the chosen/preferred continuation and the rejected/non-preferred continuations. All methods *only* consume the chosen/preferred data, but RFT and IRFT can distinguish between chosen and rejected continuations; see Example A.2.

Several advantages of (4) over (1): First, formulating SFT as a RL / IRL problem can alleviate distribution shift and improve the generalization power (Ross et al., 2011). In fact, we observe that (4) tends to give a less extreme policy even when the demonstration dataset is. The latter is observed in the stylized example A.1 in the appendix. Second, since the lower level problem in (4) encapsulates a generation process, we anticipate the proposed method to better distinguish between the preferred and non-preferred data than SFT, even if it is only trained on the demonstration dataset. The numerical example in Fig. 1, Example A.2 highlights this point. Below we show that (4) can be simplified (proof in Appendix A.4):

Lemma 3.1. *Problem (4) is equivalent to the minimax optimization problem:*

$$\max_{\boldsymbol{\theta}} \min_{\pi} \mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi(\cdot|x)} \Big[\frac{r(x,y;\boldsymbol{\theta}) - r(x,\tilde{y};\boldsymbol{\theta})}{\beta} + D_{\mathrm{KL}}\Big(\pi(\cdot|x) \| \pi_{\mathrm{ref}}(\cdot|x)\Big)\Big]. \tag{5}$$

The reward optimization problem takes a similar form as in RLHF (3), where two reward functions are contrasted. The key difference is that here one reward is on the continuation y in $\mathcal{D}$, the other is on $\tilde{y}$ *generated* from the current policy $\pi(\cdot|x)$. We believe that such contrast is the key reason that enables the IRL formulation to distinguish the preferred continuations over the non-preferred ones.

We can develop a gradient-descent-ascent type method for minimax problem (5) — an algorithm that we call *Reward-learning Fine-tune* (RFT) in Algorithm 1 of the appendix.

3.2. Implicit Reward-learning Fine-tuning

We show that (4) can be simplified into a supervised learning problem. Observe (see Appendix A.4 for proof):

Lemma 3.2. *For the loss function ℓ in (4), we have:*

$$\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}) = \frac{1}{\beta} \mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|x)} \left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)} \right]. \tag{6}$$

Lemma 3.2 leads to a simple scheme for *implicit reward-based supervised fine-tune* (IRFT) – for each training batch, it samples the response from current model, and construct the gradient estimator (6) to update $\boldsymbol{\theta}$. This results in Algorithm 2 in the appendix.

3.3. Convergence Theory

We conclude the section by theoretically inspecting the proposed algorithms. We observe:

Theorem 3.3. *Under Assumption A.1, for Algorithm 1 and 2 with $\eta_t = \Theta(1/\sqrt{TK})$ we have*

$$\min_{t=1,\ldots,T,\ k=1,\ldots,K} \mathbb{E}[\|\nabla \ell(\boldsymbol{\theta}_{t,k})\|^2] \leq \mathcal{O}\left(1/\sqrt{TK} + 1/T\right).$$

Assume that the total number of data samples is N (which is a large number) and we perform generation for each sample once (i.e., epoch $= 1$), then we have $N = T \times K$. In this case, as long as $K \leq \sqrt{N}$, then the convergence rate is always $\mathcal{O}(1/\sqrt{N})$. If K is larger than $\sqrt{N}$, then the bias in the estimator of the gradient will degrade the performance.

4. Numerical experiments

In this section we study the proposed Algorithm 1 and 2 numerically. Details of the experiments are in Appendix A.5.

4.1. Results of RFT (Algorithm 1)

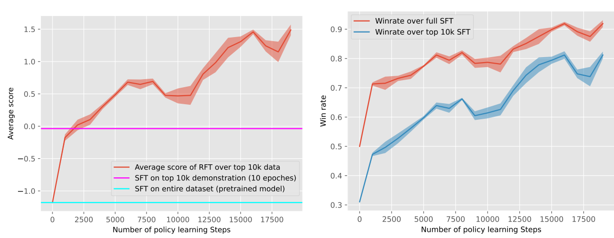
We first fine-tuned `pythia-1.4b` using supervised fine-tune over Anthropic-HH dataset. We use only preferred/chosen data for 10 epochs and pick up the best checkpoint as our base model. Next, we fine-tune the base model using SFT and Algorithm 1. Figure 2 shows the experiment results on averaged reward and win rate, where we record the average score of the continuation generated for test datasets, also the win rate of the proposed Algorithm 1 over the full SFT base model and the top 10k SFT model. The figures show that the proposed algorithm improves over SFT in terms of the helpfulness and harmlessness of model continuation. See Section A.5 for implementation details.

Table 1. Test performance of `SPIN` (Chen et al., 2024) and IRFT (Algorithm 2) based on `pythia-1.4b` across HuggingFace Open LLM Leaderboard datasets. We keep training for 2 epochs after each generation process and K are calculated after this rule.

Tasks Metrics	T	K	AI2_Arc acc_norm	TruthfulQA acc	Winogrande acc	GSM8k exact_match	HellaSwag acc_norm	MMLU acc	Average
`pythia-1.4b`	0	0	54.54	31.00	57.46	1.44	53.55	25.63	37.27
`SFT`	0	$\frac{\#\text{ samples}}{\text{batchsize}} * 2$	54.74	30.93	57.30	2.05	52.98	25.62	37.27
`IRFT` (SPIN iter 0)	1	$\frac{\#\text{ samples}}{\text{batchsize}} * 2$	54.00	31.73	57.70	1.36	53.76	25.54	37.35
`IRFT` (SPIN iter 1)	2	$\frac{\#\text{ samples}}{\text{batchsize}} * 2$	52.85	**32.04**	57.38	1.74	53.57	25.49	37.18
`IRFT`	5	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{5}$	53.75	31.67	56.91	1.74	**54.79**	25.32	37.36
`IRFT`	10	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{5}$	53.75	31.92	57.85	**2.43**	54.77	25.44	**37.69**
`IRFT`	8	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{8}$	53.75	31.40	56.91	2.35	54.62	25.52	37.43
`IRFT`	16	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{8}$	**56.34**	31.54	**58.41**	1.59	54.54	**25.69**	37.57

Table 2. Test performance of `SPIN` (Chen et al., 2024) and IRFT (Algorithm 2) based on `zephyr-7b-sft-full` across HuggingFace Open LLM Leaderboard datasets.

