Offline Reinforcement Learning for LLM Multi-step Reasoning

Huaijie Wang[♢*], **Shibo Hao**[♣*], **Hanze Dong**[♡], **Shenao Zhang**[♠]
Yilin Bao[♣], **Ziran Yang**[♣], **Yi Wu**[♢†]
[♣]UC San Diego, [♢]Tsinghua University,
[♡]Salesforce Research, [♠]Northwestern University

Abstract

Improving the multi-step reasoning ability of large language models (LLMs) with offline reinforcement learning (RL) is essential for quickly adapting them to complex tasks. While Direct Preference Optimization (DPO) has shown promise in aligning LLMs with human preferences, it is less suitable for multi-step reasoning tasks because (1) DPO relies on paired preference data, which is not readily available for multi-step reasoning tasks, and (2) it treats all tokens uniformly, making it ineffective for credit assignment in multi-step reasoning tasks, which often come with sparse reward. In this work, we propose OREO (Offline REasoning Optimization), an offline RL method for enhancing LLM multi-step reasoning. Building on insights from previous works of maximum entropy reinforcement learning, it jointly learns a policy model and value function by optimizing the soft Bellman Equation. We show in principle that it reduces the need to collect pairwise data and enables better credit assignment. Empirically, OREO surpasses existing offline learning methods on multi-step reasoning benchmarks, including mathematical reasoning tasks (GSM8K, MATH), and embodied agent control (ALFWorld). The approach can be extended to a multi-iteration framework when additional resources are available. Furthermore, the learned value function can be leveraged to guide the tree search for free, which can further boost the performance during test time.

1 Introduction

Large Language Models (LLMs) are increasingly applied to complex tasks requiring multi-step reasoning, such as mathematical problem solving (Uesato et al., 2022; Shao et al., 2024; Hendrycks et al., 2021), embodied agent control (Wang et al., 2023; Huang et al., 2022; Shridhar et al., 2020; Xiang et al., 2024), and web navigation (Deng et al., 2024; Zhou et al., 2023; Koh et al., 2024). Enhancing LLM reasoning with reinforcement learning (RL) has gained significant interest, as it offers the potential for self-improvement and learning without relying on human-labeled trajectories. However, many popular RL algorithms require costly online data collection, either by generating language on-the-fly or interacting with an environment. For instance, tuning LLMs with Proximal Policy Optimization (PPO, Schulman et al., 2017) is often prohibitively expensive for most users, which limits practical applications (Hu et al., 2023).

In contrast, offline RL methods, such as Direct Preference Optimization (DPO, Rafailov et al., 2024b), provide a more practical approach for aligning LLMs with human preferences. These methods enable practitioners to tune models using pre-existing datasets, eliminating the need for live interaction or data generation. However, attempts to enhance LLMs' multi-step reasoning abilities with DPO are not always successful. For example, recent works find that DPO may perform close to or even worse than simpler methods like SFT (Yuan et al., 2024; Chen et al., 2024b). Additionally, DPO requires pairwise preference data. In multi-step reasoning tasks, however, data normally consists of independent trajectories with sparse rewards indicating success or failure. A common alternative is to extract correct trajectories from offline datasets and use them for supervised fine-tuning (Zelikman et al., 2022; Aksitov et al., 2023; Dong et al., 2023; Paulus et al., 2024). While this approach is simple and often effective, it fails to fully exploit the offline dataset's potential—particularly the opportunity to learn from failure experience and enhance model robustness (Kumar et al., 2022).

In this paper, we introduce OREO (Offline REasoning Optimization), an offline RL algorithm designed to enhance LLMs' multi-step reasoning capabilities. Building on insights from the exten-

[*]Equal contribution. † corresponding author

Findings of the Association for Computational Linguistics: ACL 2025, pages 8881–8893
July 27 - August 1, 2025

sive literature on maximum entropy RL (Ziebart, 2010; Nachum et al., 2017; Haarnoja et al., 2017), especially Path Consistency Learning (Nachum et al., 2017), OREO jointly learns a policy model and a value function by optimizing the soft Bellman Equation. OREO can leverage unpaired data with only sparse rewards and enables finer-grained credit assignment, which is especially critical as the correctness of reasoning trajectories often depends on a few key tokens. Additionally, OREO can be extended into an iterative framework when online exploration is allowed. The trained value function is also directly available to guide the step-level beam search at inference time, further boosting performance.

We demonstrate the effectiveness of our approach on both math reasoning (GSM8K, MATH) and embodied agent control (ALFWorld) tasks. It consistently outperforms baseline methods, including rejection sampling, DPO, and KTO, across different model sizes. Notably, we train a 1.5B model to achieve 52.5% accuracy on the MATH dataset using only the original training set. Moreover, iterative OREO steadily improves model performance with additional training rounds, whereas baseline methods like rejection sampling exhibit signs of saturation. The value function learned by our method proves highly effective in guiding beam search for math reasoning tasks or selecting the best-of-K actions in embodied agent control. This results in up to a 17.9% relative improvement over greedy decoding on the MATH dataset.

2 Related Work

2.1 Reinforcement Learning for LLM

Reinforcement Learning (RL) has become a standard approach in the post-training stage of LLMs. A widely adopted method, known as reinforcement learning from human feedback (RLHF) (Ziegler et al., 2019; Ouyang et al., 2022), is designed to align LLM responses more closely with human preferences. Traditional RL methods, such as Proximal Policy Optimization (PPO) (Schulman et al., 2017), have been extensively used in LLM post-training (Achiam et al., 2023; Team et al., 2023; Dubey et al., 2024). Alternative approaches, such as rejection-sampling-based methods (Dong et al., 2023; Gulcehre et al., 2023; Zelikman et al., 2022; Hoffman et al., 2024), preference-based reinforcement learning (RL) (Rafailov et al., 2024b; Xiong et al., 2024a; Ethayarajh et al., 2024), and REINFORCE-like RL (Williams, 1992; Shao et al., 2024; Li et al., 2023; Ahmadian et al., 2024), have recently gained traction in the LLM literature.

