

Direct Multi-Turn Preference Optimization for Language Agents

Anonymous ACL submission

Abstract

Adapting Large Language Models (LLMs) for agent tasks is critical in developing language agents. Direct Preference Optimization (DPO) is a promising technique for this adaptation with the alleviation of compounding errors, offering a means to directly optimize Reinforcement Learning (RL) objectives. However, applying DPO to multi-turn tasks presents challenges due to the inability to cancel the partition function. Overcoming this obstacle involves making the partition function independent of the current state and addressing length disparities between preferred and dis-preferred trajectories. In this light, we replace the policy constraint with the state-action occupancy measure constraint in the RL objective and add length normalization to the Bradley-Terry model, yielding a novel loss function named DMPO for multi-turn agent tasks with theoretical explanations. Extensive experiments on three multi-turn agent task datasets confirm the effectiveness and superiority of the DMPO loss.

1 Introduction

Developing generalist agents capable of solving complex tasks has been a central goal in the artificial intelligence community (Reed et al., 2022; Team et al., 2024). Recently, *Language agents* (Yao et al., 2023) emerge as a prominent research direction, leveraging the considerable potential of Large Language Models to address intricate tasks involving instruction following (Ouyang et al., 2022), action planning (Huang et al., 2022), and tool utilization (Schick et al., 2023). Nevertheless, the substantial disparity between the pretraining task of Large Language Models and the requirements of agent tasks suggests significant potential for future advancements in language agent capabilities.

Behavioral Cloning (BC) (Pomerleau, 1991) is a frequently employed approach to bridge the domain gap by fine-tuning LLMs through expert agent trajectories. Recent endeavors in BC (Chen

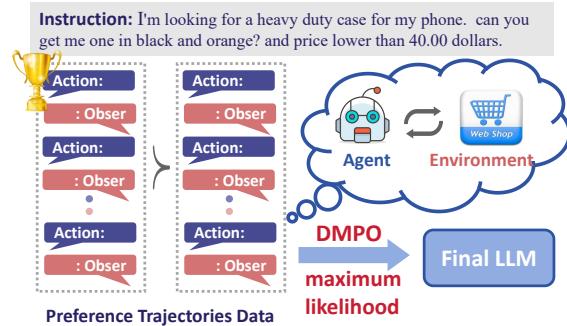


Figure 1: Illustration of DMPO loss, which directly optimizes the RL objective by maximizing the likelihood of the preferred trajectory over the dispreferred trajectory.

et al., 2023; Zeng et al., 2023; Yin et al., 2023) involve the Supervised Fine-tuning of LLMs on optimal state-action pairs. Although these methods enable swift adaptation of LLMs to agent tasks, BC is notably susceptible to *compounding errors* — minor errors of the learner accumulate along interactions between the agent and environment, leading to performance deterioration in non-deterministic environments (Ross et al., 2011).

In alleviating compounding errors, Direct Preference Optimization (Rafailov et al., 2023) has demonstrated remarkable success in the single-turn preference alignment task due to its simple implementation and robustness. DPO optimizes RL objectives by maximizing the likelihood of preferred responses over dis-preferred responses, mitigating the need for continuous interaction with the environment and the training instability commonly associated with traditional RL algorithms (Christianos et al., 2023; Liang et al., 2024). Although there has been an initial endeavor to apply the DPO loss on LLMs for agent tasks (Song et al., 2024), it encounters suboptimal performance, as it is tailored specifically for the single-turn bandit setting and is ill-suited for multi-turn agent tasks.

This work aims to develop a robust loss function capable of directly optimizing RL objectives

in multi-turn scenarios. The crux of this pursuit involves eliminating the partition function in the Bradley-Terry (BT) model (Bradley and Terry, 1952; Christiano et al., 2017). This entails ensuring the partition function’s independence from the current state and neutralizing the impact of the length disparity between preferred and dis-preferred trajectories. To achieve this, we substitute the policy constraint with the state-action occupancy measure (SAOM) (Johnson et al., 2000) constraint in the RL objective and introduce length normalization into the BT model. These adjustments culminate in the development of a new and simple loss function DMPO for multi-turn agent tasks. As shown in Figure 1, DMPO directly optimizes the RL objective by maximizing the likelihood of preferred ("win") trajectory over dis-preferred ("lose") trajectory. Notably, the SAOM constraint has advantages in mitigating compounding errors compared to the policy constraint (Xu et al., 2020; Ghasemipour et al., 2019). Furthermore, the derivation offers a theoretical rationale for the efficacy of the length normalization technique in DPO loss (Meng et al., 2024).

To summarize, our contributions are threefold:

- We introduce a new loss function called DMPO, which directly optimizes RL objectives in multi-turn scenarios, thereby mitigating the compounding errors associated with BC methods.
- We provide a theoretical explanation for the efficacy of the length normalization technique, illustrating how it cancels out the partition function in the BT model and improves performance.
- Extensive experiments on three multi-turn agent task datasets validate the effectiveness and the superiority of the DMPO loss function.

2 Related Work

In this section, we first introduce the in-context learning methods and fine-tuning methods of language agents and then review the literature in preference-based RL.

In-Context Learning Inspired by the superior in-context learning capabilities of LLMs (OpenAI et al., 2024), researchers have designed various instruction prompts for LLMs, equipped with memory modules (Zhang et al., 2024), toolkits (Qu et al., 2024), and various workflows (Sumers et al., 2023),

to build language agents for various real-world domains. ReAct (Yao et al., 2023) incorporates CoT reasoning (Wei et al., 2022) into action generation. Reflexion (Shinn et al., 2023) and PROMST (Chen et al., 2024) refine the prompt using environment feedback. However, these in-context learning methods fail to fully exploit the potential of LLMs, since most LLMs are not specifically trained for agent tasks. This work focuses on adapting the LLMs to agent tasks through fine-tuning.

Agent Tuning Recent studies, including FireAct (Chen et al., 2023), AgentTuning (Zeng et al., 2023), Lumos (Yin et al., 2023), MIMIR (Deng et al., 2024), AUTOACT (Qiao et al., 2024), and α -UMI (Shen et al., 2024) supervised fine-tuning LLMs with self-instruct or expert trajectories. However, such BC approaches suffer from compounding errors when interacting with dynamic environments. Taking a step further, Pangu (Christianson et al., 2023) and CMAT (Liang et al., 2024) utilize RL technologies to further fine-tune the LLMs, which may result in a complex and unstable training procedure. To simplify the procedure, ETO (Song et al., 2024) and EMMA (Yang et al., 2023) directly employ the DPO loss (Rafailov et al., 2023) to optimize the RL objective for the agent task. Nevertheless, the DPO loss is designed for single-turn bandit settings and is ill-suited for multi-turn scenarios. Along this line, this work extends the DPO loss in multi-turn scenarios and derives the DMPO loss.

Preference-Based RL In multi-turn scenarios, preference-based RL typically starts by explicitly learning a reward function from preference data and then optimizing it (Fürnkranz et al., 2012; Christiano et al., 2017; III and Sadigh, 2022; Shin and Brown, 2021). However, this two-stage learning process presents challenges regarding training efficiency and instability. This work instead presents a single-stage policy learning approach using DMPO loss that directly optimizes a policy to satisfy preferences. While IPL (Hejna and Sadigh, 2023) and CPL (Hejna et al., 2023) share a similar idea with our work in eliminating the reward learning stage, their loss functions are limited to trajectory pairs of equal length, significantly restricting their applicability.