Tasks Metrics	T	K	AI2_Arc acc_norm	TruthfulQA acc	Winogrande acc	GSM8k exact_match	HellaSwag acc_norm	MMLU acc	Average
`zephyr-7b-sft-full`	0	0	74.83	34.07	76.09	31.92	81.09	**58.86**	59.48
`IRFT` (SPIN iter 0)	1	$\frac{\#\text{ samples}}{\text{batchsize}} * 2$	75.08	36.57	76.01	33.59	82.81	57.83	60.32
`IRFT` (SPIN iter 1)	2	$\frac{\#\text{ samples}}{\text{batchsize}} * 2$	76.13	36.56	76.64	**35.56**	**83.39**	57.82	61.02
`IRFT`	5	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{5}$	75.82	**39.99**	77.19	31.24	82.07	57.93	60.71
`IRFT`	10	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{5}$	**76.78**	36.84	**77.43**	34.34	83.05	57.72	**61.03**
`IRFT`	8	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{8}$	75.23	36.67	75.85	31.84	80.89	58.60	59.85
`IRFT`	16	$\frac{\#\text{ samples}}{\text{batchsize}} * \frac{2}{8}$	75.79	35.55	76.56	32.52	82.3	58.77	60.25



(a) Average Score (b) Win Rate

Figure 2. Fine-tuning result of `pythia-1.4b` over Anthropic-HH (with top 10k data picked by `PKU-Alignment/beaver-7b-v3.0-reward`) using Algorithm 1. We record the average score of test dataset on the left figure and the win rate of Algorithm 1 over the (full SFT) base model and the SFT model.

4.2. Results of IRFT (Algorithm 2)

Different from the time consuming Algorithm 1, Algorithm 2 is more capable of handling large data and models. We first present the result for `pythia-1.4b models` over Ultrachat200k data. Note that $T = 1$ in Algorithm 2 is equivalent to SPIN (Chen et al., 2024)[1]. We tested on different choices of T and identify that $T = 5$ to 8 gives the best performance in the Open LLM Leaderboard evaluations.

[1]IRFT with $T = 1$ and 2 epochs is equivalent to SPIN iteration 0, and $T = 2$ with 2 epochs for each T is equivalent to SPIN iteration 1, etc.

The Open LLM Leaderboard result is presented in Table 1. We have the following main observations from Table 1:

1. SFT is not efficient in terms of boosting the pre-trained model performance on downstream tasks comparing to methods which promote the decreasing of the likelihood of synthetic data, namely SPIN and IRFT;
2. SPIN and IRFT (Algorithm 2) are both capable of further improving the performance of pythia model over downstream tasks, whereas IRFT shows better results due to more frequent generation comparing to SPIN. IRFT with $T > 1$ outperforms both SFT and SPIN on most of the tasks as well as the average score;
3. More frequent generation might also result in more variances, therefore a reasonable T (around 5) results in the best evaluation performance.

Apparently 1b model is not strong enough to handle hard tasks, e.g. GSM8k and all model performances are not desirable. Now we present the result for `zephyr-7b-sft-full`. We remind the reader that this is a fully SFT-ed model and further SFT would only detriment the model performance (see Chen et al. (2024)). The results are presented in Table 2 where we can see that similar to the 1b case, both SPIN and IRFT could effectively improve the performance of SFT-ed model and the average performance of IRFT with $T = 5$ stands out. The success of IRFT and SPIN suggest that reward learning is beneficial for aligning with demonstration data.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

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

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

Michael Bloem and Nicholas Bambos. Infinite time horizon maximum causal entropy inverse reinforcement learning. In *53rd IEEE conference on decision and control*, pages 4911–4916. IEEE, 2014.

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

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv preprint arXiv:2312.09390*, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

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

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=TyFrPOKYXw.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

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

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 12 2023. URL https://zenodo.org/records/10256836.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of LLM via a human-preference dataset. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=g0QovXbFw3.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. *Advances in neural information processing systems*, 24, 2011.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

Fei Liu et al. Learning to summarize from human feedback. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, Jan Peters, et al. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 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*, 35:27730–27744, 2022.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

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

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. *arXiv preprint arXiv:2210.01241*, 2022.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

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

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

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

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

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

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*, 2024.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Siliang Zeng, Mingyi Hong, and Alfredo Garcia. Structural estimation of markov decision processes in high-dimensional state space with finite-time guarantees. *arXiv preprint arXiv:2210.01282*, 2022a.

Siliang Zeng, Chenliang Li, Alfredo Garcia, and Mingyi Hong. Maximum-likelihood inverse reinforcement learning with finite-time guarantees. *Advances in Neural Information Processing Systems*, 35:10122–10135, 2022b.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. The principle of maximum causal entropy for estimating interacting processes. *IEEE Transactions on Information Theory*, 59(4):1966–1980, 2013.

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

A. Appendix

A.1. Related works

Fine-tuning language models is prevailing to improve LLMs performance on various instructional tasks, and has shown great success in enabling LLMs to generalize to efficiently respond out-of-sample instructions (Chung et al., 2024). Despite many successful applications of SFT, people soon realized the great potential of reward learning and RL based fine-tuning over preference datasets for different tasks, including text-summarizing (Liu et al., 2020; Ziegler et al., 2019), story-telling (Ziegler et al., 2019), instruction-following (Ouyang et al., 2022; Ramamurthy et al., 2022), etc. Equipped with the popular Bradley-Terry model (Bradley and Terry, 1952), RLHF fine-tune a language model using policy optimization methods, such as REINFORCE (Williams, 1992), proximal policy optimization (PPO, (Schulman et al., 2017)) and a lot more. On major obstacle for preference dataset fine-tuning is the costly and time-consuming process of human labeling, and methods such as self-play fine-tune (SPIN, (Chen et al., 2024)), synthetic data with binary feedback in self-training (Singh et al., 2023), weak-to-strong generalization (Burns et al., 2023) and self-rewarding fine-tuning (Yuan et al., 2024) seek for improvement over SFT under weaker data supervisions comparing to preference datasets. In particular, SPIN generates synthetic samples for input prompts in the demonstration dataset and use them as the rejected data to for a 'pseudo' preference data. As we will see, SPIN actually coincides with our implicit reward learning approach where we motivate the synthetic data in a more natural way.