Maximum-entropy RL (Ziebart, 2010; Haarnoja et al., 2017) aims to maximize the weighted sum of the accumulated reward and the policy entropy. Notable algorithms such as path-consistency learning (PCL) (Nachum et al., 2017) and soft actor-critic (SAC) (Haarnoja et al., 2018) effectively utilize this framework. Recent works (Guo et al., 2021; Richemond et al., 2024; Liu et al., 2024a) revealed a strong connection between maximum-entropy RL and the RLHF objective, indicating a promising direction to fine-tune LLMs with soft Q-learning-based algorithms. DRO (Richemond et al., 2024) proposes a similar approach as ours in the soft Q-learning framework. They consider the bandit setting where the entire model response is treated as a single action. In contrast, OREO leverages a token-level value function, enabling finer-grained credit assignment, which we empirically find beneficial for multi-step reasoning tasks. Concurrent with our work, Liu et al. (2024a) leverages the SAC framework and derives a similar algorithm for LLM multi-step reasoning. However, our work includes more analysis against DPO, evaluates the method on more tasks, and conducts a wider variety of experiments, including test-time scaling with value-based search.

2.2 LLM Reasoning

As an emergent ability of model scale, it is shown that LLMs are able to generate intermediate reasoning steps to solve complex problems, known as "scratchpad" (Nye et al., 2021) or chain-of-thoughts (Wei et al., 2022; Kojima et al., 2022). Recent efforts have enhanced LLM reasoning through supervised fine-tuning (Yue et al., 2023, 2024; Yu et al., 2023; Luo et al., 2023). When human-annotated reasoning trajectories are unavailable, rejection-sampling-based methods have proven effective. Among them, Self-Taught Reasoner (STaR) (Zelikman et al., 2022) generates rationales and fine-tunes on those leading to correct answers. Singh et al. (2023) further proposes an iterative approach based on expectation maximization. Recently, the application of RL algorithms to improve LLM reasoning has gained increasing interest (Aksitov et al., 2023; Gou et al., 2023; Dong et al., 2024; Havrilla et al., 2024; Shao et al., 2024; Zhao et al., 2024), but the direct application of DPO has been less successful (Yuan et al., 2024; Chen et al., 2024b), and

people have to specifically collect pairwise preference data for it (Chen et al., 2024a; Song et al., 2024). Our method addresses these limitations of DPO in reasoning with a principled solution. Another line of work aims to train a *Process Reward Model (PRM)*, to provide finer-grained feedback on RL. It is typically trained with Monte-carlo rollout (Wang et al., 2024a,b; Luo et al., 2024), which can be viewed as a special case of the value function learned through our method. We show that our value function enables test-time scaling (Snell et al., 2024; Wu et al., 2024; Brown et al., 2024; Yao et al., 2024; Hao et al., 2024; Liu et al., 2024b) to further boost the reasoning performance through tree search.

3 Preliminaries

3.1 Markov Decision Process for LLM Reasoning

We define the Markov Decision Process (MDP) for LLM reasoning. At each time step, the action a_t is to generate a new token. The state is represented as a token sequence. For reasoning tasks that don't involve interaction with environment, s_t records the context for LLMs, i.e., $\mathbf{s}_t = (x_0, \ldots, x_L, y_0, \ldots, y_{t-1})$, where $(x_0, \ldots, x_L)$ is the input prompt and $(y_0, \ldots, y_{t-1})$ is the sequence of generated tokens up to step $t-1$. The transition function f for these tasks deterministically updates the state as $\mathbf{s_{t+1}} = f(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{s}_t \mid \mathbf{a}_t$, where $\mid$ is concatenation. For those tasks requiring interacting with an external environment, like embodied agent control, the state and transition function is slightly different: if $\mathbf{a}_t$ is the final token of the agent's response (e.g., *"go to desk 1"*), then $\mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{s}_t \mid \mathbf{a}_t \mid$ *next observation*.

The reward function $r(\mathbf{s}_t, \mathbf{a}_t)$ is generally defined for every state-action pair to provide feedback throughout the generation process. However, in this work, we specifically focus on the challenging case where the reward is non-zero only at the terminal step T, reflecting the correctness of the reasoning chain, or whether the task is successfully accomplished.

Following the standard setup in Reinforcement Learning with Human Feedback (RLHF) (Ouyang et al., 2022; Rafailov et al., 2024b), a KL-regularization term is introduced to encourage the learned policy to remain close to a reference policy while optimizing for rewards. Therefore, the optimal policy π_θ can be described as follows:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{(\mathbf{s}_0,\ldots,\mathbf{s}_T)\sim\rho_\pi} \sum_{t=0}^{T} \Big[r\left(\mathbf{s}_t, \mathbf{a}_t\right) - \beta \log \frac{\pi\left(\mathbf{a}_t \mid \mathbf{s}_t\right)}{\pi_{\text{ref}}(\mathbf{a}_t \mid \mathbf{s}_t)} \Big], \tag{1}$$

where π_{ref} is the reference policy, ρ_π is the state-action trajectory distribution generated by following policy π, and β controls the strength of the regularization. Typically, π_{ref} is a pre-trained LLM followed by supervised fine-tuning. The discount factor γ is normally omitted in the RLHF setting.

3.2 Soft Bellman Equation

Entropy-regularized reinforcement learning (RL) (Ziebart, 2010; Nachum et al., 2017; Haarnoja et al., 2017) augments the standard reward maximization objective with an entropy term to encourage exploration and improve the robustness of the learned policy. This formulation has a strong connection to entropy-regularized RL, as the Kullback-Leibler (KL) divergence between two distributions can be decomposed into a cross-entropy term and an entropy term, i.e., $D_{\text{KL}}(\pi(\cdot|s)\|\pi_{\text{ref}}(\cdot|s)) = \mathbb{E}_\pi[-\log \pi_{\text{ref}}(a|s)] - \mathbb{E}_\pi[-\log \pi(a|s)]$.