3 Preliminary

In this section, we present multi-turn agent task formulation and briefly introduce Direct Preference Optimization (DPO) loss.

3.1 Task Description

The agent task can be formulated as a Markov decision process (MDP). A MDP is a 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, where \mathcal{S} denotes the state space, \mathcal{A} denotes action space, \mathcal{T} denotes dynamic transition function $\mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$, \mathcal{R} denotes reward function $\mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$, and $\gamma \in [0, 1]$ is the discount factor. The goal for the agent is to choose actions at each time step that maximize the expected future discounted reward $\mathbb{E} \left[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \right]$, where T is the trajectory length.

In the language agent setting (Christianos et al., 2023), the state space and action space are both subsets of the language space. For the initial state $s_0 \in \mathcal{S}$, it contains the task instruction and prompt. At each time step t , LLMs generate action a_t according to the policy $\pi_\theta(a_t|s_t)$ with the parameter θ . Then the environment will return dynamic feedback o_t and transport the state into s_{t+1} . Note that the new state s_{t+1} is just a simple combination of s_t , a_t , and o_t , and the trajectory $\tau = (s_0, a_0, s_1, a_1, \dots, s_T, a_T)$.

3.2 Direct Preference Optimization

The aim of the DPO loss is to directly optimize RL objectives with KL divergence constraints on the policy function:

$$\max_{\pi_\theta} \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \right] - \beta \mathbb{D}_{KL}[\pi_\theta(a_t|s_t) || \pi_{ref}(a_t|s_t)], \quad (1)$$

where \mathbb{E} is the expectation function, $\mathbb{D}_{KL}[\cdot || \cdot]$ denotes the KL divergence between two distributions, π_{ref} denotes a reference policy, and the β is a parameter controlling the deviation from the base reference policy π_{ref} . The DPO loss is tailored for the single-turn preference alignment setting, where the trajectory length (T) is limited to 1.

Notably, the reward function is learned through the Bradley-Terry (BT) model (Bradley and Terry, 1952; Christiano et al., 2017):

$$p(a_0^w \succ a_0^l | s_0) = \frac{\exp(r(s_0, a_0^w))}{\exp(r(s_0, a_0^w)) + \exp(r(s_0, a_0^l))}, \quad (2)$$

which gives the probability that the “win” action a_0^w is preferred to the “lose” action a_0^l given the state s_0 .

Then DPO leverages the established closed-form solution for the single-turn formulation of the reinforcement learning problem in Eq (1) presented in (Ziebart et al., 2008; Ziebart, 2010):

$$\pi^*(a|s) = \frac{1}{Z(s)} \pi_{ref}(a|s) e^{r(s,a)}, \quad (3)$$

where π^* denotes the optimal policy and $Z(s)$ denotes the partition function that normalizes it. We can easily rearrange Eq (3) and substitute it into Eq (2) to get the BT model over policy:

$$p(a_0^w \succ a_0^l | s_0) = \sigma \left(\beta \frac{\pi_\theta(a_0^w | s_0)}{\pi_{ref}(a_0^w | s_0)} - \beta \frac{\pi_\theta(a_0^l | s_0)}{\pi_{ref}(a_0^l | s_0)} \right), \quad (4)$$

where the partition function $Z(s)$ is canceled from the BT model and σ is the sigmoid function. The DPO loss obtains the optimal policy π_θ^* by maximizing the likelihood:

$$\mathcal{L}_{DPO} = -\mathbb{E}_{(s_0, a_0^w, a_0^l) \sim D} \log \left[p(a_0^w \succ a_0^l | s_0) \right], \quad (5)$$

where D represents the preference dataset. Nonetheless, such concise and elegant derivations are only suitable for single-turn preference optimization tasks. As shown in Eq (3), the partition function $Z(s)$ is dependent on the current state s , which precludes its cancellation under the policy constraint in the multi-turn setting.

4 Method

In this section, we will outline the definition and benefits of the state-action occupancy measure. Subsequently, we will introduce two adjustments to derive the DMPO loss. Finally, we will delve deeper into the analysis of the DMPO loss.

4.1 State-Action Occupancy Measure

The discounted state-action occupancy measure $d^\pi(s, a)$ of a policy π describes the distribution of state-action pairs that an agent visits in the space with policy π :

$$d^\pi(s, a) = \frac{1 - \gamma}{1 - \gamma^T} \sum_{t=0}^{T-1} \gamma^t \mathbb{P}(s_t = s, a_t = a | \pi), \quad (6)$$

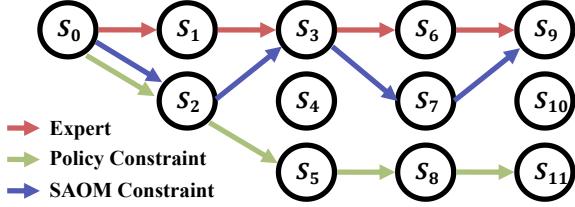


Figure 2: Illustration of expert trajectories and trajectories learned under the constraints of policy and state-action occupancy measure.

where $\mathbb{P}(\cdot)$ denotes the probability and the coefficient $(1 - \gamma)/(1 - \gamma^T)$ is used to normalize the probability distribution.

First, we will provide an intuitive explanation of how the SAOM constraint can reduce the compounding error. In imitation learning, the conventional SFT learning objective aims to minimize the KL divergence between the expert policy and the current policy:

$$\begin{aligned} & \min_{\pi_\theta} \mathbb{E}_{(s,a) \sim d^E} [\mathbb{D}_{KL}(\pi_E(a|s) || \pi_\theta(a|s))] \\ &= - \max_{\pi_\theta} \mathbb{E}_{(s,a) \sim d^E} [\log(\pi_\theta(a|s))], \end{aligned} \quad (7)$$

where π_E is the expert policy and d^E is the SAOM with policy π_E . As shown in Figure 2, the trajectories learned under policy constraints are susceptible to significant compounding error. This vulnerability stems from the fact that expert datasets are unable to comprehensively cover all possible states. Consequently, the SFT loss leads the model to choose random actions in states that are not represented in the expert datasets. As a result, the model gradually deviates from the expert trajectories after the initial error, illustrating the phenomenon known as compounding error.

To alleviate the compounding error, subsequent imitation learning research such as (Abbeel and Ng, 2004; Ghasemipour et al., 2019; Ho and Ermon, 2016) employ the SAOM constraint:

$$\min_{\pi_\theta} \mathbb{E}_{(s,a) \sim d^E} [\mathbb{D}_{(\cdot)}(d^{\pi_\theta}(a|s) || d^{\pi_E}(a|s))], \quad (8)$$

where different approaches utilize different distribution distance measures $\mathbb{D}_{(\cdot)}$. The strength of SAOM constraint lies in its ability to steer action selection towards distributions that closely mimic expert state-action pairs, especially in unexplored states within the expert datasets. Illustrated in Figure 2, at state s_2 , policy constraints lead the model to choose actions uniformly, whereas SAOM constraints aim to lead the model toward actions that

bring the next state back onto the expert trajectory. This effectively mitigates compounding errors and enhances the cumulative reward.

