SAMPLE EFFICIENT ALIGNMENT FOR LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study methods for sample-efficiently aligning large language models with human preferences given budgeted online feedback. We first formulate the LLM alignment problem in the frame of contextual dueling bandits. This bandit formulation, subsuming the recently emerging online RLHF / online DPO paradigms, naturally quests for sample-efficient algorithms. Leveraging insights from bandits, we investigate two algorithms for active exploration based on Thompson sampling and shed light on their use cases. Our agent, termed as **SEA** (**S**ample **E**fficient **A**lignment), is empirically validated with extensive experiments, across 3 scales (1B, 2.8B, 6.9B) and 3 preference learning algorithms (DPO, IPO, SLiC). The results show that **SEA** aligns the LLM with oracle's preferences highly sample-efficiently, surpassing recent active exploration methods for LLMs. We will open-source our codebase to accelerate the research in this field.

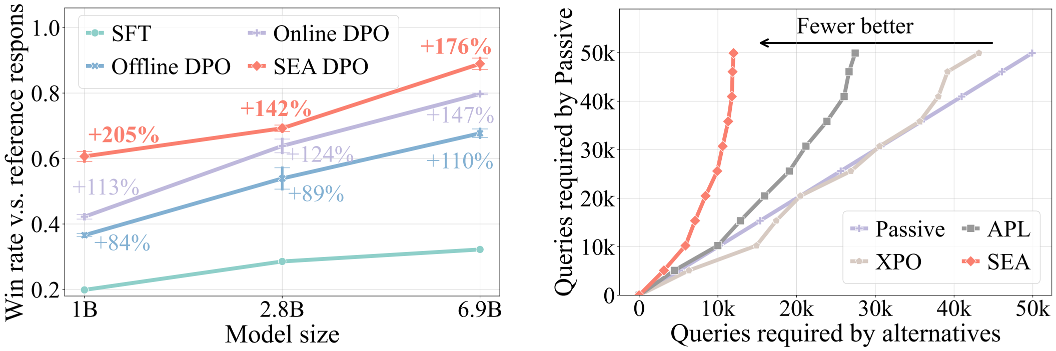


Figure 1: (Left) The win rate of different models' responses over the reference responses on the task of `TL;DR`, judged by the oracle preference model. With *active exploration*, **SEA** DPO improves over (passive) `Online DPO` by a large margin across 3 model scales from the Pythia family. **(Right)** The number of queries required by online DPO (`Passive`) versus that by active exploration methods to achieve the various levels of win rates. **SEA** achieves the best sample-efficiency for online alignment compared to two prior methods (`XPO` and `APL`).

1 INTRODUCTION

LLMs have shown remarkable abilities in various tasks, making it increasingly crucial to align them with human values. Existing alignment approaches, typically formulated using RLHF (Christiano et al., 2017; Stiennon et al., 2020; Bai et al., 2022; Ouyang et al., 2022), rely heavily on a huge amount of human annotations to provide preference feedback. As a result, the availability of high-quality human annotations becomes a major bottleneck in practical alignment applications. This poses a challenging and under-explored research question:

How to align LLMs with minimal human annotations?

In this work, we frame LLM alignment as a contextual dueling bandits (CDB) problem and highlight the strong connections between the two (detailed in Section 3). We recognize that the CDB framework facilitates LLM alignment across two different scenarios. First, when aligning LLMs with annotations collected online from commercial systems such as ChatGPT, CDB framework supports sample-efficient alignment while still delivering high-quality responses to end users. Second, when aligning LLMs through crowdsourcing, the CDB framework enables efficient learning of human preferences with minimal annotations. Additionally, this framework subsumes almost all existing

LLM alignment methods, which we will scrutinize within the CDB framework in Section 3.1, upon two key properties:

Property 1.1 (**Online interaction**). Interacting *online* allows the agent to select the most appropriate dueling responses for acquiring the preference labels based on the experience it has gathered so far. A more relaxed approach is *iterative* interaction, where the agent acquires preference labels for a batch of data and consumes them through multiple gradient steps. The most relaxed approach is *offline*, where the agent is given a dataset with pre-collected preference labels. Intuitively, the sample efficiency progressively decreases across these three paradigms.

Property 1.2 (**Active exploration**). Active exploration strategically selects dueling responses, potentially improving sample efficiency over passive exploration.

Inspired by the insights from bandits, we propose a principled algorithm based on Thompson sampling. By incorporating techniques such as including online direct alignment with active exploration, an epistemic reward model for modeling the reward posterior, and stochastic search for dueling responses selection, we introduce **S**ample **E**fficient **A**lignment for LLMs (**SEA**), a solution aligns LLM efficiently at scale. Through extensive experiments, **SEA** shows strong empirical results (see Figure 1), consistently achieving higher win rate and improved sample efficiency compared to baseline approaches across 3 different model scales. Moreover, we developed a highly efficient, distributed learning system for studying online LLM alignment methods (see Section 5.1), eliminating barriers to empirical comparisons of different algorithms. We will open-source both the learning system and the method codebase.

2 A brief review on contextual dueling bandits

We first review the definitions and two types of objectives of *Contextual Dueling Bandits* in Section 2.1. Then, in Section 2.2, we introduce a practical and efficient algorithm class known as *Thompson Sampling* that can be used to solve complex bandit problems (such as LLM alignment).

2.1 Contextual dueling bandits

Contextual dueling bandits (CDB) (Yue et al., 2012; Dudík et al., 2015) is proposed to study online learning problems where feedback consists of relative pairwise comparisons. A CDB problem can be characterized by a tuple $(\mathcal{C}, \mathcal{A}, \mathbb{P})$, where $\mathcal{C}$ is the context space, $\mathcal{A}$ is the action space, and $\mathbb{P} : \mathcal{A} \times \mathcal{A} \times \mathcal{C} \mapsto [0, 1]$ denotes the unknown *preference model*. An agent learns by iteratively interacting with the environment or oracle (i.e., the preference model $\mathbb{P}$) as follows. At each round t of the learning process, a context $\boldsymbol{c}_t \in \mathcal{C}$ is presented to the agent, who needs to take two actions $\boldsymbol{a}_t, \boldsymbol{a}'_t \in \mathcal{A}$ for a "dueling" comparison. The agent then receives stochastic feedback in the form of a comparison result $z_t \sim \mathrm{Ber}\left(\mathbb{P}\left(\boldsymbol{a}_t \succ \boldsymbol{a}'_t | \boldsymbol{c}_t\right)\right)$ from the environment, where $\mathrm{Ber}(\cdot)$ is the Bernoulli distribution and $\succ$ denotes that the first action is preferred.