Adapting from the well-established theory in entropy-regularized RL to our setting, we first define the value function V^π of a policy, which quantifies the expected KL-regularized reward of a policy π from any given state:

$$V^\pi\left(\mathbf{s}_t\right) = \mathbb{E}_{\left(\mathbf{a}_t, \mathbf{s}_{t+1}, \ldots\right)\sim\rho_\pi} \sum_{l=0}^{T-t} \Big[r\left(\mathbf{s}_{t+l}, \mathbf{a}_{t+l}\right) - \beta \log \frac{\pi(\mathbf{a}_{t+l} \mid \mathbf{s}_{t+l})}{\pi_{\text{ref}}(\mathbf{a}_{t+l} \mid \mathbf{s}_{t+l})} \Big] \tag{2}$$

Compared to the value function in standard RL, which only includes expected rewards, the above definition incorporates an additional term $\beta \log \frac{\pi(\mathbf{a}_{t+l}|\mathbf{s}_{t+l})}{\pi_{\text{ref}}(\mathbf{a}_{t+l}|\mathbf{s}_{t+l})}$ corresponding to the KL regularization.

Theorem 1 *The optimal policy and its value function satisfy the* ***soft Bellman Equation****:*

$$V^*(\mathbf{s}_t) - V^*(\mathbf{s}_{t+1}) = r(\mathbf{s}_t, \mathbf{a}_t) - \beta \log \frac{\pi^*(\mathbf{a_t} \mid \mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a_t} \mid \mathbf{s}_t)} \tag{3}$$

where $\mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{a}_t)$.

Building on Nachum et al. (2017); Haarnoja et al. (2017), we extend their theorem with a lightweight derivation tailored to our setting, with the proof provided in Appendix B.

This equation characterizes the relationship between the optimal policy and its value function, providing a theoretical basis for our proposed method. When $\beta = 0$, the equation reduces to the Bellman equation in standard RL. Importantly, when the soft Bellman Equation is satisfied everywhere, the policy and the value function are guaranteed to be the optimal ones:

Theorem 2 *If a policy $\pi(\mathbf{a} \mid \mathbf{s})$ and state value function $V(\mathbf{s})$ satisfy the consistency property (3) for all states $\mathbf{s}$ and actions $\mathbf{a}$ (where $\mathbf{s}' = f(\mathbf{s}, \mathbf{a})$), then $\pi = \pi^*$ and $V = V^*$.*

Similarly, the proof is a simple extension to Nachum et al. (2017). Based on Theorem 2, our proposed method OREO aims to learn both a policy model π_θ and a value model V_ϕ towards the optimal policy and value function. This is achieved by minimizing the inconsistency of Soft Bellman Consistency. A more formal description of our method is presented in Section 4.

3.3 Connection to DPO

In this section, we introduce how DPO can be derived from the formulation above with two additional assumptions. This enables us to understand the limitation of DPO on LLM reasoning from the principle, and motivates us to propose the new method. Rafailov et al. (2024a) present a related derivation to analyze the properties of DPO.

First, DPO relaxes the requirements of soft Bellman Equation by telescoping consequent time steps:

$$\sum_{t=0}^{T-1} r(\mathbf{s}_t, \mathbf{a}_t) = V^*(\mathbf{s}_0) + \sum_{t=0}^{T-1} \beta \log \frac{\pi^*(\mathbf{a}_t \mid \mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t \mid \mathbf{s}_t)} \quad (4)$$

It then introduces the Bradley-Terry preference model (Bradley and Terry, 1952), which assumes that the probability of one response being preferred over another is determined by the normalized relative exponential rewards of the responses. Specifically:

$$p^*\left(\tau^w \succeq \tau^l\right) = \frac{\exp\left(r\left(\mathbf{s}_T^w, \mathbf{a}_T^w\right)\right)}{\exp\left(r\left(\mathbf{s}_T^w, \mathbf{a}_T^w\right)\right) + \exp\left(r\left(\mathbf{s}_T^l, \mathbf{a}_T^l\right)\right)}. \quad (5)$$

By maximizing the log-likelihood that a winning response is preferred over a losing response with a preference dataset $D = \{(\tau^w, \tau^l)\}$, the loss function of DPO can be derived:

$$\mathcal{L} = -\mathbb{E}_{(\tau_w, \tau_l) \sim \mathcal{D}} \left[\log \sigma \left(\left(\sum_{t=0}^{T-1} \beta \log \frac{\pi^*(\mathbf{a}_t^w \mid \mathbf{s}_t^w)}{\pi_{\text{ref}}(\mathbf{a}_t^w \mid \mathbf{s}_t^w)} \right) - \left(\sum_{t=0}^{T-1} \beta \log \frac{\pi^*(\mathbf{a}_t^l \mid \mathbf{s}_t^l)}{\pi_{\text{ref}}(\mathbf{a}_t^l \mid \mathbf{s}_t^l)} \right) \right) \right] \quad (6)$$

The additional assumptions of DPO introduce two challenges for multi-step reasoning problems: (1) **Unnecessary pairwise data collection**: While the BT model is reasonable for a general dialogue system where the reward can only be implicitly inferred from human preference, it's unnecessary for multi-step reasoning tasks where a ground-truth reward exists. To apply DPO on these tasks, previous work has to collect pairwise data on reasoning tasks (Song et al., 2024; Yuan et al., 2024; Xiong et al., 2024b), which is an inefficient usage of offline data. (2) **No credit assignment**: Relaxing the soft Bellman Equation from every time step to the entire trajectory loses the granularity of credit assignment, which is especially critical in multi-step reasoning tasks where correctness often depends on a few key tokens.

4 OREO: Offline Reasoning Optimization

Based on the theorems presented in Section 3, we present the detailed formulation of our method OREO. We further introduce two objective function variants, an iterative extension of our approach, and a test-time search strategy leveraging the value function.