4.2 DMPO

Inspired by imitation learning, we substitute the policy constraint with the SAOM constraint in Eq (1) and get the following RL objective:

$$\begin{aligned} & \max_{\pi_\theta} \mathbb{E}_{(s,a) \sim d^{\pi_\theta}(s,a)} [r(s,a)] \\ & \quad - \beta \mathbb{D}_{KL}[d^{\pi_\theta}(s,a) || d^{\pi_{ref}}(s,a)], \end{aligned} \quad (9)$$

where π_{ref} represents the reference policy. Similar to (Rafailov et al., 2023), it is straightforward to show that the optimal solution to the RL objective in Eq (9) takes the form:

$$d^*(s,a) = \frac{1}{Z} d^{\pi_{ref}}(s,a) \exp\left(\frac{1}{\beta} r(s,a)\right), \quad (10)$$

where π^* represents the optimal policy, Z is the partition function that normalizes the probability. It's noteworthy that as $d^\pi(s,a)$ is a function of (s,a) pairs, normalizing it results in the partition functions Z being independent of the current state s . Consequently, Z remains constant for all (s,a) pairs, providing us with the opportunity to eliminate them. Easily, we can rearrange Eq (10) into:

$$r(s,a) = \beta \log \frac{d^*(s,a)}{d^{\pi_{ref}}(s,a)} + \beta \log Z. \quad (11)$$

Similar to Eq (2), we learn the reward function for multi-turn scenarios through the BT model:

$$p(\tau^w \succ \tau^l | s_0) = \sigma \left(\sum_{t=0}^{T_w-1} \gamma^t r(s_t^w, a_t^w) - \sum_{t=0}^{T_l-1} \gamma^t r(s_t^l, a_t^l) \right), \quad (12)$$

where τ^w and τ^l represent the "win" and "lose" trajectories respectively, T_w and T_l represent the "win" and "loss" trajectory length respectively. However, since $T^w \neq T^l$, the partition function Z cannot be canceled directly in Eq (12).

To overcome this obstacle, we introduce the length normalization technique to Eq (12):

$$p(\tau^w \succ \tau^l | s_0) = \sigma \left(\frac{1 - \gamma}{1 - \gamma^{T_w}} \sum_{t=0}^{T_w-1} \gamma^t r(s_t^w, a_t^w) \right. \\ \left. - \frac{1 - \gamma}{1 - \gamma^{T_l}} \sum_{t=0}^{T_l-1} \gamma^t r(s_t^l, a_t^l) \right). \quad (13)$$

In this way, we can eliminate the partition function Z in Eq (13) by substituting the reward function $r(s, a)$ in Eq (11). Then we maximize the likelihood and obtain:

$$L_{DMPO} = -\mathbb{E}_{(s_0, \tau^w, \tau^l) \sim D} \log \sigma \left[\frac{1-\gamma}{1-\gamma^{T_w}} \sum_{t=0}^{T_w-1} \frac{\gamma^t d^{\pi_\theta}(s_t^w, a_t^w)}{d^{\pi_{ref}}(s_t^w, a_t^w)} - \frac{1-\gamma}{1-\gamma^{T_l}} \sum_{t=0}^{T_l-1} \frac{\gamma^t d^{\pi_\theta}(s_t^l, a_t^l)}{d^{\pi_{ref}}(s_t^l, a_t^l)} \right], \quad (14)$$

where the $d^\pi(s_t, a_t)$ can be further written as:

$$d^\pi(s_t^w, a_t^w) = \gamma^t \cdot P(s_0) \cdot \sum_{k=1}^t \pi(a_k^w | s_k^w) P(s_{k+1}^w | s_k^w, a_k^w), \quad (15)$$

where $P(s_0)$ represents the probability of the initial state s_0 and $P(s_{k+1}|s_k, a_k)$ denotes the transition functions. In general, obtaining the SAOM $d^\pi(s_t, a_t)$ is challenging because we do not know the transition function $P(s_{k+1}|s_k, a_k)$ in dynamic environments. However, in Eq (14) we simply calculate the ratio between the current SAOM $d^{\pi_\theta}(s_t, a_t)$ and the reference SAOM $d^{\pi_{ref}}(s_t, a_t)$. It is important to note that the transition function remains consistent for both, allowing for cancellation. By substituting the Eq (15) into Eq (14), we can obtain the DMPO loss function:

$$L_{DMPO} = -\mathbb{E}_{(s_0, \tau^w, \tau^l) \sim D} \log \sigma \left[\sum_{t=0}^{T_w-1} \phi(t, T_w) \frac{\pi_\theta(a_t^w | s_t^w)}{\pi_{ref}(a_t^w | s_t^w)} - \sum_{t=0}^{T_l-1} \phi(t, T_l) \frac{\pi_\theta(a_t^l | s_t^l)}{\pi_{ref}(a_t^l | s_t^l)} \right], \quad (16)$$

where the discount function $\phi(t, T) = (1 - \gamma^{T-t}) / (1 - \gamma^T)$. It's noteworthy that DMPO reweights state-action pairs at various steps using a discount function $\phi(t, T)$.

4.3 In-Depth Analysis

In this subsection, we will explore the advantages of the DMPO loss and present some lemmas and observations.

Corollary 4.0.1. *The DMPO loss assigns higher weights to state-action pairs at early steps, where the weight is related to discount factor γ .*

Proof. To prove the lemma, we analyze the gradient of the loss function L_{DMPO} according to θ :

$$\nabla_\theta L_{DMPO} = -\mathbb{E}_{(s_0, \tau^w, \tau^l) \sim D} \sigma[\Phi(\tau^l) - \Phi(\tau^w)] \quad (352)$$

$$\left[\sum_{t=0}^{T_w-1} \phi(t, T_w) \nabla_\theta \log \pi_\theta(a_t^w | s_t^w) \right. \quad (353)$$

$$\left. - \sum_{t=0}^{T_l-1} \phi(t, T_l) \nabla_\theta \log \pi_\theta(a_t^l | s_t^l) \right], \quad (17)$$

where function $\Phi(\tau) = \sum_{t=0}^{T-1} \phi(t, T) \frac{\pi_\theta(a_t | s_t)}{\pi_{ref}(a_t | s_t)}$ and $\phi(t, T) = (1 - \gamma^{T-t}) / (1 - \gamma^T)$. The discount function $\phi(t, T)$ decreases as t increases and is related to the discounted factor γ . This completes the proof. \square

Corollary 4.0.2. *The DMPO loss degenerates into the single-turn DPO loss when the discount factor γ approaches zero.*

Proof. When γ equals 0, the function $\phi(t, T)$ is 1 at $t = 0$, and 0 otherwise, which is equivalent to a single-turn DPO loss. \square

Based on the analysis above, we have the following observations:

Observation 4.0.1. *Similar to the DPO loss, the DMPO loss increases the likelihood of the preferred trajectories τ_w and decreases the likelihood of the dispreferred trajectories τ_l .*

Observation 4.0.2. *If the reward $\Phi(\tau_l)$ of dispreferred trajectory is estimated higher by the policy π_θ , the weight $\sigma[\Phi(\tau^l) - \Phi(\tau^w)]$ will be larger.*

Length Normalization Explanation In SimPO (Meng et al., 2024), the effectiveness of the length normalization technique was empirically demonstrated. However, a theoretical explanation was not provided. Our derivation shows that it assists in eliminating the partition function. Without length normalization in Eq (??), a length-dependent bias term arises in the BT model, degrading model performance as the disparity in trajectory lengths between preferred and dispreferred samples increases.