In the RL literature, IRL proposes to jointly learn the reward r which best explains an expert policy π^E and the policy π which in turn mimics this expert policy π^E from demonstration data. The most popular framework is the maximum entropy IRL (MaxEnt-IRL) framework (Ziebart et al., 2008; Levine et al., 2011; Ziebart et al., 2013; Bloem and Bambos, 2014; Zeng et al., 2022b), which seeks for a policy maximizing the entropic-regularized reward that matches the empirical averages in expert's demonstrations data. MaxEnt-IRL utilizes only the demonstration dataset for reward learning and already yields superior performance over the plain behavior cloning (Pomerleau, 1988; Osa et al., 2018) approach on various RL tasks.

A.2. Examples Illustrating the Benefits of Joint Reward and Policy Learning (4)

Example A.1. *Suppose we have only one state (input prompt) x and three actions (continuations) y_1, y_2, y_3. Let the reference model π_{ref} be a uniform distribution over all continuations, and the demonstration dataset is $\mathcal{D} = \{y_3\}$. One could easily compute the optimal solution for* (1) *and* (4) *by first-order optimality conditions. From Table 3 we can see that SFT (imitation learning) pushes all the likelihood toward the demonstration dataset, whereas ML-IRL* (4) *maintains non-zero weights for unseen data in the demonstration datasets. This is particular useful when we want to fine-tune from a pre-trained model, which is presumed to be powerful and have useful information already.*

Action	y_1	y_2	y_3
π_{ref}	0.33	0.33	0.33
$\mathcal{D}$		$\{y_3\}$	
π_{SFT}	0.0	0.0	1.0
π_{IRL}	$\frac{2}{2+e^{R/\beta}}$	$\frac{2}{2+e^{R/\beta}}$	$\frac{e^{R/\beta}}{2+e^{R/\beta}}$

Table 3. A state-less counter-example with three actions where IRL-based fine-tune (4) shows regularization effect over SFT (1) to maintain weights over unseen data in the demonstration dataset $\mathcal{D}$. Here we assume $r \in [0, R]$.

Example A.2. *We compare the solution of SFT* (1) *and IRL* (4) *numerically, where the latter is solved using two algorithms RFT and IRFT (to be introduced shortly). We choose the preference based dataset Anthropic-HH and only keep the preferred continuation to form a demonstration dataset $\tilde{\mathcal{D}} = \{(x, y_w)\}$ to implement SFT and IRL. We then compute the log probability gap $\log(\pi(y_w|x)) - \log(\pi(y_l|x))$ between the preferred y_w and non-preferred y_l on the test dataset; see Figure 1 right side. We observe that although all three methods are not exposed to the non-preferred data y_l during the training process, the IRL-based methods effectively distinguish the preferred continuation over the non-preferred one, while SFT assigns larger probability to the non-preferred continuation (see Section 4 for the details).*

Algorithm 1 *Reward-learning Fine-Tune* (RFT)

Input: Initialize reward parameter $\boldsymbol{\theta}_0(\boldsymbol{\theta}_{-1,K} = \boldsymbol{\theta}_0)$ and policy model π^0, the stepsize of reward update η_t, and T, K the outer and inner iterations.
for $t = 0, 1, \ldots, T-1$ **do**
  Take $\boldsymbol{\theta}_{t,0} = \boldsymbol{\theta}_t := \boldsymbol{\theta}_{t-1,K}$
  Data Sample: Sample state $x_{t,k} \sim \rho$, an expert response $y_{t,k} \sim \pi^{\mathrm{E}}(\cdot|x_{t,k})$ and agent response $\tilde{y}_{t,k} \sim \pi^t(\cdot|x_{t,k})$, for $k = 0, 1, ..., K-1$
  for $k = 0, 1, ..., K-1$ **do**
    Estimate Gradient: Calculate the stochastic gradient $g_{t,k}$ w.r.t. $\boldsymbol{\theta}$ via $g_{t,k} = \frac{1}{\beta}\nabla_{\boldsymbol{\theta}} r(x_{t,k}, y_{t,k}; \boldsymbol{\theta}_{t,k}) - \frac{1}{\beta}\nabla_{\boldsymbol{\theta}} r(x_{t,k}, \tilde{y}_{t,k}; \boldsymbol{\theta}_{t,k})$
    Reward Alignment: $\boldsymbol{\theta}_{t,k+1} := \boldsymbol{\theta}_{t,k} + \eta_t g_{t,k}$
  end for
  Policy Alignment: Update the optimal $\pi^t(y|x) \propto \exp(r(x, y; \boldsymbol{\theta}_{t,K}))$ according to (8)
end for

Algorithm 2 *Implicit Reward-learning Fine-Tune* (IRFT)

Input: Initialize model parameter $\boldsymbol{\theta}_0(\boldsymbol{\theta}_{-1,K} = \boldsymbol{\theta}_0)$, the stepsize of reward update η_t, and T, K the outer and inner iterations.
Output: $\hat{\boldsymbol{\theta}}$
for $t = 0, 1, ..., T-1$ **do**
  Take $\boldsymbol{\theta}_{t,0} = \boldsymbol{\theta}_t := \boldsymbol{\theta}_{t-1,K}$
  Data Sample: Sample state $x_{t,k} \sim \rho$, an expert response $y_{t,k} \sim \pi^{\mathrm{E}}(\cdot|x_{t,k})$ and agent response $\tilde{y}_{t,k} \sim \pi_{\boldsymbol{\theta}_{t,0}}(\cdot|x_{t,k})$, for $k = 0, 1, ..., K-1$
  for $k = 0, 1, ..., K-1$ **do**
    Estimate Gradient: Calculate the stochastic estimator $\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})$ via (6)
    Implicit Reward Alignment: Update $\boldsymbol{\theta}_{t,k+1} = \boldsymbol{\theta}_{t,k} + \eta_t \hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})$
  end for
end for

A.3. Discussions & Implementation Details

Implementation details of RFT. As mentioned, training a reward model and a policy at the same time is costly. In our experiments, we discovered that the reward alignment step can be completely separated from the policy alignment step. In particular, we take $T = 1$ and $K = \frac{\text{data size}}{\text{batch size}} * \text{epoch}$ so that we train the reward over the entire dataset and then switch to the policy alignment. In our experiments, we indeed observe that only one round of above procedure can readily show superior performance over SFT and implicit reward-learning methods for `pythia-1.4b` model.