4.1 Learning Objetive

We adopt a similar method as PCL (Nachum et al., 2017) to fine-tune the LLM. Inspired by Theorem 2, we optimize the policy by enforcing the soft Bellman Equation property given in Eq. 3. In our setting where the reward signal is sparse, we aim to enforce the telescoped version of Eq. 3, namely

$$V^*(\mathbf{s}_t) = R_t - \beta \sum_{i \geq t} \log \frac{\pi^*(\mathbf{a}_i|\mathbf{s}_i)}{\pi_{\text{ref}}(\mathbf{a}_i|\mathbf{s}_i)}, \quad (7)$$

where $R_t = \sum_{i \geq t} r(\mathbf{s}_i, \mathbf{a}_i)$. Note that DPO leverages Eq. 4, which is a special case of Eq. 7 with $t = 0$. We train a separate value network V_ϕ together with the policy π_θ. We adopt the MSE loss for the value network:

$$\mathcal{L}_V(\phi) = \frac{1}{T} \sum_{t=0}^{T-1} \left(V_\phi(\mathbf{s}_t) - R_t + \beta \sum_{i \geq t} \log \frac{\pi_\theta(\mathbf{a}_i|\mathbf{s}_i)}{\pi_{\text{ref}}(\mathbf{a}_i|\mathbf{s}_i)} \right)^2. \quad (8)$$

The policy objective is given by

$$\mathcal{L}_\pi(\theta) = \frac{1}{T}\sum_{t=0}^{T-1}\Bigg(V_\phi(\mathbf{s}_t) - R_t + \beta \log \frac{\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)} + \text{sg}\left[\beta \sum_{i>t} \log \frac{\pi_\theta(\mathbf{a}_i|\mathbf{s}_i)}{\pi_{\text{ref}}(\mathbf{a}_i|\mathbf{s}_i)}\right]\Bigg)^2 + \alpha \mathcal{L}_{\text{reg}}. \tag{9}$$

Here $\text{sg}[\cdot]$ denotes the stop gradient operator, which makes each step have the same scale in the gradient. $\mathcal{L}_{\text{reg}} = \frac{1}{T}\sum_{t=0}^{T-1} \text{KL}[\pi_\theta(\cdot|\mathbf{s}_t)\|\pi_{\text{ref}}(\cdot|\mathbf{s}_t)]$ is a regularization term that helps stabilize training.

4.2 Loss Variants

In addition to our OREO learning objective, we present two variants: step-level OREO and response-level OREO.

In step-level OREO, an action is considered to be an entire reasoning step instead of a single generated token. In the context of language models, the probability of taking an action $\mathbf{a} = (t_1 t_2 \cdots t_k)$ is $\pi(\mathbf{a}|\mathbf{s}) = \prod_{i=1}^{k} p(t_i|st_1 t_2 \cdots t_{i-1})$, where t_i denotes the ith token of the action and p denotes the language model. The step-level OREO objective can thus be modified accordingly. This objective can also be grounded in the token-level MDP formulation.

Response-level OREO aims to mimic the behavior of DPO. Instead of enforcing the consistency property at each time step, the action objective considers only the initial state, i.e.,

$$\mathcal{L}_\pi^{\text{resp}}(\theta) = \left(V_\phi(\mathbf{s}_0) - R_0 + \beta \sum_{i\geq 0} \log \frac{\pi_\theta(\mathbf{a}_i|\mathbf{s}_i)}{\pi_{\text{ref}}(\mathbf{a}_i|\mathbf{s}_i)}\right)^2 + \alpha \mathcal{L}_{\text{reg}}. \tag{10}$$

It is worth noting that response-level OREO closely aligns with DRO (Richemond et al., 2024), with the exception of the regularization term.

4.3 Iterative OREO

Previous works have shown that offline LLM fine-tuning methods can be applied iteratively to improve model performance (Pang et al., 2024; Song et al., 2024; Xiong et al., 2024a). After each training iteration, a new dataset is collected using the updated policy model to generate responses or explore the environment, which is then used for further training. More specifically, let M_0 denote the initial SFT model. One can iteratively collect a training set $\mathcal{D}_i$ using model M_i and optimize for a new model M_{i+1}. In this paper, we adopt this idea and implement an iterative version of OREO.

4.4 Test-Time Search with Value Function

Recently, inference-time scaling (Hao et al., 2024; Snell et al., 2024; Wu et al., 2024) has received significant research attention. One notable approach is the use of Process Reward Models (PRM), which evaluate whether a reasoning step is correct. During the inference time, rather than decoding the reasoning chain autoregressively from the policy model, one can conduct a tree search (e.g., beam search) guided by the PRM. Our method provides a value model for free, which estimates the expected future reward and can be directly used to guide beam search. It is also worth noting that the value function must be token-level or step-level. It is not feasible to perform test-time search with response-level value functions like DRO (Richemond et al., 2024).

In fact, previous PRM methods (Wang et al., 2024a,b; Luo et al., 2024) train their models using Monte Carlo rollouts, which are essentially similar to the objective used for training the value function in our approach (Eq. 8). Our principled formulation removes the need for the extensive heuristic designs commonly required in prior works.

We implement step-level beam search for math reasoning tasks. At each step, we maintain a set of B candidate partial trajectories. For each candidate, we generate B potential next reasoning steps. From the resulting B^2 candidates, we retain the B with the highest values.

For embodied agent tasks, where environment dynamics are unknown, beam search is not applicable. Instead, we sample K actions at each step and select the action with the highest value.

5 Experiments

In this section, we evaluate our method OREO in math reasoning and embodied agent tasks. We also demonstrate that the value function trained alongside the policy can further improve model performance at test time through step-level beam search or choosing the best-of-K action.

Datasets and Evaluation Metric. We adopt the GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) dataset for the task of math reasoning. GSM8K is a dataset of grade school math problems. It contains 7473 training problems and 1319 test problems. MATH consists of competition-level math problems, with a training set of size 7500 and a test set of size 5000. All problems in these datasets are labeled

with step-by-step ground-truth solutions. We use the script from DeepSeekMath[1] to extract the final answer from the solution and evaluate its correctness.

We adopt ALFWorld (Shridhar et al., 2020) for the task of embodied agent control. ALFWorld provides interactive TextWorld environments for household tasks. Each task is labeled with an expert trajectory. However, these data do not contain any reasoning process. Song et al. (2024) annotates 3119 ALFWorld training trajectories with rationales for each step, allowing model training with ReAct-style (Yao et al., 2022) prompting. The evaluation set of ALFWorld contains 140 tasks in seen environments and 134 tasks in unseen environments. We evaluate the success rates of agents in completing the tasks within 40 steps.

Base Models. For the math reasoning task, we select `Qwen2.5-Math-1.5B` (Yang et al., 2024) and `DeepSeekMath-7B-Instruct` (Shao et al., 2024) as our base model. For the embodied agent task, we use `MiniCPM-2B-dpo-bf16` (Hu et al., 2024) as the base model.