Regret. The quality of the dueling actions selected by the agent is measured by the *immediate regret*: $R_t = \mathbb{P}(\boldsymbol{a}^\star_t \succ \boldsymbol{a}_t | \boldsymbol{c}_t) + \mathbb{P}(\boldsymbol{a}^\star_t \succ \boldsymbol{a}'_t | \boldsymbol{c}_t) - 1$, where $\boldsymbol{a}^\star_t$ is the best action[1] the agent would take at round t if it had complete knowledge of $\mathbb{P}$. Intuitively, if the agent has learned how to act optimally from round t onwards, it would no longer suffer any regret since its actions would be indistinguishable from the best action ($\mathbb{P}(\boldsymbol{a}^\star_\tau \succ \boldsymbol{a}_\tau | \boldsymbol{c}_\tau) = \frac{1}{2}$ and $R_\tau = 0$ for $\tau \geq t$).

Optimal policy. A policy $\pi \in \Delta_{\mathcal{A}}^{\mathcal{C}}$[2] associates each context $\boldsymbol{c} \in \mathcal{C}$ with a probability distribution $\pi(\cdot|\boldsymbol{c}) \in \Delta_{\mathcal{A}}$ over the action space. The *total preference* of policy π over policy μ given a context sampling distribution $p_{\mathcal{C}}$ and a preference model $\mathbb{P}$ is defined as

$$P_{p_{\mathcal{C}},\mathbb{P}}(\pi \succ \mu) = \mathop{\mathbb{E}}_{\boldsymbol{c} \sim p_{\mathcal{C}}} \left[\mathop{\mathbb{E}}_{\boldsymbol{a} \sim \pi(\cdot|\boldsymbol{c})} \mathop{\mathbb{E}}_{\boldsymbol{a}' \sim \mu(\cdot|\boldsymbol{c})} \left[\mathbb{P}(\boldsymbol{a} \succ \boldsymbol{a}' | \boldsymbol{c}) \right] \right]. \tag{1}$$

We adopt the *von Neumann winner* (Dudík et al., 2015) as the measure of optimality, which requires the optimal policy $\pi^\star$ to satisfy that

$$\forall \pi' \in \Delta_{\mathcal{A}}^{\mathcal{C}},\ P_{p_{\mathcal{C}},\mathbb{P}}(\pi^\star \succ \pi') \geq \frac{1}{2}, \tag{2}$$

[1] We assume that a best action $\boldsymbol{a}^\star$ in the sense that $\mathbb{P}(\boldsymbol{a}^\star \succ \boldsymbol{a} | \boldsymbol{c}) \geq \frac{1}{2}, \forall \boldsymbol{a} \in \mathcal{A}$ exists for all context $\boldsymbol{c} \in \mathcal{C}$.

[2] We denote by $\Delta_{\mathcal{A}}^{\mathcal{C}}$ the set of all mappings $\mathcal{C} \mapsto \Delta_{\mathcal{A}}$, where $\Delta_{\mathcal{A}}$ denotes the set of all probability distributions over $\mathcal{A}$.

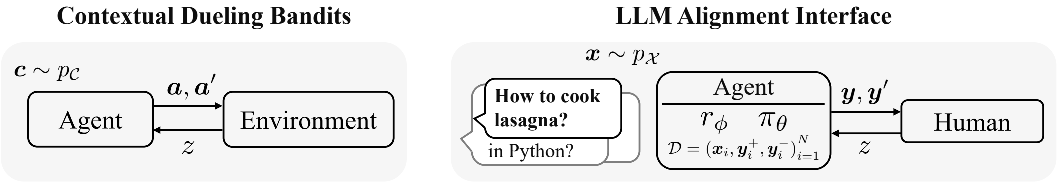


Figure 2: Illustrative comparison between CDB and LLM alignment.

In words, the von Neumann winner policy should beat or tie with every policy (i.e., is zero-regret) on average.

Learning objectives. The goal of bandit agents is to learn the optimal policy through environment interaction. There are two subtypes of objectives that focus on different learning scenarios. The first type considers the conventional *explore and exploit (E&E)* setting (Robbins, 1952; Auer et al., 2002), where the agent learns fully online and tries to minimize the cumulative regret over T rounds: $\sum_{t=1}^{T} R_t$. The second type of objective concerns the *best arm identification (BAI)* setting (Bubeck et al., 2009; Audibert & Bubeck, 2010), where the agent is only evaluated offline on its average performance, possibly at any round (a.k.a., anytime regret), and tries to learn the optimal policy with minimum interaction. Their differences will be made clearer with real scenarios in Section 3.

2.2 Thompson sampling

Thompson sampling (TS) (Thompson, 1933) is widely adopted to solve bandit problems at scale due to its great efficiency and strong empirical performance for general online learning problems (Chapelle & Li, 2011; Russo et al., 2018). A bandit agent with Thompson sampling typically maintains and incrementally updates a posterior distribution of the oracle reward model $p_t(r|\mathcal{D}_t)$ at each round given experience $\mathcal{D}_t$. Meanwhile, the agent takes actions following a greedy policy with respect to a sampled reward model: $\boldsymbol{a}_t = \arg\max_{\boldsymbol{a}} r_t(\boldsymbol{a})$, where $r_t \sim p_t(\cdot|\mathcal{D}_t)$. This simple yet effective algorithm balances exploration and exploitation naturally: when the agent has limited knowledge about the environment, its posterior estimate exhibits high uncertainty so that the sampled greedy policy explores; after gathering necessary experience, the sampled reward model may approximate the oracle well and deliver near optimal policy to exploit.