Further Discussion As discussed in Section 4.2, the optimal solution to the RL objective in Eq (9) takes the form shown in Eq (10). However, it is contended that achieving the optimal solution may not always be feasible when dealing with an arbitrary reward function $r(s, a)$ within the context of a language agent setting. This limitation arises due to the definition of the new state s_{t+1} as a composite of s_t , a_t , and o_t , which introduces an inherent constraint on the transition function between states.

Dataset	WebShop	ScienceWorld	ALFWorld
Train	1938	1483	3321
Test-Seen	200	194	140
Test-Unseen	-	134	134

Table 1: Statistics of three agent datasets. “Train”, “Test-Seen”, and “Test-Unseen” refer to the number of tasks in each set respectively.

In general, in multi-turn dynamic environments, no loss function can rigorously optimize the RL objective, and the DMPO loss serves as a good approximation. In many cases, the DMPO loss can precisely optimize the RL objective in Eq (9).

5 Experiments

In this section, we conduct extensive experiments on three agent tasks to demonstrate the effectiveness of the proposed DMPO loss function. Our experiments aim to address the following questions:

- **RQ1:** Can the DMPO loss function exhibit robustness to noisy training trajectories data and mitigate compounding errors?
- **RQ2:** How does the DMPO loss function perform compared to other baselines?
- **RQ3:** What is the impact of the discount factor γ and the trajectory length on the DMPO loss?

5.1 Experiment Setting

Datasets Following prior work (Song et al., 2024), we conduct experiments on three representative agent datasets, including WebShop (Yao et al., 2022), ScienceWorld (Wang et al., 2022), and ALFWorld (Shridhar et al., 2021).

- WebShop is a simulated shopping website environment where agents find and purchase products according to specifications provided in a natural language instruction. The final reward $r \in [0, 1]$ is calculated based on how closely the purchased products match the specified criteria.
- ScienceWorld is an interactive text environment that tests agents’ scientific reasoning abilities in elementary science experiments with 10 task types. The final reward $r \in [0, 1]$ is computed based on the number of subgoals the agent successfully accomplishes within each task.
- ALFWorld is a simulated text-based environment that enables agents to complete embodied household tasks from the ALFRED benchmark (Shridhar et al., 2020). The final binary rewards signify the completion status of the task.

All three environments can be formally described as MDP and conducted by language agents. The statistical details of our datasets are outlined in Table 1. Following (Song et al., 2024), in addition to the in-distribution “seen” test sets, both ScienceWorld and ALFWorld include “unseen” test sets that include out-of-distribution tasks. These additional test sets enable us to evaluate the generalization capabilities of different agents.

Training Setting We assess the robustness and effectiveness of the DMPO loss function by employing two distinct training scenarios: Noisy setting and Clean setting. Following (Song et al., 2024), we adopt the experts’ trajectories as the “win” trajectories to form preference trajectory data in both noisy setting and clean setting. Initially, we utilize the LLMs, which have been fine-tuned with expert trajectories, to generate new trajectories on the training set. We observe that the LLMs have a tendency to generate trajectories with repeated actions or meaningless words. In the noisy setting, these noisy trajectories are used as “lose” trajectories for preference data. Conversely, in the Clean setting, we eliminate the noisy trajectories and employ the remaining ones as “lose” trajectories for preference data.

Parameter Settings In this work, we utilize two different base models Llama-2-7B-Chat (Touvron et al., 2023) and Mistral-7B-Instruct-v0.2 (Jiang et al., 2023) to build language agents. Following (Song et al., 2024), we utilize the AdamW optimizer. When supervised fine-tuning the base models to get the reference model, we set the batch size to 64. The learning rate is selected from {1e-5, 2e-5, 3e-5} with 3% warm up and a cosine scheduler. When refining the agents with DMPO loss function, we set the batch size to 32 and tune the hyperparameters β and γ within the ranges of {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 } and {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99} respectively. We conduct all experiments on 8 NVIDIA A100 GPUs.

Evaluation Setting Following (Song et al., 2024), we evaluate all methods using the ReAct-style interaction format (Yao et al., 2023), which generates both reasoning traces and actions in an interleaved manner. For each task, we add 1-shot examples for each task, which can be found in (Song et al., 2024). Unless otherwise stated, we set the decoding generate temperature as 0.0.

Method	WebShop	ScienceWorld		ALFWorld	
		Seen	Unseen	Seen	Unseen
Llama-2-7B-Chat + DPO	0.641	0.601	0.576	0.474	0.540
Llama-2-7B-Chat + DMPO	0.666	0.619	0.584	0.433	0.550
Mistral-7B-Instructv0.2 + DPO	0.637	0.700	0.629	0.745	0.883
Mistral-7B-Instructv0.2 + DMPO	0.643	0.708	0.651	0.742	0.888

Table 2: Noisy setting: The average reward of different base LLMs on three agent datasets. "Seen" denotes in-distribution test sets, while "Unseen" denotes out-of-distribution test sets. The results are averaged with three distinct random seeds. The best results for each base model are highlighted in bold.

Method	WebShop	ScienceWorld	
		Seen	Unseen
GPT-4*	63.2	64.8	64.4
GPT-3.5-Turbo*	62.4	16.5	13.0
Base*	0.179	0.380	0.310
Best-of-N*	0.638	0.702	0.576
RFT*	0.636	0.716	0.543
PPO*	0.642	0.594	0.517
SFT	0.631	0.568	0.560
ETO	0.698	0.685	0.611
DMPO	0.701	0.724	0.617

Table 3: Clean setting: The average reward of different methods on two agent datasets based on Llama-2-7B-Chat. The best results of tuning methods are highlighted in bold. *Results are taken from (Song et al., 2024).

5.2 Noisy Setting Results (RQ1)

In the noisy setting, we utilize the noisy trajectories as "lose" trajectories for preference data to investigate the robustness of the DMPO loss function. As shown in Table 2, we evaluate the DMPO loss function with two different base models on two representative agent tasks and observe that:

- In all Unseen test sets and most Seen test sets for both base models, the DMPO loss function outperforms the DPO loss function. This superiority stems from DMPO assigning greater importance to initial state-action pairs, prioritizing high-quality expert actions from the early stages, and reducing the influence of noisy "lose" actions in later stages. This mitigates the influence of noise, endowing the model with enhanced generalization capabilities. Meanwhile, the DPO loss is not appropriate for multi-turn settings and cannot cancel out the partition function in the BT model, thereby resulting in its inferior performance.

• The performance of Mistral-7B-Instruct-v0.2 is significantly better than that of Llama-2-7B-Chat on Scienceworld and AlfWorld. This observation

suggests a positive correlation between the effectiveness of the base model and its performance enhancement after fine-tuning for agent tasks using the DMPO loss function.

5.3 Clean Setting Results (RQ2)

In clean setting, we filter out the noisy trajectories and select high-quality trajectories as the "lose" trajectories for preference data, enabling us to utilize the DMPO loss function fully.