Baseline Methods. In addition to supervised fine-tuning, we compare our method against three other baselines:

- Rejection Sampling: The method uses the successful trajectories in the offline dataset to supervise the policy model. Despite its simplicity, rejection sampling proves to be effective in many reasoning tasks. It's also known as STaR (Zelikman et al., 2022), RAFT (Dong et al., 2023), REST (Gulcehre et al., 2023), REST$^{\text{EM}}$ (Singh et al., 2023).

- DPO (Rafailov et al., 2024b) leverages offline preference data and has been applied to solve reasoning tasks (Pang et al., 2024; Song et al., 2024).

- KTO (Ethayarajh et al., 2024) is a popular variant of DPO utilizing the Kahneman-Tversky model of human utility and is able to work on unpaired data.

We leave more details on implementation in Appendix A.

[1] https://github.com/deepseek-ai/DeepSeek-Math/tree/main/evaluation/eval

5.1 Main Results

We present the experimental results on mathematical reasoning in Table 1. Consistent with prior research (Yuan et al., 2024; Pang et al., 2024), we observe that while DPO provides marginal improvements over the SFT checkpoint used for its initialization, simpler methods such as rejection sampling often outperform DPO. In contrast, OREO demonstrates consistent superiority over all baselines across both datasets (GSM8K and MATH). This improvement is also observed universally across models in the Qwen and DeepSeekMath families. Specifically, for Qwen-2.5-Math 1.5B, OREO achieves a 5.2% relative improvement over SFT on GSM8K and a 10.5% improvement on MATH. For DeepSeekMath 7B, despite the SFT checkpoint being heavily tuned with 776K samples (Shao et al., 2024), OREO still delivers meaningful improvements, with relative gains of 3.6% on GSM8K and 5.1% on MATH. These results highlight the robustness and effectiveness of our approach across different models and datasets.

The experimental results on ALFWorld, an embodied control task, are presented in Table 2. OREO outperforms all baselines in both settings. Interestingly, rejection sampling performs well in seen environments within ALFWorld. However, its improvement is marginal in unseen settings, whereas OREO achieves a significant 17.7% relative improvement over the baseline. Compared to SFT which only learns from successful experience, OREO effectively leverages the failed trajectories, which results in more generalizable capabilities.

We evaluate different variants of the OREO objective on the math reasoning task. As shown in Figure 5, the response-level objective variant performs worse than the token-level objective. Specifically, the accuracy of response-level OREO is 74.5% in GSM8K and 52.5% in MATH. This variant treats all actions in the trajectories uniformly, making it challenging to properly assign the sparse reward to individual tokens. This limitation also sheds light on the suboptimal performance of DPO, as it struggles with credit assignment. In contrast, our method explicitly trains a value function, enabling better credit assignment and improved performance. The step-level objective, on the other hand, performs comparably to the token-level ob-

[2] The `DeepSeekMath-7B-Instruct` model is already supervised fine-tuned (Shao et al., 2024). So the SFT results are directly adopted from their paper.

Methods	Qwen 1.5B		DeepSeekMath 7B	
	GSM8K	MATH	GSM8K	MATH
SFT[2]	73.5	47.5	82.9	46.8
Rej. Sampling	74.9	50.3	83.6	47.2
DPO	74.4	49.2	82.4	47.2
KTO	73.4	48.3	82.5	46.9
OREO (Ours)	**77.3**	**52.5**	**85.9**	**49.2**

Table 1: Accuracies in GSM8K and MATH. OREO gives higher accuracies than the baselines across different datasets and model sizes. The accuracy of response-level OREO is 74.5% in GSM8K and 52.5% in MATH, using `Qwen2.5-Math-1.5B` as the base model.

Methods	Unseen	Seen
SFT	67.2	62.9
Rej. Sampling	68.7	79.3
DPO	69.4	64.3
OREO (Ours)	**79.1**	**80.7**

Table 2: Success rates in ALFWorld. OREO consistently outperforms all baselines.

jective and even slightly outperforms it on GSM8K. This may be due to the value function's limited accuracy at each step, which introduces noise into policy learning. Despite the slight performance gap, we adopt the token-level objective as the standard objective in our main experiments due to its simpler implementation (eliminating the need to segment reasoning steps). Nonetheless, step-level policy optimization remains an intriguing avenue for future exploration.

5.2 Iterative OREO

Figure 2 illustrates the performance of various algorithms on the math reasoning task across multiple iterations. OREO demonstrates steady and consistent improvements in accuracy over three iterations, showcasing its robustness in leveraging iterative training. While baseline methods also benefit from collecting additional data during each iteration, their performance consistently lags behind that of OREO. Notably, rejection sampling shows signs of saturation by the third iteration, with diminishing performance gains. In contrast, OREO continues to improve, likely due to its ability to effectively learn from failed trajectories. The updated policy model in each new iteration may be able to explore novel failure patterns, and incorporate these insights into the learning process. This potentially explains why OREO benefits more from multiple iterations compared to rejection sampling.

5.3 Comparing Implicit and Explicit Value Functions

In DPO, the policy model is viewed as an implicit value function (Rafailov et al., 2024a). However, our results in Section 5.1 have demonstrated that OREO benefits from explicitly parameterizing a separate value function. In this section, we present case studies to compare the explicit value function V_ϕ and the implicit value function derived from π_θ, aiming to provide an intuitive understanding of their differences.

Our setting is shown in Figure 1: given a problem and an existing reasoning chain prefix, we evaluate different possible continuations of the next reasoning steps, with different choices of value functions.

Assuming the token indices for the next reasoning step range from i to j, the advantage function derived from the value model V_ϕ is:

$$A_\phi = V_\phi(\mathbf{s}_j) - V_\phi(\mathbf{s}_i), \tag{11}$$

which quantifies the contribution of the new reasoning step to the expected reward. If the new step introduces an error, the estimated value of the resulting state s_j will be lower than that of the previous state, resulting in a negative advantage.