3 LLM alignment as contextual dueling bandits

The problem of LLM alignment can be framed as a CDB with their correspondences illustrated in Figure 2. Specifically, a text prompt (cf. context) $\boldsymbol{x} \in \mathcal{X}$ is sampled from a prompt distribution $p_{\mathcal{X}}$. Then, two distinct responses (cf. actions), $\boldsymbol{y}, \boldsymbol{y}' \in \mathcal{Y}$, are chosen by the agent, and presented to human annotators (cf. the environment) for preference ranking. The winning and losing responses are labeled as $(\boldsymbol{y}^+, \boldsymbol{y}^-)$ based on a binary stochastic feedback z. The agent is expected to learn (or so-called align with) human preferences from the interaction experiences $\mathcal{D}$.

A standard assumption is that the human preference follows the Bradly-Terry (BT) model (Bradley & Terry, 1952):

$$\mathbb{P}(\boldsymbol{y} \succ \boldsymbol{y}'|\boldsymbol{x}) = \frac{\exp\left(r^\star(\boldsymbol{x},\boldsymbol{y})\right)}{\exp\left(r^\star(\boldsymbol{x},\boldsymbol{y})\right) + \exp\left(r^\star(\boldsymbol{x},\boldsymbol{y}')\right)} = \sigma(r^\star(\boldsymbol{x},\boldsymbol{y}) - r^\star(\boldsymbol{x},\boldsymbol{y}')), \tag{3}$$

where σ is the sigmoid function and $r^\star$ encodes human's implicit reward. With this assumption, the regret of LLM alignment can be rewritten as $R_t = r^\star(\boldsymbol{x},\boldsymbol{y}^\star) - \left(r^\star(\boldsymbol{x},\boldsymbol{y}) + r^\star(\boldsymbol{x},\boldsymbol{y}')\right)/2$ (Saha, 2021; Li et al., 2024), where $\boldsymbol{y}^\star$ is the best response for prompt $\boldsymbol{x}$ given the oracle implicit reward, i.e., $r^\star(\boldsymbol{x},\boldsymbol{y}^\star) \geq r^\star(\boldsymbol{x},\boldsymbol{y}), \forall \boldsymbol{y} \in \mathcal{Y}$. The von Neumann winner policy is also redefined as

$$\pi^\star \in \arg\max_{\pi} J(\pi), \text{ where } J(\pi) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{\mathcal{X}}} \mathop{\mathbb{E}}_{\boldsymbol{y}\sim\pi(\cdot|\boldsymbol{x})} \left[r^\star(\boldsymbol{x},\boldsymbol{y})\right] \text{ is the objective,} \tag{4}$$

by substituting Eq. (3) into Eq. (1) and maximizing $P_{p_{\mathcal{C}},\mathbb{P}}(\pi \succ \pi^\star)$ towards $1/2$.

The **two objectives in bandits** have their respective applications in LLM alignment. **(1)** The E&E setting applies to the scenario of serving an LLM-based application online and aligning it continually with users' preferences. In this setting, the cumulative regret is of interest because the quality of *every* answer matters. In fact, commercial systems like ChatGPT would strategically ask users to make a dueling comparison, while upholding the quality of both answers. Please see Figure 8 for an

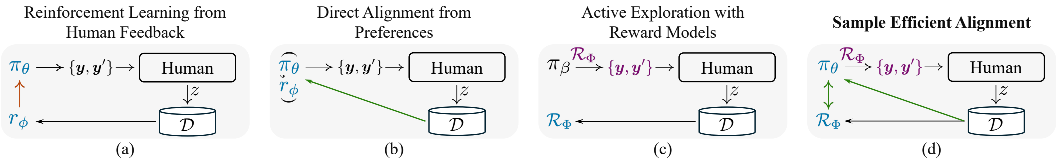


Figure 3: Different instantiations for solving the LLM alignment problem in the CDB framework. We use different colors to denote learnable components, RL optimizer, direct optimizer, and active exploration.

example. **(2)** The BAI setting corresponds to the other scenario where annotators are paid to provide human feedback (Christiano et al., 2017; Ouyang et al., 2022). The desideratum in this scenario is to align the LLM at the minimum labeling cost, while the quality of the dueling responses is not important as long as the experience helps *sample-efficiently* learn the von Neumann winner policy.

After formalizing LLM alignment in the framework of CDB and uncovering their tight connections, we next thoroughly discuss related work in the context of the CDB framework (Section 3.1), then present a unified (but intractable) Thompson sampling algorithm to efficiently align LLMs for different objectives (Section 3.2). Both of the insufficiencies of existing work and the algorithmic insights will lead us to finalizing one practical algorithm in Section 4.

3.1 How does prior work solve LLM alignment as CDB?

Before moving on to discuss existing LLM alignment literature in the CDB framework, there is a need to align terminologies used so far with commonly referred ones in the LLM community. Previously, we use the word “agent” to denote everything except the environment, and refer to its behavior as a “policy”, following a standard abstraction in RL (Sutton & Barto, 2018; Sutton et al., 2022). This, for example, applies to the definition of optimal policy in Eq. (4). However, in LLM literature, a “policy” typically refers to the generative language model alone, excluding components like reward models the agent might additionally build. To avoid confusion, from now on we use π_{θ^t} to denote the generative language model (policy) at time t, and r_{ϕ^t} to denote the (optional) reward model learned from preference data D_t collected till time t. We will omit t when the time-indexing is not important in the context.