Implementation details of IRFT. It is worth noticing that in (6), the policy π is not parameterized by $\boldsymbol{\theta}$ directly. In our numerical experiment, we directly parameterize the LLM π by $\boldsymbol{\theta}$, making (6) the gradient of an supervised optimization problem itself. Meanwhile, it is not straightforward to calculate the self-generation gradient (6) directly, thus we need to design a loss function for back-propagation in main-stream packages such as PyTorch and TensorFlow. In practice, at each training iteration we first sample $\tilde{y} \sim \pi(\cdot|x; \boldsymbol{\theta})$ and pass the following loss function

$$\sigma\left(\log\frac{\pi(y|x;\boldsymbol{\theta})}{\pi_{\mathrm{ref}}(y|x)} - \log\frac{\pi(\tilde{y}|x;\boldsymbol{\theta})}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right) \tag{7}$$

into the standard optimizers (such as `SGD` or `Adam`) for back-propagation. Here σ is a non-increasing nonlinear function. We take the same logistic loss function $\sigma(t) := \log(1 + \exp(-t))$ as (Chen et al., 2024) for its non-negativity, smoothness and exponentially decaying tail to avoid excessive growth in the absolute value of the log-likelihood.

Comparison to SPIN. We discuss here the connection between our proposed algorithms with the self-play fine-tune algorithm (SPIN in (Chen et al., 2024)), which also maximizes the gap between two rewards. First, SPIN is motivated by certain two-player games, while in our case, we show that the difference of two rewards in (5) naturally comes from *a single*, *reward learning* agent; see (4).

Second, IRFT covers SPIN as a special case. In particular, if we take $T = 1$ and K as the total number of training iterations, then the IRFT algorithm is equivalent to SPIN. In practice, we tested on different choices of T and show that a reasonable generation frequency can results in a strong model performance.

Finally, since SPIN does not involve explicit reward learning, it is not directly related to RFT. It is worth noting that the relation between the proposed Algorithm 1 and Algorithm 2 is similar to that of RLHF to DPO. There has been intensive discussions regarding whether reward-based or reward-free algorithm gives better model performances, but this topic is beyond the scope of the current paper. We refer to (Xu et al., 2024) for a comprehensive study.

Our convergence result in Theorem 3.3 also indicates that SPIN does not converge, because $T = 1$ implies that there is always an error of $\mathcal{O}(1)$. Therefore, one typically needs to run the algorithm for multiple outer iterations (i.e., generate for the entire dataset multiple times) to claim convergence to the inverse RL problem (4). Numerically, it has indeed been observed that running multiple rounds of SPIN is beneficial; see in Section 4.

A.4. Proofs for Section 3

We first present the proof of Lemma 3.1.

Proof of Lemma 3.1. It is straightforward to see that the lower-level problem in (4) enjoys a closed-form solution:

$$\pi_{\boldsymbol{\theta}}(y|x) = \frac{\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y;\boldsymbol{\theta})\right)}{\sum_{\tilde{y}\in\mathcal{A}}\pi_{\text{ref}}(\tilde{y}|x)\exp\left(\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})\right)} \tag{8}$$

where $\mathcal{A}$ is the set of all possible responses. Plugging (8) into (4), we obtain:

$$\max_{\boldsymbol{\theta}}\ \mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x)}\left[\log\left(\pi_{\text{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y;\boldsymbol{\theta})\right)\right) - \log\left(\sum_{\tilde{y}\in\mathcal{A}}\pi_{\text{ref}}(\tilde{y}|x)\exp\left(\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})\right)\right)\right] \tag{9}$$

Plugging (8) into the lower level problem of (4), we observe the following equivalence on the objective value:

$$\log\left(\sum_{\tilde{y}\in\mathcal{A}}\pi_{\text{ref}}(\tilde{y}|x)\exp\left(\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})\right)\right) = \max_{\pi}\mathbb{E}_{\tilde{y}\sim\pi(\cdot|x)}[\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})] - D_{\text{KL}}\Big(\pi(\cdot|x)\|\pi_{\text{ref}}(\cdot|x)\Big)$$

we obtain the following max-min problem (omitting some constant terms):

$$\max_{\boldsymbol{\theta}}\min_{\pi}\ \mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x),\tilde{y}\sim\pi(\cdot|x)}\left[\frac{r(x,y;\boldsymbol{\theta})-r(x,\tilde{y};\boldsymbol{\theta})}{\beta} + D_{\text{KL}}\Big(\pi(\cdot|x)\|\pi_{\text{ref}}(\cdot|x)\Big)\right] \tag{10}$$

The proof is completed. □

We then present the proof of Lemma 3.2

Proof of Lemma 3.2. Omitting the constant terms not related to $\boldsymbol{\theta}$ in (9), we have

$$\max_{\boldsymbol{\theta}}\ \ell(\boldsymbol{\theta}) = \mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x)}\left[\frac{1}{\beta}r(x,y;\boldsymbol{\theta}) - \log\left(\sum_{\tilde{y}\in\mathcal{A}}\pi_{\text{ref}}(\tilde{y}|x)\exp\left(\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})\right)\right)\right] \tag{11}$$

Calculating the derivative we get

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}) =&\frac{1}{\beta}\mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|s)}[\nabla_{\boldsymbol{\theta}} r(x,y;\boldsymbol{\theta})] - \mathbb{E}_{x\sim\rho}\left[\nabla_{\boldsymbol{\theta}}\log\left(\sum_{\tilde{y}\in\mathcal{A}}\pi_{\mathrm{ref}}(\tilde{y}|x)\exp\left(\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})\right)\right)\right]\\ =&\frac{1}{\beta}\mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x)}[\nabla_{\boldsymbol{\theta}} r(x,y;\boldsymbol{\theta})] - \frac{1}{\beta}\mathbb{E}_{x\sim\rho}\left[\sum_{y\in\mathcal{A}}\frac{\pi_{\mathrm{ref}}(y|x)\exp\left(\frac{1}{\beta}r(x,y;\boldsymbol{\theta})\right)}{\sum_{\tilde{y}\in\mathcal{A}}\pi_{\mathrm{ref}}(\tilde{y}|x)\exp\left(\frac{1}{\beta}r(x,\tilde{y};\boldsymbol{\theta})\right)}\nabla_{\boldsymbol{\theta}} r(x,y;\boldsymbol{\theta})\right]\\ =&\frac{1}{\beta}\mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x)}[\nabla_{\boldsymbol{\theta}} r(x,y;\boldsymbol{\theta})] - \frac{1}{\beta}\mathbb{E}_{x\sim\rho,y\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}[\nabla_{\boldsymbol{\theta}} r(x,y;\boldsymbol{\theta})]\\ =&\frac{1}{\beta}\mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x),\tilde{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|x)}[\nabla_{\boldsymbol{\theta}} r(x,y;\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} r(x,\tilde{y};\boldsymbol{\theta})]\end{aligned}$$

which implies that to minimize $\ell(\boldsymbol{\theta})$, one should always generate samples based on the current estimation of the policy $\tilde{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|x)$ and then update.