In contrast, Rafailov et al. (2024a) represent an implicit value function using the policy model, defined as: $V_\theta(\mathbf{s}_t) = V(\mathbf{s}_0) + \sum_{t'=0}^{t} \beta \log \frac{\pi_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'})}{\pi_{\text{ref}}(\mathbf{a}_{t'}|\mathbf{s}_{t'})}$. Therefore, the advantage function derived from the policy model π_θ is:

$$A_\theta = V_\phi(\mathbf{s}_j) - V_\phi(\mathbf{s}_i) = \sum_{t=i}^{j-1} \beta \log \frac{\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t \mid \mathbf{s}_t)}. \tag{12}$$

When the soft Bellman equation holds at every timestep, the two advantage functions should be equivalent, which aligns with the assumption of Rafailov et al. (2024a). However, as illustrated in Figure 1, there are cases where the two advantage functions diverge.

We observed that the magnitude of A_θ is generally smaller than that of A_ϕ. For example, in the case *"Betty now has $50 + $30 = $80"*, which is

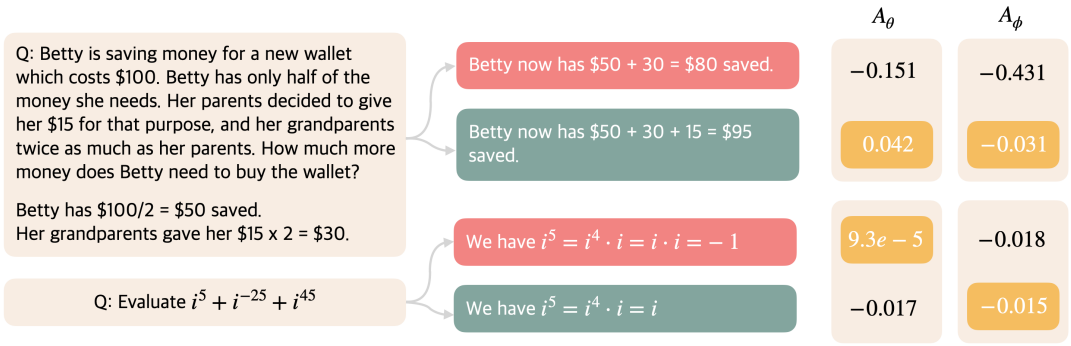

Figure 1: Case studies on the implicit and explicit value functions. Correct reasoning steps are shown in green, while incorrect ones are shown in red. Higher advantages predicted by the value functions are highlighted in yellow. Ideally, a good value function should predict a higher advantage for the correct reasoning step.

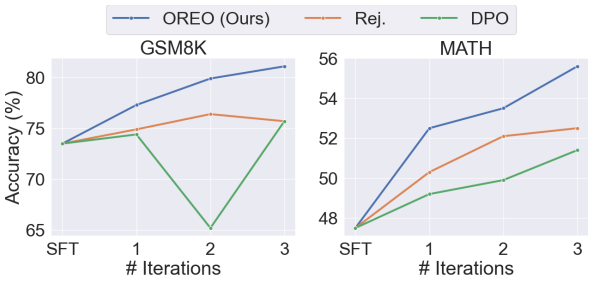

Figure 2: The accuracies on GSM8K and MATH after several iterations. Rej. stands for "rejection sampling". OREO improves as new data are collected.

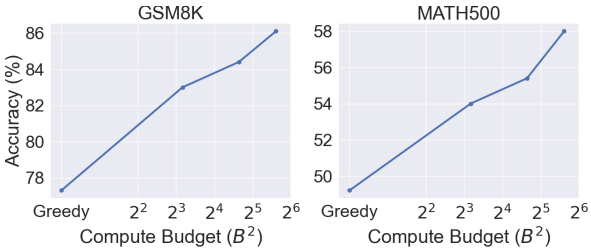

Figure 3: The accuracies of OREO on GSM8K and MATH improve with the compute budget. "Rej." stands for rejection sampling. MATH500 is a subset of MATH containing 500 queries.

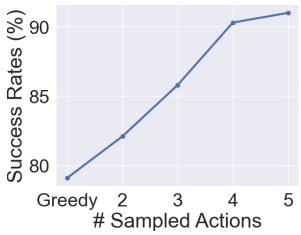

Figure 4: Success rates in ALFWorld when choosing best-of-K actions. Sampling 5 actions and choosing the one with the largest value gains significant improvement in success rates.

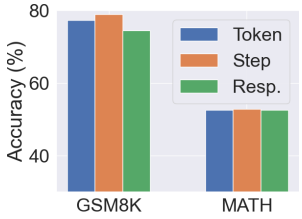

Figure 5: Different variants of OREO objective. "Token" stands for the standard OREO objective. "Step" denotes the step-level one and "Resp." denotes the response-level one. Specifically, the accuracy of response-level OREO is 74.5% in GSM8K and 52.5% in MATH.

incorrect, the value function decreases significantly from $V_\phi(s_i) = 0.816$ to $V_\phi(s_j) = 0.385$, resulting in an advantage of -0.431. In contrast, A_θ is much more conservative, indicating its weaker ability to distinguish correct and incorrect reasoning steps. While both advantage functions correctly favor the correct response in the GSM8K example above, in a more challenging MATH example below, A_ϕ successfully identifies the correct reasoning step, whereas A_θ fails.

These observations are consistent with the effectiveness of searching with an explicit value model (Feng et al., 2023; Silver et al., 2017; Snell et al., 2024). In the context of LLM reasoning, the gap between A_θ and A_ϕ may be attributed to the *softmax bottleneck* (Yang et al., 2017). Specifically, predictions for $\pi_\theta(a_t \mid s_t)$ across different actions a_t are generated from the same final hidden state, differing only with a linear language model head followed with softmax. In contrast, $V_\phi(s_{t+1})$ input the entire text sequence to the transformer network, enabling a richer representation. Exploring the gap between policy and value networks presents an intriguing direction for future research.

5.4 Test-Time Search with Value Functions

The superiority of the explicit value function motivates its use to enhance inference through search-based methods. For our experiments, we evaluate a subset of the MATH dataset containing 500 queries, a commonly used benchmark in prior work (Lightman et al., 2023; Sun et al., 2024).

Figure 3 shows the performance of step-level beam search in math reasoning. OREO leverages

the value function to achieve progressively higher accuracies as the computational budget increases. Compared to greedy decoding, beam search with $B = 7$ provides a 11.4% relative improvement in GSM8K and a 17.9% relative improvement in MATH. This indicates that the explicit value function is more effective than the policy in distinguishing between correct and incorrect reasoning steps.