Commonly adopted RLHF pipelines (Christiano et al., 2017; Stiennon et al., 2020; Bai et al., 2022; Ouyang et al., 2022) learn reward models as the first step with a negative log-likelihood loss:

$$\mathcal{L}_r(\phi|\mathcal{D}) = -\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}^+,\boldsymbol{y}^-)\sim p_\mathcal{D}}\left[\log\sigma\left(r_\phi\left(\boldsymbol{x},\boldsymbol{y}^+\right) - r_\phi\left(\boldsymbol{x},\boldsymbol{y}^-\right)\right)\right], \tag{5}$$

where $\mathcal{D}$ is collected by querying human annotators using a behavior policy π_{ref}, which is typically obtained by supervised fine-tuning. Afterwards, *offline* RL (Levine et al., 2020) is conducted with respect to the learned reward r_ϕ internally within the agent in the CDB framework (Figure 3**(a)**). However, the learned model π_θ might be inaccurate at regions out of the distribution (o.o.d.) of π_{ref} because little training data can be collected. A typical remedy is to incorporate a pessimistic objective to combat such distributional shift. To this end, we rewrite the objective of von Neumann winner policy in Eq. (4) as

$$J(\pi_\theta) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_\mathcal{X}}\mathop{\mathbb{E}}_{\boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x})}\left[\underbrace{r_\phi(\boldsymbol{x},\boldsymbol{y})}_{\text{estimated } r^\star} \underbrace{-\eta\log\frac{\pi_\theta(\boldsymbol{y}|\boldsymbol{x})}{\pi_{\text{ref}}(\boldsymbol{y}|\boldsymbol{x})}}_{\text{o.o.d. reward penalty}}\right] \tag{6}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_\mathcal{X}}\left[\mathop{\mathbb{E}}_{\boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x})}\left[r_\phi(\boldsymbol{x},\boldsymbol{y})\right] - \eta D_{\text{KL}}(\pi_\theta(\cdot|\boldsymbol{x})||\pi_{\text{ref}}(\cdot|\boldsymbol{x}))\right], \tag{7}$$

which involves a KL penalty widely used for language model fintuning (Jaques et al., 2020; Xiong et al., 2024). PPO (Schulman et al., 2017) as an *RL optimizer* naturally suits Eq. (7) well due to its KL-regularized trust region policy update (Schulman et al., 2015) and has been used widely.

The RLHF pipeline is illustrated by subfigure **(a)** of Figure 3. Though effective in aligning LLMs with human values, it is either performed with iterative interactions, or with an offline dataset to learn the internal reward model, instead of with online interactions. Moreover, Applying RL optimizer with the internal reward model is complex and requires many subtle tricks to stabilize the training, limiting its broader application (Huang et al., 2024). Direct alignment from preferences

Algorithm 1 Thompson Sampling for LLM Alignment (Intractable)

```
   Input: Prompt distribution $p_{\mathcal{X}}$, unknown but queryable preference oracle $\mathbb{P}$.
1: Initialize $\mathcal{D}_0 \leftarrow \varnothing$.
2: for $t = 1, \dots, T$ do
3:     Receive a prompt $\boldsymbol{x}_t \sim p_{\mathcal{X}}$.
4:     Sample $r \sim p(\cdot|\mathcal{D}_{t-1})$ and set $\boldsymbol{y} \leftarrow \arg\max_{\boldsymbol{b} \in \mathcal{Y}} r(\boldsymbol{x}_t, \boldsymbol{b})$.        // Select 1st response $\boldsymbol{y}$.
   // E&E objective: aligning an online system.
5:     repeat
           Sample $r \sim p(\cdot|\mathcal{D}_{t-1})$ and set $\boldsymbol{y}' \leftarrow \arg\max_{\boldsymbol{b} \in \mathcal{Y}} r(\boldsymbol{x}_t, \boldsymbol{b})$.        // Select 2nd response $\boldsymbol{y}'$.
       until $\boldsymbol{y}' \neq \boldsymbol{y}$
   // BAI objective: labeling via crowdsourcing.
6:     Set $\boldsymbol{y}' \leftarrow \arg\max_{\boldsymbol{b} \in \mathcal{Y}} \mathbb{V}\left[\sigma\left(r(\boldsymbol{x}, \boldsymbol{y}) - r(\boldsymbol{x}, \boldsymbol{b})\right)\right]$,        // OR select 2nd response $\boldsymbol{y}'$.
           where $\mathbb{V}[\cdot]$ computes variance over the posterior $p(\cdot|\mathcal{D}_{t-1})$.
7:     Query $\mathbb{P}$ and update experience $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \bigcup \{\boldsymbol{x}_t, \boldsymbol{y}_t^+, \boldsymbol{y}_t^-\}$.
8: end for
```

(DAP), introduced by DPO (Rafailov et al., 2023), simplifies training by eliminating the need for an internal reward model (Figure 3(**b**)). However, many early works following DPO either rely on offline datasets (Azar et al., 2024; Ethayarajh et al., 2024; Wu et al., 2024; Meng et al., 2024) or engage in iterative interactions (Dong et al., 2024). OAIF (Guo et al., 2024) takes a step forward by enabling fully online learning, improving sample efficiency over its offline and iterative counterparts. Nevertheless, these methods still employ passive exploration strategies.

A line of work (Mehta et al., 2023; Das et al., 2024; Melo et al., 2024; Dwaracherla et al., 2024) adopts the bandit formulation and focuses on reward model learning, incorporating active exploration and online interactions (Figure 3(**c**)). They generate responses from a fixed policy, π_β, and utilize a reward model to select the dueling responses. However, their performance is limited by the sub-optimal sample efficiency due to the non-adaptive proposal distribution π_β. Built on recent OAIF (Guo et al., 2024), several works have proposed to incorporate active exploration, either by adding an optimistic term in the loss function (Zhang et al., 2024; Xie et al., 2024), or by actively selecting dueling responses using implicit rewards induced from DPO training (Muldrew et al., 2024). Yet, these methods are tightly coupled with DPO and are not compatible with other direct optimizers. Given their relevance to our approach, we will include comparisons with these methods in our experiments when using DPO as the direct optimizer. We summarize the prior work in Table 2 in the Appendix. Next we introduce our approach

3.2 Thompson sampling for LLM alignment

We could leverage the BT model assumption (Eq. (3)) to cast the preference oracle $\mathbb{P}$ into the reward oracle $r^\star$, so that we can model the reward posterior $p(r|\mathcal{D})$ similar as in conventional bandits. Note that the LLM agent is fully described by the posterior $p(r|\mathcal{D})$ in this context. We take inspiration from prior work (Wu & Liu, 2016; González et al., 2017), which only discusses non-contextual K-arm bandits and preferential Bayesian optimization problems, and generalize them to the context of LLM alignment and develop a unified algorithm as shown in Algorithm 1.