Now from (8) we get:

$$r(x,y;\boldsymbol{\theta}) = \beta\log\frac{\pi_{\boldsymbol{\theta}}(y|x)}{\pi_{\mathrm{ref}}(y|x)} + \beta\log Z_{\boldsymbol{\theta}}(x) \tag{12}$$

where $Z_{\boldsymbol{\theta}}(x)$ is the denominator of (8). In the view of (12), we can actually directly estimate:

$$\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}) = \frac{1}{\beta}\mathbb{E}_{x\sim\rho,y\sim\pi^{\mathrm{E}}(\cdot|x),\tilde{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}}\log\frac{\pi_{\boldsymbol{\theta}}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}}\log\frac{\pi_{\boldsymbol{\theta}}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right] \tag{13}$$

The proof is completed. □

Now we move to the proof for Section 3.3. We state the assumption needed for proving the final result:

Assumption A.1. For Algorithm 1 and 2, we assume that

1. The policy distribution $\pi_{\boldsymbol{\theta}}$ is uniformly lower and upper bounded, i.e.

   $$\pi_{\min} \le \|\pi_{\boldsymbol{\theta}}(\cdot|x)\|_\infty \le \pi_{\max}$$

   where $0 < \pi_{\min} < \pi_{\max}$, for all x;
2. $\nabla\pi_{\boldsymbol{\theta}}$ is bounded, i.e. $\|\nabla\pi_{\boldsymbol{\theta}}(\cdot|x)\| \le L_0$ for all x;
3. $\nabla\pi_{\boldsymbol{\theta}}$ is Lipschitz, i.e. $\|\nabla\pi_{\boldsymbol{\theta}_1}(y|x) - \nabla\pi_{\boldsymbol{\theta}_2}(y|x)\| \le L_1\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|$, for all x and y;

where $\pi_{\boldsymbol{\theta}}$ is as defined in (8).

The above assumption can readily establish the assumption below, which is needed for our final convergence result.

Assumption A.2. For Algorithm 2, we assume that

1. ℓ is L-Lipschitz smooth w.r.t. $\boldsymbol{\theta}$, i.e.

   $$\|\nabla\ell(\boldsymbol{\theta}_1) - \nabla\ell(\boldsymbol{\theta}_2)\| \le L\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.$$

2. The stochastic estimator $\hat{\nabla}\ell$ is bounded, i.e.

   $$\|\hat{\nabla}\ell(\boldsymbol{\theta})\| \le G.$$

These are all standard assumptions in nonconvex smooth stochastic optimization. We have the following lemma:

Lemma A.3. *If Assumption A.1 holds, Assumption A.2 also holds with the following parameters:*

$$L = \frac{L_0(3L_0 + L_1)}{\pi_{\min}^2},\ G = \frac{2L_0}{\pi_{\min}}$$

Proof. We just show the value for L since G can be similarly computed. Since

$$\nabla \ell(\boldsymbol{\theta}) = \frac{1}{\beta}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]$$

we have

$$\begin{aligned}
&\|\nabla \ell(\boldsymbol{\theta}_1) - \nabla \ell(\boldsymbol{\theta}_2)\| \\
=&\frac{1}{\beta}\left\|\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_1}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right. \\
&\left. - \mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_2}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right\| \\
\leq&\frac{1}{\beta}\left\|\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_1}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right. \\
&\left. - \mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right\| \\
&+ \frac{1}{\beta}\left\|\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right. \\
&\left. - \mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_2}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right\|.
\end{aligned} \tag{14}$$

For the first part, since

$$\nabla \log \pi_{\boldsymbol{\theta}}(y|x) = \frac{\nabla \pi_{\boldsymbol{\theta}}(y|x)}{\pi_{\boldsymbol{\theta}}(y|x)}$$

we have

$$\begin{aligned}
&\frac{1}{\beta}\left\|\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_1}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} + \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right\| \\
\leq&\frac{1}{\beta}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left\|\frac{\nabla\pi_{\boldsymbol{\theta}_1}(y|x)}{\pi_{\boldsymbol{\theta}_1}(y|x)} - \frac{\nabla\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\boldsymbol{\theta}_2}(y|x)} - \frac{\nabla\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)}{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)} + \frac{\nabla\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}\right\| \\
\leq&\frac{1}{\beta}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left\|\frac{\nabla\pi_{\boldsymbol{\theta}_1}(y|x)}{\pi_{\boldsymbol{\theta}_1}(y|x)} - \frac{\nabla\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\boldsymbol{\theta}_2}(y|x)}\right\| + \frac{1}{\beta}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left\|\frac{\nabla\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)}{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)} - \frac{\nabla\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}\right\| \\
\leq&\frac{1}{\beta}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\frac{\|\pi_{\boldsymbol{\theta}_2}(y|x)\nabla\pi_{\boldsymbol{\theta}_1}(y|x) - \pi_{\boldsymbol{\theta}_1}(y|x)\nabla\pi_{\boldsymbol{\theta}_2}(y|x)\|}{\pi_{\boldsymbol{\theta}_1}(y|x)\pi_{\boldsymbol{\theta}_2}(y|x)} + (\text{same term for } \tilde{y}) \\
\leq&\frac{2}{\beta}\frac{\pi_{\max}L_1 + L_0^2}{\pi_{\min}^2}\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.
\end{aligned}$$