Similarly, Figure 4 presents the success rates in ALFWorld when selecting the best-of-K actions. The success rates improve rapidly as the number of sampled actions increases, stabilizing with just five samples. Importantly, the value function is a natural byproduct of the OREO training framework, unlike prior work on PRM, which often involves substantial data engineering and heuristic design efforts.

6 Conclusion

In this paper, we present Offline REasoning Optimization (OREO), an offline RL algorithm for LLM reasoning and embodied LLM agent tasks. OREO leverages the idea of soft Q-learning. It trains an explicit value function together with the LLM policy by optimizing the soft Bellman Equation. This alleviates the need for paired preference data in DPO and enables fine-grained credit assignment among the reasoning steps. In addition, our value function can be used in test-time search to improve the model performance. We evaluate our method in GSM8K, MATH, and ALFWorld, demonstrating a consistent improvement compared to previous offline RLHF methods like DPO.

7 Limitations

Due to limited computation resources, some of our experiments, including ablation studies, iterative OREO, and test-times search, use 1.5B models. We plan to run experiments on larger scales in the future. Our method has primarily been evaluated on mathematical reasoning and embodied agent tasks. As future work, we aim to extend OREO to a wider variety of tasks, such as coding and web browsing, to explore its effectiveness in domains with different structures and requirements.

References

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

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

Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu, Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, et al. 2023. Rest meets react: Self-improvement for multi-step reasoning llm agent. *arXiv preprint arXiv:2312.10003*.

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

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024a. Step-level value preference optimization for mathematical reasoning. *arXiv preprint arXiv:2406.10858*.

Huayu Chen, Guande He, Lifan Yuan, Ganqu Cui, Hang Su, and Jun Zhu. 2024b. Noise contrastive alignment of language models with explicit rewards. *arXiv preprint arXiv:2402.05369*.

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

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2024. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36.

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

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. 2023. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*.

Han Guo, Bowen Tan, Zhengzhong Liu, Eric P Xing, and Zhiting Hu. 2021. Efficient (soft) q-learning for text generation with limited good data. *arXiv preprint arXiv:2106.07704*.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pages 1352–1361. PMLR.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *Preprint*, arXiv:1801.01290.

Shibo Hao, Yi Gu, Haotian Luo, Tianyang Liu, Xiyan Shao, Xinyuan Wang, Shuhua Xie, Haodi Ma, Adithya Samavedhi, Qiyue Gao, et al. 2024. Llm reasoners: New evaluation, library, and analysis of step-by-step reasoning with large language models. *arXiv preprint arXiv:2404.05221*.

Alex Havrilla, Yuqing Du, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, Sainbayar Sukhbaatar, and Roberta Raileanu. 2024. Teaching large language models to reason with reinforcement learning. *arXiv preprint arXiv:2403.04642*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Matthew Douglas Hoffman, Du Phan, David Dohan, Sholto Douglas, Tuan Anh Le, Aaron Parisi, Pavel Sountsov, Charles Sutton, Sharad Vikram, and Rif A Saurous. 2024. Training chain-of-thought via latent-variable inference. *Advances in Neural Information Processing Systems*, 36.

Jian Hu, Li Tao, June Yang, and Chandler Zhou. 2023. Aligning language models with offline reinforcement learning from human feedback. *arXiv preprint arXiv:2308.12050*.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. 2024. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. 2024. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Aviral Kumar, Joey Hong, Anikait Singh, and Sergey Levine. 2022. When should we prefer offline reinforcement learning over behavioral cloning? *arXiv preprint arXiv:2204.05618*.

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

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Guanlin Liu, Kaixuan Ji, Renjie Zheng, Zheng Wu, Chen Dun, Quanquan Gu, and Lin Yan. 2024a. Enhancing multi-step reasoning abilities of language models through direct q-function optimization. *arXiv preprint arXiv:2410.09302*.

Jiacheng Liu, Andrew Cohen, Ramakanth Pasunuru, Yejin Choi, Hannaneh Hajishirzi, and Asli Celikyilmaz. 2024b. Don't throw away your value model! generating more preferable text with value-guided monte-carlo tree search decoding. In *First Conference on Language Modeling*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. 2017. Bridging the gap between value and policy based reinforcement learning. *Advances in neural information processing systems*, 30.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

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

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *arXiv preprint arXiv:2404.19733*.

Anselm Paulus, Arman Zharmagambetov, Chuan Guo, Brandon Amos, and Yuandong Tian. 2024. Advprompter: Fast adaptive adversarial prompting for llms. *arXiv preprint arXiv:2404.16873*.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. 2024a. From r to q^*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*.

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

Pierre Harvey Richemond, Yunhao Tang, Daniel Guo, Daniele Calandriello, Mohammad Gheshlaghi Azar, Rafael Rafailov, Bernardo Avila Pires, Eugene Tarassov, Lucas Spangher, Will Ellsworth, et al. 2024. Offline regularised reinforcement learning for large language models alignment. *arXiv preprint arXiv:2405.19107*.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2020. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

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

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

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. *arXiv preprint arXiv:2403.02502*.

Zhiqing Sun, Longhui Yu, Yikang Shen, Weiyang Liu, Yiming Yang, Sean Welleck, and Chuang Gan. 2024. Easy-to-hard generalization: Scalable alignment beyond human supervision. *arXiv preprint arXiv:2403.09472*.

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

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024a. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Zihan Wang, Yunxuan Li, Yuexin Wu, Liangchen Luo, Le Hou, Hongkun Yu, and Jingbo Shang. 2024b. Multi-step problem solving through a verifier: An

empirical analysis on model-induced process supervision. *arXiv preprint arXiv:2402.02658*.

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

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

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*.

Jiannan Xiang, Tianhua Tao, Yi Gu, Tianmin Shu, Zirui Wang, Zichao Yang, and Zhiting Hu. 2024. Language models meet world models: Embodied experiences enhance language models. *Advances in neural information processing systems*, 36.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. 2024a. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. In *Forty-first International Conference on Machine Learning*.