As Algorithm 1 presents, the first response of the duel is always selected via a typical TS step (Line 4). The selection of the second response varies across different settings. Line 5 will be used for scenarios where preference feedback is collected from online users (the E&E setting). The dueling responses selected in this case will both try to maximize a sampled reward model, so that the online user experience is warranted with best effort. However, such algorithm can have poor asymptotic performance for BAI problems (Russo, 2016), because sub-optimal responses with confidently high rewards might be tried for a long time at the expense of not exploring other potentially better responses. In light of this, Line 6 provides an alternative for scenarios where we could hire annotators for feedback and low-quality but exploratory responses are safe. Specifically, Line 6 selects the second response as the one that maximizes the variance of the preference outcome (Eq. (3)) compared with the first response $\boldsymbol{y}$. This variance quantifies the *epistemic uncertainty* of the reward model, pointing the agent to the maximally informative direction to explore.

However, Algorithm 1 is yet to be practical for LLM alignment for three main reasons. First, existing LLM agents (Achiam et al., 2023; Touvron et al., 2023) typically consist in a generative model (e.g., a transformer (Vaswani et al., 2017)), while the algorithm above is centered around a reward posterior that cannot be easily converted into a generative model. Second, computing and sampling from a reward posterior is intractable for nearly all reward models that can be used for LLMs, which are mostly based on large transformers (Lambert et al., 2024). Last but not least, even if we managed to approximate a reward posterior, the arg max operation for action selection is also intractable since it requires searching over the entire response space $\mathcal{Y}$ which is massive for language. We will address these problems and present a highly sample-efficient alignment (**SEA**) algorithm in Section 4.

4 **SEA**: SAMPLE EFFICIENT ALIGNMENT FOR LLMS

We now present our approach to bridging the gap between Algorithm 1—a principled algorithm inspired by bandits' insights—and a practical, efficient LLM alignment algorithm. This approach addresses the three issues discussed in Section 3.2. First, we conduct preference tuning directly from the agent's exploratory on-policy experience to improve the generative model π_θ online. Second, we employ an epistemic reward model based on ensemble to efficiently approximate the posterior $p(r|\mathcal{D})$. Third, we further update π_θ to align with the internal epistemic reward model, which allows for a better approximation of the $\arg\max$ operation used in TS.

4.1 ONLINE DIRECT ALIGNMENT WITH ACTIVE EXPLORATION

We update the policy π_θ with DAP losses (i.e., direct optimizers) online in a similar vein to Guo et al. (2024):

$$\mathcal{L}_\pi(\theta^t|\mathcal{B}^t, \pi_{\text{ref}}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}^+,\boldsymbol{y}^-)\sim p_{\mathcal{B}^t}} \left[F_{\theta^t}(\boldsymbol{x}, \boldsymbol{y}^+, \boldsymbol{y}^-, \pi_{\text{ref}})\right], \tag{8}$$

where $\mathcal{B}^t$ is a batch of preference data collected online by the online policy π_{θ^t} (thus $\mathcal{B}^t$ is on-policy), and F could be any DAP loss. Importantly, this makes our method generally applicable to any direct optimizer that solves Eq. (4) directly from preference data.

In a stark contrast to Guo et al. (2024), which only passively explores by sampling $\boldsymbol{y}, \boldsymbol{y}' \sim \pi_{\theta^t}$ randomly, we learn the reward posterior and actively explore via Thompson sampling. As benefits of active exploration, the dueling responses would either provide better online user experience (for E&E agents) or incur less labelling cost (for BAI agents). Another more subtle difference between ours and Guo et al. (2024) is regarding $\mathcal{B}^t$: by active exploration we obtain more informative preference feedback that could help the agent learn more sample-efficiently towards the optimal policy.

4.2 EPISTEMIC REWARD MODEL FOR POSTERIOR SAMPLING

To implement the active exploration with TS, we seek an efficient way to maintain and incrementally update the reward posterior $p(r|\mathcal{D})$. We consider ensemble for our purpose for its capability to model epistemic uncertainty well (Lakshminarayanan et al., 2017) and provable results when used for TS in linear bandits (Qin et al., 2022). In particular, we update a set of reward models independently online using the preference data and a regularized negative log likelihood loss:

$$\mathcal{L}_\mathcal{R}(\Phi^t|\mathcal{D}_t) = \sum_{k=1}^{K} \left(\mathcal{L}_r(\phi_k^t|\mathcal{D}_t) - \lambda||\phi_k^t - \phi_k^0||\right), \tag{9}$$

where $\mathcal{L}_r$ is defined in Eq. (5), $\Phi^t = \{\phi_k^t\}_{k=1}^K$ is the weights of the ensemble of size K, and λ controls the regularization towards its initial weights ϕ_k^0 to retain the diversity across ensemble members (Dwaracherla et al., 2020). In practice, we train K MLP heads on top of a pretrained transformer. We refer to the ensemble as the Epistemic Reward Model (ERM, denoted as $\mathcal{R}$), with which the posterior sampling simply amounts to randomly picking a ϕ_k from Φ.