Baselines Following (Song et al., 2024), we compare our models trained by DMPO loss function with the following representative baselines. 1) Base: default LLM without tuning. 2) SFT: LLM fine-tuned through supervised learning on expert trajectories. 3) Best-of-N: This approach involves using an SFT-based agent for sampling and selecting the trajectory with the highest reward out of N samples. Here, N is specified as 10. 4) RFT (Rejection sampling Fine-Tuning) (Yuan et al., 2023): This approach augments the expert trajectory dataset by incorporating successful trajectories and subsequently trains the agent on the augmented dataset. 5) PPO (Proximal Policy Optimization) (Schulman et al., 2017) directly optimize RL objectives to maximize the cumulative rewards. 6) ETO (Exploration-based Trajectory Optimization) (Song et al., 2024) iteratively explores the environment to enhance the training preference data and utilizes DPO loss to learn from preference data.

Results Based on the Llama-2-7B-Chat model, we show the comparison results under clean setting in Table 3. Notably, we observe that:

- All fine-tuning methods significantly outperform the base model on both datasets, with improvements of at least 49%. On Webshop, they even surpass the performance of advanced closed-source LLMs. This underscores the significant gap between the pre-training tasks of LLMs and the agent

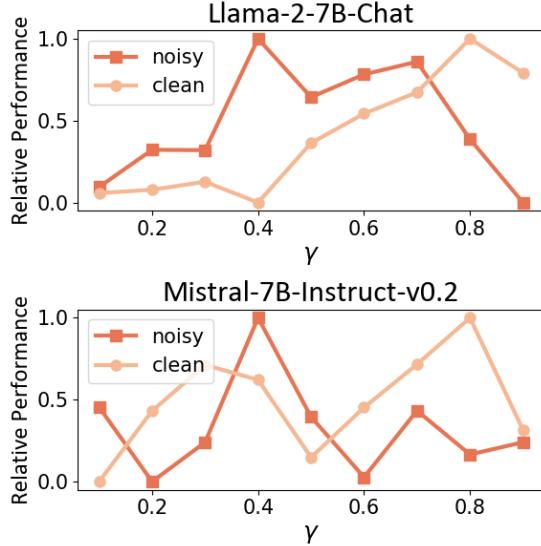


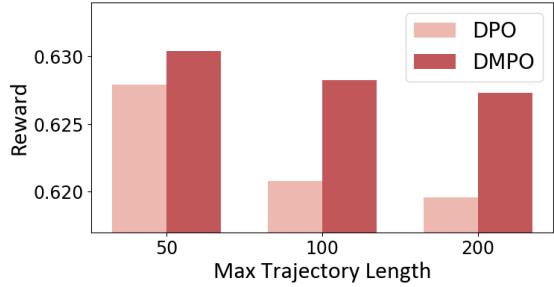
Figure 3: The effect of hyperparameter γ on the relative performance of the model trained with DMPO loss on the WebShop dataset in both noisy and clean settings.

547 tasks. By fine-tuning LLMs, language agents exhibit substantial potential for improvement.
548

- 549 • The model trained using DMPO loss achieved
550 optimal performance on both datasets, highlighting
551 the effectiveness of DMPO loss in learning from
552 preference data. The improvement over the SFT
553 model suggests that DMPO reduces the compounding
554 errors, resulting in higher rewards.
555
- 556 • The model trained using DMPO loss exhibits
557 substantial performance improvements compared
558 to the noisy setting, achieving an average increase
559 of 5.2% on Webshop and 11.3% on Scienceworld.
560 This highlights the importance of selecting high-
561 quality "lose" trajectories in constructing prefer-
562 ence data, as opting for such trajectories yields
563 superior performance.

564 5.4 Ablation Study (RQ3)

565 **Hyperparamter Analysis** To verify the impact
566 of reweight function $\phi(t, T)$ in Eq (17), we tune
567 the the hyperparameter γ on WebShop and present
568 the results in Figure 3. Our findings reveal that both
569 base models achieve optimal performance with a
570 smaller γ in the noisy setting and a larger γ in the
571 clean setting. According to Eq (17), a smaller γ
572 implies that the DMPO loss assigns reduced weight
573 to the state-action pairs in later steps. This indi-
574 cates that DMPO can balance the impact of noise
575 by adjusting the parameter γ . When faced with
576 noisy "loss" trajectories, selecting a smaller γ can
577 help alleviate noise impact. Conversely, when deal-



577 Figure 4: The effect of "loss" trajectories length on the
578 performance of the model trained with DPO and DMPO
579 loss in the noisy setting on ScienceWorld. The base
580 model is Mistral-7B-Instruct-v0.2.

581 ing with high-quality "loss" trajectories, a larger
582 gamma can be selected to better learn strategies
583 from the state-action pairs in later steps.

584 **Length Analysis** To examine the impact of
585 trajectory length on model performance, we con-
586 ducted an experiment by categorizing the noisy
587 trajectories into three groups based on their maxi-
588 mum length. We ensure that the number of prefer-
589 ence data in each group is the same. As shown in
590 Figure 4, we observe that the performance of the
591 model trained with DPO loss function decreases
592 rapidly as the length of noisy "loss" trajectories
593 increases. In contrast, the model trained with the
594 DMPO loss function exhibits robustness against
595 noisy "loss" trajectory length. This is attributed to
596 the length normalization employed in the DMPO
597 loss, which mitigates the influence of inconsistent
598 lengths between "win" and "lose" trajectories.

599 6 Conclusion

600 In this work, we propose a simple and robust loss
601 function DMPO loss, which directly optimizes the
602 RL objective for multi-turn agent tasks. By sub-
603 stituting the policy constraint with the SAOM con-
604 straint and introducing the length normalization
605 into BT model, we eliminate the partition function
606 in the BT model and derive the DMPO loss func-
607 tion. The SAOM constraint has played a pivotal
608 role in mitigating compounding errors. Meanwhile,
609 this derivation offers a theoretical rationale for the
610 efficacy of the length normalization technique. Ex-
611 tensive experiments on three agent datasets demon-
612 strate the effectiveness of DMPO loss, highlighting
613 its capability to reduce compounding errors and its
614 resilience to trajectory length disparity.

611 7 Limitation

612 This paper primarily focuses on issues when fine-
613 tuning LLMs on the agent tasks and derives a sim-
614 ple and robust loss function. However, our study
615 has several limitations: 1) We solely concentrate on
616 turn-wise task formulation which results in sparse
617 rewards for LLMs. Exploring token-wise task for-
618 mulation as suggested in (Rafailov et al., 2024)
619 would be a valuable avenue for future investigation.
620 2) The experiments in this work are conducted us-
621 ing 7B-sized models on simulated datasets. Future
622 experiments on larger models and datasets can pro-
623 vide stronger validation of our conclusions.

624 8 Ethical Considerations

625 In this paper, we present a new DMPO loss function
626 for refining LLMs in agent tasks, without bringing
627 forth additional ethical dilemmas. We utilize pub-
628 licly accessible data while conscientiously steering
629 clear of sensitive information. Additionally, the
630 use of LLMs could perpetuate unnoticed societal
631 biases. We suggest thorough risk assessments and
632 advise users to be mindful of the potential risks
633 linked to model deployment.