For the second term in the last line of (14), we have

$$\begin{aligned}
&\frac{1}{\beta}\left\|\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_1}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right. \\
&\left. - \mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x), \tilde{y}\sim\pi_{\boldsymbol{\theta}_2}(\cdot|x)}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right]\right\| \\
\leq&\frac{1}{\beta}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x)}\left\|\sum_{\tilde{y}\in\mathcal{A}}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(y|x)}{\pi_{\mathrm{ref}}(y|x)} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\mathrm{ref}}(\tilde{y}|x)}\right](\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x) - \pi_{\boldsymbol{\theta}_2}(\tilde{y}|x))\right\| \\
\leq&\frac{1}{\beta}\frac{2L_0}{\pi_{\min}}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x)}\left\|\sum_{\tilde{y}\in\mathcal{A}}(\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x) - \pi_{\boldsymbol{\theta}_2}(\tilde{y}|x))\right\| \\
=&\frac{1}{\beta}\frac{2L_0}{\pi_{\min}}\mathbb{E}_{x\sim\rho, y\sim\pi^{\mathrm{E}}(\cdot|x)}\left\|\sum_{\tilde{y}\in\mathcal{A}}\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)\frac{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x) - \pi_{\boldsymbol{\theta}_2}(\tilde{y}|x)}{\pi_{\boldsymbol{\theta}_1}(\tilde{y}|x)}\right\| \leq \frac{1}{\beta}\frac{2L_0^2}{\pi_{\min}^2}\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.
\end{aligned}$$

Plugging these back to (14) we get

$$\|\nabla \ell(\boldsymbol{\theta}_1) - \nabla \ell(\boldsymbol{\theta}_2)\| \le \frac{2}{\beta}\left(\frac{\pi_{\max} L_1 + 2L_0^2}{\pi_{\min}^2}\right)\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.$$

□

Now since we generate at the beginning of the inner loop, the estimator $\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})$ is not an unbiased estimator of $\nabla \ell(\boldsymbol{\theta}_{t,k})$ for any $k > 0$, i.e.

$$\nabla \ell(\boldsymbol{\theta}_{t,k}) = \frac{1}{\beta}\mathbb{E}_{(x_{t,k},y_{t,k})\sim\mathcal{D},\tilde{y}_{t,k}\sim\pi_{\boldsymbol{\theta}_{t,k}}(\cdot|x_{t,k})}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_{t,k}}(y_{t,k}|x_{t,k})}{\pi_{\text{ref}}(y_{t,k}|x_{t,k})} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_{t,k}}(\tilde{y}_{t,k}|x_{t,k})}{\pi_{\text{ref}}(\tilde{y}_{t,k}|x_{t,k})}\right] \tag{15}$$

$$\neq \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}) = \frac{1}{\beta}\mathbb{E}_{(x_{t,k},y_{t,k})\sim\mathcal{D},\tilde{y}_{t,k}\sim\pi_{\boldsymbol{\theta}_{t,0}}(\cdot|x_{t,k})}\left[\nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_{t,k}}(y_{t,k}|x_{t,k})}{\pi_{\text{ref}}(y_{t,k}|x_{t,k})} - \nabla_{\boldsymbol{\theta}} \log \frac{\pi_{\boldsymbol{\theta}_{t,k}}(\tilde{y}_{t,k}|x_{t,k})}{\pi_{\text{ref}}(\tilde{y}_{t,k}|x_{t,k})}\right] \tag{16}$$

We thus need to carefully analyze this biasedness so that the convergence can be boosted by a large K, since a large K will result in a very large bias.

Now we are ready to re-state and prove Theorem 3.3:

Theorem A.4. *Suppose Assumption A.1 holds, then for Algorithm 1 and 2 with* $\eta_t = \Theta(1/\sqrt{TK})$ *we have*

$$\min_{t=1,\ldots,T,\ k=1,\ldots,K} \mathbb{E}[\|\nabla \ell(\boldsymbol{\theta}_{t,k})\|^2] \le \mathcal{O}\left(\frac{\Delta_0 + LG^2}{\sqrt{TK}} + \frac{\tilde{L}^2 G^2}{T}\right)$$

where $\Delta_0 = \ell^* - \ell(\boldsymbol{\theta}_0)$ *and we omit constant factors in* $\tilde{\mathcal{O}}$.

Proof. We prove directly for Algorithm 2 since the gradient estimator (6) and the estimator $g_{t,k}$ Algorithm 1 (we do solve the π subproblem to its optimum) are both for the original bilevel problem (4).

From the Lipschitz gradient of ℓ we have

$$\ell(\boldsymbol{\theta}_{t,k+1}) \ge \ell(\boldsymbol{\theta}_{t,k}) + \eta_t\langle \hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}), \nabla \ell(\boldsymbol{\theta}_{t,k})\rangle - \frac{\eta_t^2 L}{2}\|\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})\|^2$$

i.e.

$$\eta_t\|\nabla \ell(\boldsymbol{\theta}_{t,k})\|^2 \le (\ell(\boldsymbol{\theta}_{t,k+1}) - \ell(\boldsymbol{\theta}_{t,k})) + \eta_t\langle \nabla \ell(\boldsymbol{\theta}_{t,k}) - \hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}), \nabla \ell(\boldsymbol{\theta}_{t,k})\rangle + \frac{\eta_t^2 L}{2}\|\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})\|^2$$

Taking expectation to $\boldsymbol{\theta}_{t,k}$ and by Assumption A.2, we have

$$\eta_t\mathbb{E}\|\nabla \ell(\boldsymbol{\theta}_{t,k})\|^2 \le (\mathbb{E}\ell(\boldsymbol{\theta}_{t,k+1}) - \ell(\boldsymbol{\theta}_{t,k})) + \eta_t\langle \nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}), \nabla \ell(\boldsymbol{\theta}_{t,k})\rangle + \frac{\eta_t^2 LG^2}{2}$$

where the expectation is taken w.r.t. the sample $\tilde{y}_{t,k}$ to generate the estimator of current iteration.

Sum up from $k = 0$ to $k = K$ we get

$$\sum_{k=0}^{K-1} \eta_t\mathbb{E}\|\nabla \ell(\boldsymbol{\theta}_{t,k})\|^2 \le (\mathbb{E}\ell(\boldsymbol{\theta}_{t,K-1}) - \ell(\boldsymbol{\theta}_{t,0})) + \eta_t\sum_{k=0}^{K-1}\langle \nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}), \nabla \ell(\boldsymbol{\theta}_{t,k})\rangle + \frac{\eta_t^2 LG^2 K}{2} \tag{17}$$