4.3 STOCHASTIC SEARCH FOR DUELING RESPONSES SELECTION

With the ERM approximating the reward posterior, we need to further approximate the $\arg\max$ operation in the TS algorithms to select dueling responses. To this end, for any prompt $\boldsymbol{x}$, we sample M responses from the online policy $\pi_{\theta^t}(\cdot|\boldsymbol{x})$ to construct a candidate set $\mathcal{S} = \{\boldsymbol{y}_i\}_{i=1}^M$, and search for the local optimum within $\mathcal{S}$. Intuitively, this Monte Carlo optimization is unbiased if $\mathcal{S}$ uniformly covers $\mathcal{Y}$ (i.e., covering infinitely many possible responses). However, as we optimize π_{θ^t} online with oracle preference data, $\mathcal{S}$ is biased to contain responses with high oracle reward $r^\star$. Bias towards high-$r^\star$ region is generally helpful because it aligns with $\arg\max_{\boldsymbol{b}\in\mathcal{Y}} r(\boldsymbol{x}, \boldsymbol{b})$ to seek high-reward response. However, optimizing π_{θ^t} only with oracle data averages out the epistemic uncertainty of $\mathcal{R}$, hindering the exploration efficiency. To mitigate this issue, we further align π_{θ^t} with the ERM using the same direct optimizer to encourage π_{θ^t} to propose high-$r_{\phi_k^t}$ responses for individual $r_{\phi_k^t}$. In practice, we implement this by optimizing Eq. (8) over a mixture batch distribution

$p_{\mathcal{B}^t_{\text{mix}}} = \gamma p_{\mathcal{B}^t} + (1-\gamma) p_{\mathcal{B}^t_{\text{ERM}}}$, where γ controls the mixture ratio and $\mathcal{B}^t_{\text{ERM}} = \{\boldsymbol{x}_i, \tilde{\boldsymbol{y}}_i^+, \tilde{\boldsymbol{y}}_i^-\}_{i=1}^b$ consists of preference data labeled by randomly sampled individual ensemble members $r_{\phi_k^t}$, which facilitates a better approximation of $\arg\max_{\boldsymbol{b}\in\mathcal{Y}} r(\boldsymbol{x}, \boldsymbol{b})$ for any sampled r.

5 Experimentation Setup

In this section we elaborate the experimentation setup we employ to validate our algorithm and fairly compare to other online alignment baselines. We start by introducing the distributed learning system[3] we build for experimenting online LLM alignment with simulated preference oracle (Section 5.1), then provide all the experiment details including models, data and performance metrics, etc. (Section 5.2).

5.1 Distributed Learning System

The interactive nature of LLM alignment necessitates an integrated online learning system that simulates the interface described in Figure 2 (right). The absence of a performant open-source online alignment system has restricted many existing work to a few iterations of batch learning (Muldrew et al., 2024; Dong et al., 2024; Zhang et al., 2024; Xie et al., 2024), which hinders the potential of online algorithms. Even worse, such absence also makes the comparison between different exploration methods difficult, thus the newly proposed method is usually only compared to the most naive iterative DAP baseline.

To fill this gap, we build a highly efficient learning system for studying methods in online LLM alignment. Inspired by distributed deep RL systems (Espeholt et al., 2018), we design an Actor-Learner-Oracle architecture for our purpose, which is depicted in Figure 4. The three types of workloads are heterogeneous and requires different optimization. In particular, we adopt vLLM (Kwon et al., 2023) for the actor to accelerate the autoregressive response generation. We also use DeepSpeed's ZeRO (Rasley et al., 2020; Rajbhandari et al., 2020) strategies to enhance the memory efficiency of the learner. The weights of the model are broadcasted from the learner master to actors after every update. We wrap the oracle reward model as a service using Mosec (Yang et al., 2021b), which supports dynamic batching and parallel processing, to minimize the query latency. Finally, we use DeepMind Launchpad (Yang et al., 2021a) to compose all workloads into a distributed program.

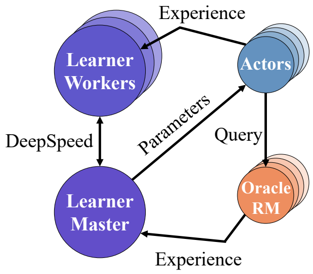


Figure 4: Our learning system for studying online LLM alignment algorithms.

5.2 Experiment Details

Models. We experiment three model scales (1B, 2.8B, 6.9B) from the Pythia family (Biderman et al., 2023). We take pretrained SFT models from Huang et al. (2024) as π_{ref} for the initial model all experiments.

Reward oracle. We simulate the process of human feedback with a strong scalar reward model and refer it as reward oracle. We choose `Skywork-Reward-Llama-3.1-8B`[4] (Liu & Zeng, 2024), which is top-ranked in RewardBench leaderboard (Lambert et al., 2024), as the reward oracle.

Epistemic reward model. We build ERM on top of a pretrained 0.4B transformer (Jiang et al., 2023), by removing its head and adding an ensemble of MLPs. The size of ensemble is set to 20, and all MLPs contain 2 hidden layers of 128 nodes. Note that the ERM is chosen to be much smaller than the reward oracle following prior work (Dwaracherla et al., 2024), which reflects the fact that human preference may be more complex than what the agent can model.

Data. We employ the widely adopted `TL;DR` dataset (Stiennon et al., 2020) for our experiments. It consists of Reddit posts as prompts, and the agent is required to give summaries that align with human preferences. We fix 50k prompts for training and limit the query budget to 50k as well.

[3] We will open-source the codebase after the reviewing process.

[4] https://huggingface.co/Skywork/Skywork-Reward-Llama-3.1-8B.

DAP methods. We adopt three DAP methods to thoroughly validate our algorithm, including DPO (Rafailov et al., 2023), IPO (Azar et al., 2024) and SLiC (Zhao et al., 2023).

Baselines. We include the offline and online variants of different DAP methods as baselines, which are studied by (Guo et al., 2024). Additionally, we compare with two active exploration baselines built on online DPO: APL (Muldrew et al., 2024) and XPO (Xie et al., 2024). We omit the comparison with SELM (Zhang et al., 2024) since SELM and XPO share a very similar algorithmic design.

Metrics. We use the win rate of agent's responses against reference responses judged by the reward oracle as the performance metric. This metric can reflect both the agent's cumulative regret and anytime regret (i.e., average performance). In the E&E setting, we measure the "online" win rate of the agent's dueling responses that are executed during experience collection. In the BAI setting, we measure the "offline" win rate by evaluating the agent's responses given a fixed set of holdout prompts periodically. We mainly focus on the BAI setting because crowdsourcing seems a major scenario for most practitioners, and present one set of experiments for comparing different exploration strategies in both settings. When the comparison is only made within a model scale, we report the relative win rate against the initial STF models. When the comparison is across scales (Figure 1 Left), we report the absolute win rate against the ground truth responses in the dataset.