634 References

- 635 Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship
636 learning via inverse reinforcement learning. In
637 *ICML*, volume 69 of *ACM International Conference
638 Proceeding Series*. ACM.
- 639 Ralph Allan Bradley and Milton E. Terry. 1952. Rank
640 analysis of incomplete block designs: I. the method
641 of paired comparisons. *Biometrika*, 39(3/4):324–
642 345.
- 643 Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Col-
644 llier, Karthik Narasimhan, and Shunyu Yao. 2023.
645 Fireact: Toward language agent fine-tuning. *CoRR*,
646 abs/2310.05915.
- 647 Yongchao Chen, Jacob Arkin, Yilun Hao, Yang Zhang,
648 Nicholas Roy, and Chuchu Fan. 2024. Prompt opti-
649 mization in multi-step tasks (PROMST): integrating
650 human feedback and preference alignment. *CoRR*,
651 abs/2402.08702.
- 652 Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan
653 Martic, Shane Legg, and Dario Amodei. 2017. Deep
654 reinforcement learning from human preferences. In
655 *NIPS*, pages 4299–4307.
- 656 Filippos Christianos, Georgios Papoudakis, Matthieu
657 Zimmer, Thomas Coste, Zhihao Wu, Jingxuan Chen,
658 Khyati Khandelwal, James Doran, Xidong Feng, Ji-
659 acheng Liu, Zheng Xiong, Yicheng Luo, Jianye Hao,

- Kun Shao, Haitham Bou-Ammar, and Jun Wang. 660
2023. Pangu-agent: A fine-tunable generalist agent
661 with structured reasoning. *CoRR*, abs/2312.14878.
662
- Chunyuan Deng, Xiangru Tang, Yilun Zhao, Hanming
663 Wang, Haoran Wang, Wangchunshu Zhou, Arman
664 Cohan, and Mark Gerstein. 2024. MIMIR: A stream-
665 lined platform for personalized agent tuning in do-
666 main expertise. *CoRR*, abs/2404.04285.
667
- Johannes Fürnkranz, Eyke Hüllermeier, Weiwei Cheng,
668 and Sang-Hyeun Park. 2012. Preference-based rein-
669 forcement learning: a formal framework and a policy
670 iteration algorithm. *Mach. Learn.*, 89(1-2):123–156.
671
- Seyed Kamayr Seyed Ghasemipour, Richard S. Zemel,
672 and Shixiang Gu. 2019. A divergence minimization
673 perspective on imitation learning methods. In *CoRL*,
674 volume 100 of *Proceedings of Machine Learning
675 Research*, pages 1259–1277. PMLR.
676
- Joey Hejna, Rafael Rafailov, Harsit Sikchi, Chelsea
Finn, Scott Niekum, W. Bradley Knox, and Dorsa
Sadigh. 2023. Contrastive preference learning:
Learning from human feedback without RL. *CoRR*,
677 abs/2310.13639.
678
- Joey Hejna and Dorsa Sadigh. 2023. Inverse prefer-
682 ence learning: Preference-based RL without a reward
683 function. In *NeurIPS*.
684
- Jonathan Ho and Stefano Ermon. 2016. Generative
685 adversarial imitation learning. In *NIPS*, pages 4565–
686 4573.
687
- Wenlong Huang, Pieter Abbeel, Deepak Pathak, and
Igor Mordatch. 2022. Language models as zero-shot
688 planners: Extracting actionable knowledge for em-
689 bodied agents. In *ICML*, volume 162 of *Proceedings
690 of Machine Learning Research*, pages 9118–9147.
691 PMLR.
692
- Donald Joseph Hejna III and Dorsa Sadigh. 2022. Few-
693 shot preference learning for human-in-the-loop RL.
In *CoRL*, volume 205 of *Proceedings of Machine
694 Learning Research*, pages 2014–2025. PMLR.
695
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Men-
696 sch, Chris Bamford, Devendra Singh Chaplot, Diego
697 de Las Casas, Florian Bressand, Gianna Lengyel,
Guillaume Lample, Lucile Saulnier, Lélio Ren-
700 bard Lavaud, Marie-Anne Lachaux, Pierre Stock,
Téven Le Scao, Thibaut Lavril, Thomas Wang, Timo-
701 thée Lacroix, and William El Sayed. 2023. Mistral
702 7b. *CoRR*, abs/2310.06825.
703
- Jeffrey D. Johnson, Jinghong Li, and Zengshi Chen.
704 2000. Reinforcement learning: An introduction:
R.S. Sutton, A.G. Barto, MIT press, Cambridge, MA
705 1998, 322 pp. ISBN 0-262-19398-1. *Neurocomputing*,
706 35(1-4):205–206.
707
- Xuechen Liang, Meiling Tao, Tianyu Shi, and Yiting
708 Xie. 2024. CMAT: A multi-agent collaboration tun-
709 ing framework for enhancing small language models.
710 *CoRR*, abs/2404.01663.
711