Since the expectation is taken only on the random sample at current iteration, and we know that the true gradient and the approximated gradient are (15) and (16), we have the following estimate:

$$\begin{aligned}
&\|\nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})\| \\
=&\frac{1}{\beta}\left\|\mathbb{E}_{x_{t,k}\sim\rho, y_{t,k}\sim\pi^{\mathrm{E}}(\cdot|x_{t,k}), \tilde{y}_{t,k}\sim\pi_{\boldsymbol{\theta}_{t,k}}(\cdot|x_{t,k})}\left[\nabla_{\boldsymbol{\theta}} \log\frac{\pi_{\boldsymbol{\theta}_{t,k}}(y_{t,k}|x_{t,k})}{\pi_{\mathrm{ref}}(y_{t,k}|x_{t,k})} - \nabla_{\boldsymbol{\theta}}\log\frac{\pi_{\boldsymbol{\theta}_{t,k}}(\tilde{y}_{t,k}|x_{t,k})}{\pi_{\mathrm{ref}}(\tilde{y}_{t,k}|x_{t,k})}\right]\right. \\
&\left.- \ \mathbb{E}_{x_{t,k}\sim\rho, y_{t,k}\sim\pi^{\mathrm{E}}(\cdot|x_{t,k}), \tilde{y}_{t,k}\sim\pi_{\boldsymbol{\theta}_{t,k}}(\cdot|x_{t,k})}\left[\nabla_{\boldsymbol{\theta}} \log\frac{\pi_{\boldsymbol{\theta}_{t,k}}(y_{t,k}|x_{t,k})}{\pi_{\mathrm{ref}}(y_{t,k}|x_{t,k})} - \nabla_{\boldsymbol{\theta}}\log\frac{\pi_{\boldsymbol{\theta}_{t,k}}(\tilde{y}_{t,k}|x_{t,k})}{\pi_{\mathrm{ref}}(\tilde{y}_{t,k}|x_{t,k})}\right]\right\| \\
=&\frac{1}{\beta}\left\|\mathbb{E}_{x_{t,k},y_{t,k}}\int\left[\nabla_{\boldsymbol{\theta}} \log\frac{\pi_{\boldsymbol{\theta}_{t,k}}(y_{t,k}|x_{t,k})}{\pi_{\mathrm{ref}}(y_{t,k}|x_{t,k})} - \nabla_{\boldsymbol{\theta}}\log\frac{\pi_{\boldsymbol{\theta}_{t,k}}(\tilde{y}|x_{t,k})}{\pi_{\mathrm{ref}}(\tilde{y}|x_{t,k})}\right]\left(\pi_{\boldsymbol{\theta}_{t,k}}(\tilde{y}|x_{t,k}) - \pi_{\boldsymbol{\theta}_{t,0}}(\tilde{y}|x_{t,k})\right)d\tilde{y}\right\| \\
\leq&\frac{1}{\beta}\frac{2L_0^2}{\pi_{\min}}\|\boldsymbol{\theta}_{t,k} - \boldsymbol{\theta}_{t,0}\|
\end{aligned}$$

Denote $\tilde{L} = \frac{1}{\beta}\frac{2L_0^2}{\pi_{\min}}$, we thus have:

$$\begin{aligned}
&\sum_{k=0}^{K-1}\langle\nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}), \nabla\ell(\boldsymbol{\theta}_{t,k})\rangle \leq \sum_{k=0}^{K-1}\|\nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})\|\|\nabla\ell(\boldsymbol{\theta}_{t,k})\| \\
&\leq \frac{1}{2}\sum_{k=0}^{K-1}\|\nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k})\|^2 + \frac{1}{2}\sum_{k=0}^{K-1}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2 \\
&\leq \frac{\tilde{L}^2}{2}\sum_{k=0}^{K-1}\|\boldsymbol{\theta}_{t,k} - \boldsymbol{\theta}_{t,0}\|^2 + \frac{1}{2}\sum_{k=0}^{K-1}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2 = \frac{\eta_t^2\tilde{L}^2}{2}\sum_{k=0}^{K-1}\left\|\sum_{i=0}^{k-1}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,i})\right\|^2 + \frac{1}{2}\sum_{k=0}^{K-1}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2
\end{aligned}$$

Therefore

$$\sum_{k=0}^{K-1}\langle\nabla \ell(\boldsymbol{\theta}_{t,k}) - \mathbb{E}\hat{\nabla}\ell(\boldsymbol{\theta}_{t,k}), \nabla\ell(\boldsymbol{\theta}_{t,k})\rangle \leq \frac{\eta_t^2\tilde{L}^2G^2}{2}\frac{K(K-1)}{2} + \frac{1}{2}\sum_{k=0}^{K-1}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2$$

Substituting back into (17) leads to

$$\frac{1}{2}\sum_{k=0}^{K-1}\eta_t\mathbb{E}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2 \leq (\mathbb{E}\ell(\boldsymbol{\theta}_{t,K-1}) - \ell(\boldsymbol{\theta}_{t,0})) + \eta_t^3\tilde{L}^2G^2\frac{K(K-1)}{2} + \frac{\eta_t^2LG^2K}{2}$$

Summing up from $t = 0$ to $T - 1$ gives

$$\frac{1}{2}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\eta_t\mathbb{E}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2 \leq \mathbb{E}\ell(\boldsymbol{\theta}_{T-1,K-1}) - \ell(\boldsymbol{\theta}_{-1,0}) + \sum_{t=0}^{T-1}\eta_t^3\tilde{L}^2G^2\frac{K(K-1)}{2} + \sum_{t=0}^{T-1}\frac{\eta_t^2LG^2K}{2}$$

With a constant step size $\eta_t = \eta > 0$, we have

$$\frac{1}{2TK}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\mathbb{E}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2 \leq \frac{\mathbb{E}\ell(\boldsymbol{\theta}_{T-1,K-1}) - \ell(\boldsymbol{\theta}_{-1,0})}{\eta TK} + \eta^2\tilde{L}^2G^2\frac{K-1}{2} + \eta\frac{LG^2}{2}$$

Taking $\eta = \Theta(1/\sqrt{TK})$, we get

$$\frac{1}{TK}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\mathbb{E}\|\nabla\ell(\boldsymbol{\theta}_{t,k})\|^2 = \mathcal{O}\left(\frac{\mathbb{E}\ell(\boldsymbol{\theta}_{T-1,K-1}) - \ell(\boldsymbol{\theta}_{-1,0}) + LG^2}{\sqrt{TK}} + \frac{\tilde{L}^2G^2}{T}\right)$$

Hence as $T \to \infty$, the rate is $\mathcal{O}(1/\sqrt{TK})$. $\square$

A.5. Details of the Numerical Experiments

Model and Datasets. Since reward-based methods can be costly by training two models at the same time, we mainly test Algorithm 1 on `pythia-1b` reward model and `pythia-1.4b` policy model (Biderman et al., 2023). We tested pythia on Anthropic-HH dataset (Bai et al., 2022). Anthropic-HH is a preference dataset that provide two continuations based on helpfulness and harmlessness, and we only pick 10k chosen/preferred continuation data to form the demonstration dataset, which enable us to check the log likelihood of the non-preferred continuation without feeding the model with such data. At each iteration, we train our model for 2 epochs (seeing each data for two times). We then use `PKU-Alignment/beaver-7b-v3.0-reward` model as our ground truth reward model. We use this model to pick 10k data from Anthropic-HH dataset with the highest reward scores. The win rate is calculated as the ratio of samples where the reward of our model's generation is higher than the model compared.