Wei Xiong, Chengshuai Shi, Jiaming Shen, Aviv Rosenberg, Zhen Qin, Daniele Calandriello, Misha Khalman, Rishabh Joshi, Bilal Piot, Mohammad Saleh, et al. 2024b. Building math agents with multi-turn iterative preference learning. *arXiv preprint arXiv:2409.02392*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen. 2017. Breaking the softmax bottleneck: A high-rank rnn language model. *arXiv preprint arXiv:1711.03953*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, et al. 2024. Advancing llm reasoning generalists with preference trees. *arXiv preprint arXiv:2404.02078*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.

Xiang Yue, Tuney Zheng, Ge Zhang, and Wenhu Chen. 2024. Mammoth2: Scaling instructions from the web. *arXiv preprint arXiv:2405.03548*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Zirui Zhao, Hanze Dong, Amrita Saha, Caiming Xiong, and Doyen Sahoo. 2024. Automatic curriculum expert iteration for reliable llm reasoning. *arXiv preprint arXiv:2410.07627*.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Brian D Ziebart. 2010. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University.

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

A Implementation Details

A.1 Dataset Construction

We use the datasets GSM8K and MATH to train the SFT model for math reasoning. For the task of math reasoning using 1.5B models, we sample 10 responses for each query in the GSM8K dataset and the MATH dataset. Only trajectories with correct answers receive a reward 1 at the terminal state. For DPO, we pair up the positive and negative instances for each query and sample at most 6 pairs without replacement.

When training 7B models, we apply a different data-collecting strategy to balance the number of positive and negative instances. We sample 16 responses for each query and randomly select at most 4 positive instances and 4 negative instances. We then enforce that the number of positive instances

does not exceed that of negative instances. For example, if there are only 3 negative instances in the 16 sampled responses, then we only select 3 positive instances instead of 4. However, we make sure that at least 1 positive instance is selected for the query (if there is any). For DPO, we then sample at most 10 pairs of data without replacement.

For the task ALFWorld, we use the annotated data from Song et al. (2024) to train our SFT model. We perform 5 rollouts for each task in the training set. Successful trajectories receive a reward 1 at the end of the trajectory. We sample at most 5 preference pairs for DPO.

A.2 Segmentation of Reasoning Steps

In step-level beam search and step-level OREO, we use line breaks and periods to indicate the end of a reasoning step. More specifically, we use line breaks in the GSM8K dataset and use both line breaks and periods in the MATH dataset. This is because the GSM8K dataset already split reasoning steps with line breaks.

A.3 Hyperparameters

The batch size is set to 128 for all experiments.

SFT. The `Qwen2.5-Math-1.5B` model is trained on the GSM8K and MATH dataset for 3 epochs with a learning rate of 2×10^{-5}. The `DeepSeekMath-7B-Instruct` model is already instruction fine-tuned, so we do not perform any additional SFT. The `MiniCPM-2B-dpo-bf16` model is trained on the annotated dataset from Song et al. (2024) for 2 epochs with a learning rate of 2×10^{-5}.

Rejection Sampling. All experiments are trained for 1 epoch. For the math reasoning task, the learning rate is set to 5×10^{-6}. For the embodied agent task, the learning rate is 2×10^{-5}.

DPO. All experiments are trained for 1 epoch with a learning rate of 5×10^{-7} and β set to 0.1.

KTO. The `Qwen2.5-Math-1.5B` model is trained for 1 epoch with a learning rate of 5×10^{-8}. The `DeepSeekMath-7B-Instruct` model is trained for 2 epochs such that the total number of samples matches. The learning rate for the 7B model is 10^{-7}.

OREO. The learning rate is 5×10^{-6}. β is set to 0.03 and α is set to 0.01. The `Qwen2.5-Math-1.5B` is trained for 1 epoch. The `DeepSeekMath-7B-Instruct` model is trained for 2 epochs such that the total number of samples matches. The `MiniCPM-2B-dpo-bf16` model is trained for 3 epochs. To save computation, we use LoRA on the critic for 7B models. The LoRA rank and LoRA alpha are both set to 64. The learning rate for the critic is set to 10^{-4}.

For step-level OREO and response-level OREO, the loss scales are different, which demands different α in the regularization term. We experiment with $\alpha \in \{0.01, 0.1, 0.3\}$ and choose the best performance parameter. For step-level OREO, $\alpha = 0.1$. For response-level OREO, $\alpha = 0.3$.

B Proof Sketch of Theorem 1

The RLHF objective can be rewritten as

$$J_{\text{RLHF}}(\pi) = \mathbb{E}_{\tau\sim\pi}\Bigg[\sum_{t=0}^{T-1} r_{\text{task}}(\mathbf{s}_t, \mathbf{a}_t) + \beta \log \pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t) + \beta\mathcal{H}(\pi(\cdot|\mathbf{s}_t))\Bigg]. \tag{13}$$

So it can be viewed as maximum-entropy RL (Haarnoja et al., 2017) with reward $r_{\text{task}}(\mathbf{s}_t, \mathbf{a}_t) + \beta \log \pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)$. Previous works in maximum-entropy RL (Haarnoja et al., 2017) show that the optimal policy satisfies

$$\beta \log \pi^*(\mathbf{a}_t|\mathbf{s}_t) = Q^*(\mathbf{s}_t, \mathbf{a}_t) - V^*(\mathbf{a}_t), \tag{14}$$

and that the optimal Q function and optimal value function satisfies

$$Q^*(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \mathbb{E}_{\mathbf{s}_{t+1}\sim T(\cdot|\mathbf{s}_t,\mathbf{a}_t)}[V^*(\mathbf{s}_{t+1})]. \tag{15}$$

In RLHF settings, the discount factor γ is normally omitted. We combine Eq. 14 and Eq. 15 and apply the reward $r_{\text{task}}(\mathbf{s}_t, \mathbf{a}_t) + \beta \log \pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)$. This gives us

$$V^*(\mathbf{s}_t) - V^*(\mathbf{s}_{t+1}) = r_{\text{task}}(\mathbf{s}_t, \mathbf{a}_t) - \beta \log \frac{\pi^*(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)}. \tag{16}$$

C Safeguard Statement

In this paper, we primarily focus on the math reasoning tasks and embodied agent control tasks in a household simulator, posing no significant ethical or harmful concerns. We recognize that future research on border applications of multi-step reasoning may pose a risk of misuse, and we recommend careful consideration of all aspects of safety before it's applied in the real world.