Hyperparameters. We set $\beta = 0.1$ for DPO and $\beta = 0.2$ for SLiC and find they are robust for all scales. We tune β from $\{0.2, 0.3, 0.5, 1.0\}$ for IPO across scales and report the best performing results. We sample 20 on-policy responses with a temperature of 0.7 during training, and use greedy decoding for offline evaluation (BAI's metric). We use the Adam optimizer with learning rate of $5e-7$ and cosine scheduling. We initialize the mixture ratio γ of **SEA** as 1 and adjust it to 0.7 after a burn-in period of 1k samples. We follow the recommended hyperparameters of APL and XPO from their papers.

Statistical significance. There are various factors to introduce randomness during online learning. We thus launch 3 independent runs for every experiment with different random seeds. All the results are reported along with standard errors indicating their statistical significance.

Computational resources. Experiments for all scales can be run on 8 A100 GPUs for learner and actors. We host a separate remote server on 16 A100 GPUs for the oracle reward model, so that it can be queried by all concurrently running experiments. All experiments conducted for this research consume about 2 A100 years.

6 Empirical Results

In this section we present our empirical results and analyses. We organize this section into four parts: (1) An overall comparison between **SEA** and baselines across direct optimizers and model scales. (2) An ablation analysis of **SEA**. (3) A comparison of different exploration strategies in E&E and BAI settings. (4) Additional results when aligning with a human simulator by GPT4o-mini.

6.1 Overall Comparison

We first compare **SEA** with all baselines across three model scales and three direct optimizers. APL and XPO are only compared when we use DPO as the direct optimizer, because they are not compatible with IPO or SLiC. Figure 5 shows the win rate curves versus query steps. Across all settings, `Online` agents improve sample efficiency over their `Offline` counterparts, validating the need of Property 1.1 for alignment algorithms. Focusing on the first row, among prior active exploration methods, `XPO` gives a minor improvement on final performance over `Online` (passive) at 1B scale, but falls short for larger scales. On the other hand, `APL` shows a significant efficiency boost at 1B scale, but the return diminishes when scaling up and it performs almost the same as `Online` at 6.9B scale. Our method, **SEA**, outperforms offline and online passive methods across all scales and all direct optimizers, confirming the role Property 1.2 plays for sample-efficient alignment. Meanwhile, in the special case of using DPO as the direct optimizer, **SEA** also shows superior performance to prior online active exploration methods including `APL` and `XPO`.

Additionally, we note that the choice of direct optimizer matters for both online learning and active exploration. Comparing different optimizers at 1B scale (the first column), all `Offline` agents learn comparably and reach the same level of final performance (about 70% win rate), but SLiC

Table 1: Decomposition of different driving factors of online active alignment algorithms.

Variant	Inference (Test)	Exploration	Learn	Remark
1	π_θ	passive	π_θ	Online DAP (Guo et al., 2024)
2	π_θ	active	$(\pi_\theta, \mathcal{R})$	**SEA** *without* self-alignment (Section 4.3)
3	π_θ	active	$(\pi_\theta \leftrightarrow \mathcal{R})$	**SEA**
4	$\mathrm{BoN}(\pi_\theta, \mathcal{R})$	passive	$(\pi_\theta, \mathcal{R})$	-
5	$\mathrm{BoN}(\pi_\theta, \mathcal{R})$	active	$(\pi_\theta, \mathcal{R})$	-
6	$\mathrm{BoN}(\pi_\theta, \mathcal{R})$	active	$(\pi_\theta \leftrightarrow \mathcal{R})$	**SEA** with Best-of-N sampling
7	$\mathrm{BoN}(\pi_{\mathtt{ref}}, \mathcal{R})$	active	$\mathcal{R}$	Not learn policy (Dwaracherla et al., 2024)

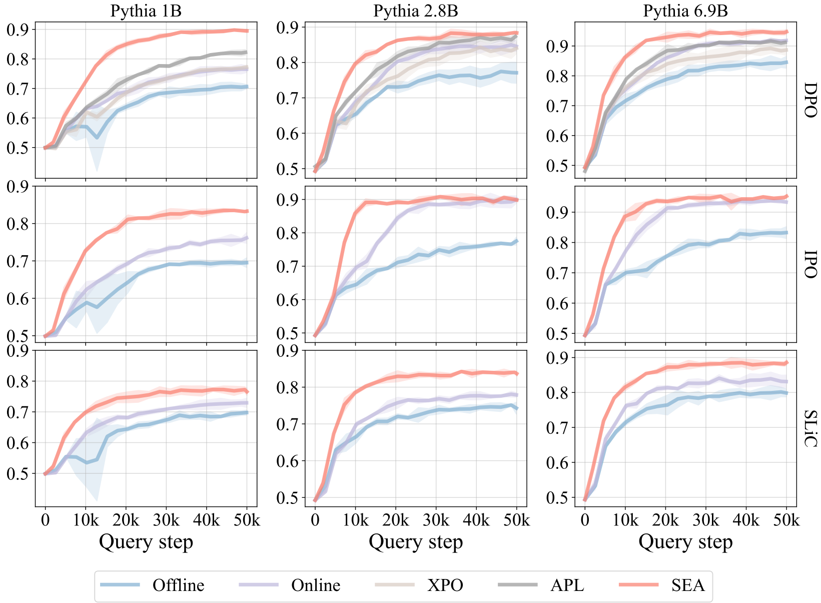


Figure 5: Comparison on the win rates of different agents against SFT models across three model scales and three direct optimizers.

Online agent deliver slightly less improvement than DPO and IPO Online agents. Besides, when incorporating active exploration, DPO **SEA** agent shows much larger improvement than the other two. This suggests that selecting the most suitable policy optimizer coupled with active exploration would yield the best agent.