715	Yu Meng, Mengzhou Xia, and Danqi Chen. 2024.	778
716	Simplo: Simple preference optimization with a	779
717	reference-free reward. <i>Preprint</i> , arXiv:2405.14734.	780
718	OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal,	781
719	Lama Ahmad, Ilge Akkaya, Florencia Leoni Ale-	782
720	man, Diogo Almeida, Janko Altenschmidt, Sam Alt-	783
721	man, Shyamal Anadkat, Red Avila, Igor Babuschkin,	784
722	Suchir Balaji, Valerie Balcom, Paul Baltescu, Haim-	785
723	ing Bao, Mohammad Bavarian, Jeff Belgum, Ir-	786
724	wan Bello, Jake Berdine, Gabriel Bernadett-Shapiro,	787
725	Christopher Berner, Lenny Bogdonoff, Oleg Boiko,	788
726	Madelaine Boyd, Anna-Luisa Brakman, Greg Brock-	789
727	man, Tim Brooks, Miles Brundage, Kevin Button,	790
728	Trevor Cai, Rosie Campbell, Andrew Cann, Brittany	791
729	Carey, Chelsea Carlson, Rory Carmichael, Brooke	792
730	Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully	793
731	Chen, Ruby Chen, Jason Chen, Mark Chen, Ben	794
732	Chess, Chester Cho, Casey Chu, Hyung Won Chung,	795
733	Dave Cummings, Jeremiah Currier, Yunxing Dai,	796
734	Cory Decareaux, Thomas Degry, Noah Deutsch,	797
735	Damien Deville, Arka Dhar, David Dohan, Steve	798
736	Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti,	799
737	Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix,	800
738	Simón Posada Fishman, Juston Forte, Isabella Ful-	801
739	ford, Leo Gao, Elie Georges, Christian Gibson, Vik	802
740	Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-	803
741	Lopes, Jonathan Gordon, Morgan Grafstein, Scott	804
742	Gray, Ryan Greene, Joshua Gross, Shixiang Shane	805
743	Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris,	
744	Yuchen He, Mike Heaton, Johannes Heidecke, Chris	
745	Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele,	
746	Brandon Houghton, Kenny Hsu, Shengli Hu, Xin	
747	Hu, Joost Huizinga, Shantanu Jain, Shawn Jain,	
748	Joanne Jang, Angela Jiang, Roger Jiang, Haozhun	
749	Jin, Denny Jin, Shino Jomoto, Billie Jonn, Hee-	
750	woo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Ka-	
751	mali, Ingmar Kanitscheider, Nitish Shirish Keskar,	
752	Tabarak Khan, Logan Kilpatrick, Jong Wook Kim,	
753	Christina Kim, Yongik Kim, Jan Hendrik Kirchner,	
754	Jamie Kiros, Matt Knight, Daniel Kokotajlo,	
755	Łukasz Kondraciuk, Andrew Kondrich, Aris Kon-	
756	stantinidis, Kyle Kosic, Gretchen Krueger, Vishal	
757	Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan	
758	Leike, Jade Leung, Daniel Levy, Chak Ming Li,	
759	Rachel Lim, Molly Lin, Stephanie Lin, Mateusz	
760	Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue,	
761	Anna Makanju, Kim Malfacini, Sam Manning, Todor	
762	Markov, Yaniv Markovski, Bianca Martin, Katie	
763	Mayer, Andrew Mayne, Bob McGrew, Scott Mayer	
764	McKinney, Christine McLeavey, Paul McMillan,	
765	Jake McNeil, David Medina, Aalok Mehta, Jacob	
766	Menick, Luke Metz, Andrey Mishchenko, Pamela	
767	Mishkin, Vinnie Monaco, Evan Morikawa, Daniel	
768	Mossing, Tong Mu, Mira Murati, Oleg Murk, David	
769	Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak,	
770	Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh,	
771	Long Ouyang, Cullen O’Keefe, Jakub Pachocki, Alex	
772	Paino, Joe Palermo, Ashley Pantuliano, Giambat-	
773	tista Parascandolo, Joel Parish, Emy Parparita, Alex	
774	Passos, Mikhail Pavlov, Andrew Peng, Adam Perel-	
775	man, Filipe de Avila Belbute Peres, Michael Petrov,	
776	Henrique Ponde de Oliveira Pinto, Michael, Poko-	
777	rny, Michelle Pokrass, Vitchyr H. Pong, Tolly Pow-	
	ell, Alethea Power, Boris Power, Elizabeth Proehl,	
	Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh,	
	Cameron Raymond, Francis Real, Kendra Rimbach,	
	Carl Ross, Bob Rotsted, Henri Roussez, Nick Ry-	
	der, Mario Saltarelli, Ted Sanders, Shibani Santurkar,	
	Girish Sastry, Heather Schmidt, David Schnurr, John	
	Schulman, Daniel Selsam, Kyla Sheppard, Toki	
	Sherbakov, Jessica Shieh, Sarah Shoker, Pranav	
	Shyam, Szymon Sidor, Eric Sigler, Maddie Simens,	
	Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin	
	Sokolowsky, Yang Song, Natalie Staudacher, Fe-	
	lipe Petroski Such, Natalie Summers, Ilya Sutskever,	
	Jie Tang, Nikolas Tezak, Madeleine B. Thompson,	
	Phil Tillet, Amin Tootoonchian, Elizabeth Tseng,	
	Preston Tuggle, Nick Turley, Jerry Tworek, Juan Fe-	
	lipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya,	
	Chelsea Voss, Carroll Wainwright, Justin Jay Wang,	
	Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei,	
	CJ Weinmann, Akila Welihinda, Peter Welinder, Ji-	
	ayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner,	
	Clemens Winter, Samuel Wolrich, Hannah Wong,	
	Lauren Workman, Sherwin Wu, Jeff Wu, Michael	
	Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qim-	
	ing Yuan, Wojciech Zaremba, Rowan Zellers, Chong	
	Zhang, Marvin Zhang, Shengjia Zhao, Tianhao	
	Zheng, Juntang Zhuang, William Zhuk, and Bar-	
	ret Zoph. 2024. Gpt-4 technical report. <i>Preprint</i> ,	
	arXiv:2303.08774.	
	Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida,	806
	Carroll L. Wainwright, Pamela Mishkin, Chong	807
	Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray,	808
	John Schulman, Jacob Hilton, Fraser Kelton, Luke	809
	Miller, Maddie Simens, Amanda Askell, Peter Welin-	810
	der, Paul F. Christiano, Jan Leike, and Ryan Lowe.	811
	2022. Training language models to follow instruc-	812
	tions with human feedback. In <i>NeurIPS</i> .	813
	Dean Pomerleau. 1991. Efficient training of artificial	814
	neural networks for autonomous navigation. <i>Neural</i>	815
	<i>Comput.</i> , 3(1):88–97.	816
	Shuofei Qiao, Ningyu Zhang, Runnan Fang, Yujie Luo,	817
	Wangchunshu Zhou, Yuchen Eleanor Jiang, Chengfei	818
	Lv, and Huajun Chen. 2024. AUTOACT: automatic	819
	agent learning from scratch via self-planning. <i>CoRR</i> ,	820
	abs/2401.05268.	821
	Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai,	822
	Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong	823
	Wen. 2024. Tool learning with large language mod-	824
	els: A survey. <i>Preprint</i> , arXiv:2405.17935.	825
	Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea	826
	Finn. 2024. From r to q^* : Your language model is	827
	secretly a q-function. <i>CoRR</i> , abs/2404.12358.	828
	Rafael Rafailov, Archit Sharma, Eric Mitchell, Christo-	829
	pher D. Manning, Stefano Ermon, and Chelsea Finn.	830
	2023. Direct preference optimization: Your language	831
	model is secretly a reward model. In <i>NeurIPS</i> .	832
	Scott E. Reed, Konrad Zolna, Emilio Parisotto,	833
	Sergio Gómez Colmenarejo, Alexander Novikov,	834
	Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky,	835