Algorithm 2 is tested on two models: `pythia-1.4b` and `zephyr-7b-sft-full` (Tunstall et al., 2023). We tested on Ultrachat200k dataset by HuggingFace, which is a subset of the high quality demonstration UltraChat dataset(Ding et al., 2023) for text generation and dialogue. For Ultrachat200k, we adopt the same strategy as (Chen et al., 2024) to pick up 50k data for training. At each iteration, we again train our model for 2 epochs.

Evaluation. For the Anthropic-HH dataset, we use `PKU-Alignment/beaver-7b-v3.0-reward` (Dai et al., 2024; Ji et al., 2023) model as an evaluator; it is a popular 7b model fine-tuned from `meta-llama/Llama-2-7b` tailored for evaluating human preferences regarding helpfulness and harmlessness. We also record win rate of the two proposed methods over base model and SFT model. For the Ultrachat200k dataset, we follow the widely used HuggingFace Open LLM Leaderboard (Beeching et al., 2023). This evaluation package assess an LLM based on six tasks: LLMs on commonsense reasoning (Arc (Clark et al., 2018), HellaSwag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2021)), multi-task language understanding (MMLU (Hendrycks et al., 2020)), human falsehood mimic (TruthfulQA (Lin et al., 2021)) and math problem solving (GSM8K, (Cobbe et al., 2021)). See the appendix for implementation details.

We follow the code as in SPIN (Chen et al., 2024), where we utilize DeepSpeed ZeRO-3 (Rajbhandari et al., 2020) and FlashAttention-2 (Dao, 2023) to reduce the memory cost. We use RMSProp (Hinton et al., 2012) optimizer with no weight decay. For 1b models, we use two NVIDIA A100-40G to do the training with per device batch size of 4 for Algorithm 1 and per device batch size of 8 for Algorithm 2. For 7b models we use eight NVIDIA A100-40G to do the training with per device batch size of 2. We train all models with bfloat16 precision. We set the peak learning rate to be 5e-7 for first two epochs and 1e-7 for the next two epochs. We fix $\beta = 0.1$ and consider the max sequence length to be 1024 for 1b models and 2048 for 7b models. We use the same prompt template "### Instruction: prompt\n\n### Response: " as in (Chen et al., 2024). For the policy optimization step in Algorithm 1, we use the PPO trainer in the TRL package (von Werra et al., 2020). For the HuggingFace Open LLM Leaderboard evaluation, we use the Language Model Evaluation Harness library (v0.4.2) (Gao et al., 2023), and we also use the same number of few-shots as in (Chen et al., 2024).

Finally, in Table 4, we further provide the generation examples of our fine-tuned model in Table 2.

A.6. Conclusions and Limitations

In this paper we proposed reward-learning approaches for aligning LLMs with demonstration datasets. We show both theoretically and numerically the great potential of reward-learning for alignment even without preference dataset. Our theory only indicate the convergence of the proposed algorithm to stationary point, and it is not clear what the policy converges to. The additional computation resources required for tuning two models or generate synthetic data in our algorithms are not negligible. Future works include exploring reward-learning for larger models and more complicated demonstration tasks, boosting the algorithm efficiency, and understanding how synthetic negative sample helps the LLMs to distinguish the preference dataset, etc.

Table 4. Generation example of fine-tuned models in Table 2.

Prompt	Craft an intriguing opening paragraph for a fictional short story. The story should involve a character who wakes up one morning to find that they can time travel.
`zephyr-7b-sft-full`	It was just another ordinary morning for Emily, until she opened her eyes and found herself in a different time and place. She was lying on her bed, surrounded by strange decor and people dressed in clothes from a bygone era. Emily rubbed her eyes, thinking she must be dreaming. But as she got up, she realized that she could feel the weight of the day ahead of her, much like she always did. It was only then that she realized that she had the power to time travel.
IRFT $T = 1$ (SPIN iter 0)	As soon as her eyes fluttered open, something was different. The room looked the same, the sun streaming in through the same window, but something felt off. It wasn't until she tried to sit up that she realized what it was - her body felt lighter, as if she were floating on air. Confused, she looked around, searching for an explanation, and that's when she saw it - the clock on her bedside table was ticking backwards. Panic set in as she tried to sit up, but her body refused to obey her commands. And just like that, she was hurled through time, landing with a thud in a completely different era.
IRFT $T = 1$ (SPIN iter 1)	As soon as the alarm clock shrilled, something felt different to James, but he brushed it aside and reached out to turn it off, just like any other morning. Yet, as soon as he sat up, he felt a strange pull as if he was being dragged backwards, and his room began to spin in a dizzying whirlwind of colors. In the blink of an eye, he found himself standing in a different place altogether, and he knew that he had just woken up in a different time.
IRFT $T = 5$ (2 epochs)	I opened my eyes groggily, ready to start another mundane day. But as I rubbed my eyes and stretched my arms, something strange happened. I felt a sudden jolt of energy course through my veins, and before I knew it, the world around me had transformed. The room that I was in was no longer my own, and I was standing in the middle of a bustling medieval town square. I blinked in disbelief, trying to make sense of what had just happened. Had I gone mad, or was I dreaming? But as I looked around me, I realized with awe that I had somehow managed to time travel.
IRFT $T = 5$ (4 epochs)	It was just another regular morning for Emily, until as soon as she opened her eyes the room started to blur and spin. Panicking, she tried to stand up and grab the nightstand but her hand went right through it. Confused and terrified, she tried to scream, but as soon as the sound started to come out of her throat, she was engulfed by a bright light that covered her body. When Emily opened her eyes again, she realized that she had traveled back in time, to a moment when she was 9 years old and standing at the foot of her parents' bed, ready to tell them the good news about acing her history test.