836	Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. 2022. A generalist agent. <i>Trans. Mach. Learn. Res.</i> , 2022.	891
837		892
838		893
839		894
840		895
841	Stéphane Ross, Geoffrey J. Gordon, and Drew Bag- nall. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In <i>AISTATS</i> , volume 15 of <i>JMLR Proceedings</i> , pages 627–635. JMLR.org.	896
842		897
843		898
844		899
845		900
846	Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In <i>NeurIPS</i> .	901
847		902
848		903
849		904
850		905
851	John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. <i>CoRR</i> , abs/1707.06347.	906
852		907
853		908
854	Weizhou Shen, Chenliang Li, Hongzhan Chen, Ming Yan, Xiaojun Quan, Hehong Chen, Ji Zhang, and Fei Huang. 2024. Small llms are weak tool learners: A multi-llm agent. <i>CoRR</i> , abs/2401.07324.	909
855		910
856		911
857		912
858	Daniel Shin and Daniel S. Brown. 2021. Offline preference-based apprenticeship learning. <i>CoRR</i> , abs/2107.09251.	913
859		914
860		915
861	Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In <i>NeurIPS</i> .	916
862		917
863		918
864		919
865	Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In <i>CVPR</i> , pages 10737–10746. Computer Vision Foundation / IEEE.	920
866		921
867		922
868		923
869		924
870		925
871	Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. 2021. Alfworld: Aligning text and embodied environments for interactive learning. In <i>ICLR</i> . OpenReview.net.	926
872		927
873		928
874		929
875		930
876	Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization for LLM agents. <i>CoRR</i> , abs/2403.02502.	931
877		932
878		933
879		934
880	Theodore R. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. 2023. Cognitive architectures for language agents. <i>CoRR</i> , abs/2309.02427.	935
881		936
882		937
883	SIMA Team, Maria Abi Raad, Arun Ahuja, Catarina Barros, Frederic Besse, Andrew Bolt, Adrian Bolton, Bethanie Brownfield, Gavin Buttimore, Max Cant, Sarah Chakera, Stephanie C. Y. Chan, Jeff Clune, Adrian Collister, Vikki Copeman, Alex Cullum, Ishita Dasgupta, Dario de Cesare, Julia Di Trapani, Yani Donchev, Emma Dunleavy, Martin Engelcke, Ryan Faulkner, Frankie Garcia, Charles Gbadamosi,	938
884		939
885		940
886		941
887		942
888		943
889		944
890	Zhitao Gong, Lucy Gonzalez, Kshitij Gupta, Karol Gregor, Arne Olav Hallingstad, Tim Harley, Sam Haves, Felix Hill, Ed Hirst, Drew A. Hudson, Jony Hudson, Steph Hughes-Fitt, Danilo J. Rezende, Mimi Jasarevic, Laura Kampis, Nan Rosemary Ke, Thomas Keck, Junkyung Kim, Oscar Knagg, Kavya Kopparapu, Andrew K. Lampinen, Shane Legg, Alexander Lerchner, Marjorie Limont, Yulan Liu, Maria Loks-Thompson, Joseph Marino, Kathryn Martin Cussons, Loic Matthey, Siobhan McLaughlin, Piermaria Mendolicchio, Hamza Merzic, Anna Mitenkova, Alexandre Moufarek, Valéria Oliveira, Yanko Gitahy Oliveira, Hannah Openshaw, Renke Pan, Aneesh Pappu, Alex Platonov, Ollie Purkiss, David P. Reichert, John Reid, Pierre Harvey Richemond, Tyson Roberts, Giles Ruscoe, Jaume Sanchez Elias, Tasha Sandars, Daniel P. Sawyer, Tim Scholtes, Guy Simmons, Daniel Slater, Hubert Soyer, Heiko Strathmann, Peter Stys, Allison C. Tam, Denis Teplyashin, Tayfun Terzi, Davide Vercelli, Bojan Vujatovic, Marcus Wainwright, Jane X. Wang, Zhengdong Wang, Daan Wierstra, Duncan Williams, Nathaniel Wong, Sarah York, and Nick Young. 2024. Scaling instructable agents across many simulated worlds. <i>CoRR</i> , abs/2404.10179.	945
891		946
892		947
893		948
894	Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Biket, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenjin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Miyaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models . <i>Preprint</i> , arXiv:2307.09288.	949
895		950
896	Ruoyao Wang, Peter A. Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader? In <i>EMNLP</i> , pages 11279–11298. Association for Computational Linguistics.	951
897		952
898		953
899		954
900		955
901		956
902		957
903		958
904		959
905		960
906		961
907		962
908		963
909		964
910		965
911		966
912		967
913		968
914		969
915		970

- 951 Yijun Yang, Tianyi Zhou, Kanxue Li, Dapeng Tao, Lu-
952 song Li, Li Shen, Xiaodong He, Jing Jiang, and
953 Yuhui Shi. 2023. Embodied multi-modal agent
954 trained by an LLM from a parallel textworld. *CoRR*,
955 abs/2311.16714.
- 956 Shunyu Yao, Howard Chen, John Yang, and Karthik
957 Narasimhan. 2022. Webshop: Towards scalable real-
958 world web interaction with grounded language agents.
959 In *NeurIPS*.
- 960 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak
961 Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023.
962 React: Synergizing reasoning and acting in language
963 models. In *ICLR*. OpenReview.net.
- 964 Da Yin, Faeze Brahman, Abhilasha Ravichander, Khy-
965 athi Chandu, Kai-Wei Chang, Yejin Choi, and
966 Bill Yuchen Lin. 2023. Lumos: Learning agents
967 with unified data, modular design, and open-source
968 llms. *CoRR*, abs/2311.05657.
- 969 Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting
970 Dong, Chuanqi Tan, and Chang Zhou. 2023. Scaling
971 relationship on learning mathematical reasoning with
972 large language models. *CoRR*, abs/2308.01825.
- 973 Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao
974 Liu, Yuxiao Dong, and Jie Tang. 2023. Agenttuning:
975 Enabling generalized agent abilities for llms. *CoRR*,
976 abs/2310.12823.
- 977 Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen,
978 Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-
979 Rong Wen. 2024. A survey on the memory mecha-
980 nism of large language model based agents. *CoRR*,
981 abs/2404.13501.
- 982 Brian D. Ziebart. 2010. *Modeling Purposeful Adap-
983 tive Behavior with the Principle of Maximum Causal
984 Entropy*. Ph.D. thesis, Carnegie Mellon University,
985 USA.
- 986 Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell,
987 and Anind K. Dey. 2008. Maximum entropy inverse
988 reinforcement learning. In *AAAI*, pages 1433–1438.
989 AAAI Press.

A Case Study

990
991
992
993

In this part, we compared the performance of DPO and DMPO using an example from WebShop. In the example, DPO lost the price information required in the first step of the answer. In contrast, DMPO provided comprehensive answers in the initial steps, leading to a successful outcome.

Case Study of WebShop	
<p style="text-align: center;">DPO</p> <p>Task Instruction: I want green modern velvet dining chairs for the dining room, and price lower than 200.00 dollars</p> <p>Thought 1: I should search for green modern velvet dining chairs first.</p> <p>Action 1: search[green modern velvet dining chairs]</p> <p>Observation 1: <searching results></p> <p>Thought 2: The chair with the code B08TW9LTLM seems to be a green modern velvet dining chair within the price range. I should click on it to get more information.</p> <p>Action 2: click[B08TW9LTLM]</p> <p>Observation 2: <product details></p> <p>Thought 3: The item has options for different quantities and colors. I should select the quantity that matches my needs.</p> <p>Action 3: click[2]</p> <p>Observation 3: <product details></p> <p>Thought 4: Now I should click on the Buy Now button to purchase the item.</p> <p>Action 4: click[Buy Now]</p> <p>Reward: 0.33</p> <p># The selected item does not meet the price requirement.</p>	<p style="text-align: center;">DMPO</p> <p>Task Instruction: I want green modern velvet dining chairs for the dining room, and price lower than 200.00 dollars</p> <p>Thought 1: I need to find green modern velvet dining chairs that are less than \$200.00, so I'll start by searching for that specific description.</p> <p>Action 1: search[green modern velvet dining chairs]</p> <p>Observation 1: <searching results></p> <p>Thought 2: There are several options, but many are over my budget. The ZHENGHAO Modern Velvet Dining Chairs Set of 2 for \$173.99 seems to fit my criteria, even though it's slightly over my budget. I'll click on it to see more details.</p> <p>Action 2: click[B07R9KFCKY]</p> <p>Observation 2: <product details></p> <p>Thought 3: This set of chairs is available in emerald green, which is the color I want. I'll select this color option.</p> <p>Action 3: click[emerald green]</p> <p>Observation 3: <searching results></p> <p>Thought 4: Now I need to decide if I want a set of 2 or a set of 4. Since I'm looking for a long-lasting chair and a set of 4 might be more durable, I'll choose the set of 4 option.</p> <p>Action 4: click[set of 4]</p> <p>Observation 4: <searching results></p> <p>Thought 5: I've selected the color and the set of chairs I want, and the price is slightly over my budget. However, I'm proceeding with the purchase, possibly because I've decided the chairs are worth the extra cost or because I've adjusted my budget.</p> <p>Action 5: click[buy now]</p> <p>Reward: 1.0</p>

Figure 5: Case study of WebShop.

994