6.2 Ablation Analysis

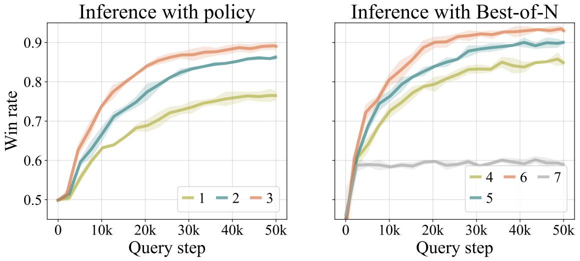


Figure 6: Comparison on the win rates of different agent variants when using **(Left)** policy and **(Right)** Best-of-N sampling for inference.

Next we decompose **SEA** into different components and ablate their contributions. Table 1 shows three axes that we dissect **SEA** on, including the inference method, exploration strategy and learning components. We construct 7 agent variants from different combinations, which cover two closely related baselines (Guo et al., 2024; Dwaracherla et al., 2024). We show the performance curves of all variants in Figure 6. The left plot compares variants that directly use their policy for inference. It clearly shows the benefits of learning ERM for active exploration (Variant-2) and aligning π_{θ^t} with $\mathcal{R}_{\Phi^t}$ (Variant-3). Since a reward model is learned within the agent, we can further incorporate inference-time alignment via Best-of-N (BoN) sampling (Nakano et al., 2021; Touvron et al., 2023). This also facilitates a comparison between **SEA** and Dwaracherla et al. (2024), which learns a similar ERM for both exploration and BoN but does not align the LLM policy. Results on the right plot of Figure 6 suggest a similar trend that

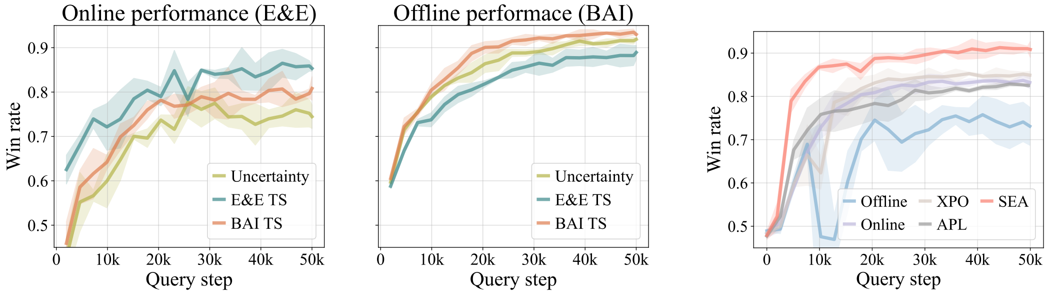


Figure 7: (Left and Middle) Comparison on the win rates of different exploration strategies measured in E&E and BAI settings. **(Right)** Comparison on the win rates of different agents when using `GPT4o-mini` to simulate human feedback via LLM-as-a-judge.

`Variant-6` $\succ$ `Variant-5` $\succ$ `Variant-4`. The `Variant-7`, however, ceases to improve after the ERM converges due to the limited performance of its fixed policy.

6.3 Choice of Exploration Strategies

Recalling that different LLM alignment settings (online system or crowdsourcing) require different exploration strategies to meet their respective learning objectives (Section 3). We investigate three strategies based on posterior sampling and compare them on both online and offline performance. The first strategy focuses on pure exploration. It seeks the pair of dueling responses that exhibits the largest epistemic uncertainty (`Uncertainty`), which is implemented by selecting the pair whose logits difference has the largest variance across ensemble members. The second (`E&E-TS`) and the third (`BAI-TS`) strategies follow the principles of Algorithm 1, and their differences are between Line 5 and Line 6. The comparison results are shown in Figure 7 (Left and Middle). Focusing on the left plot, we observe that `E&E-TS` strategy achieves the best online performance, which is within our expectation. In contrast, `Uncertainty` shows the worst online performance because it tries to maximize the information gain but does not prioritize reward maximization. On the other hand, conclusions are interestingly different when taking the offline performance as a metric. In this case, `BAI-TS` $\succ$ `Uncertainty` both improve the agent's offline performance more efficiently than `E&E-TS`. This can be attributed to that exploration for uncertainty minimizing helps to identify more informative responses to train the LLM policy. `E&E-TS`, however, always chooses two responses with similarly high quality to exploit, and may be less efficient to explore for the optimal policy.

6.4 Aligning LLMs with a Human Simulator

Results presented so far are based on experimenting LLM alignment with the preference oracle being a scalar reward model, which is deterministic and does not capture the potential randomness of the choice by real humans. To test different agents in a more realistic setting, we use generative models as human simulator in an LLM-as-a-judge (Bubeck et al., 2023; Zheng et al., 2023) manner. In particular, we directly query the OpenAI API and use the `gpt-4o-mini-2024-07-18` model as the judge to provide preference feedback. We follow the prompt template of Li et al. (2023). The results are shown in Figure 7 (Right). We can observe the performance curves generally exhibit higher variance, possibly because of the randomness introduced in the feedback process, which puts more stringent requirements for learning algorithms. The two active exploration methods demonstrate opposite results to those in Section 6.1 – `APL` learns fast initially but is eventually outperformed by `Online`, and `XPO` improves over `Online` after stabilizing its training and delivers a better final performance. Our agent, **SEA**, is shown to offer the best sample efficiency as well as asymptotic performance, further validating the importance of online learning and well-designed active exploration mechanism.

7 Conclusion

In this paper, we study the problem of LLM alignment through the lens of contextual dueling bandits and propose an algorithm based on Thompson sampling to align LLMs from preference feedback. Through extensive empirical investigation, we validate the superior sample efficiency of our method compared to existing baselines. To our knowledge, our work is the first to study active exploration for online LLM alignment with true online experimental verification.

REFERENCES

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

Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *Conference on learning theory*, pp. 41–53, 2010.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pp. 4447–4455. PMLR, 2024.

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

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

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

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *Algorithmic Learning Theory: 20th International Conference, ALT 2009, Porto, Portugal, October 3-5, 2009. Proceedings 20*, pp. 23–37. Springer, 2009.

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

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24, 2011.

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

Nirjhar Das, Souradip Chakraborty, Aldo Pacchiano, and Sayak Ray Chowdhury. Provably sample efficient rlhf via active preference optimization. *arXiv preprint arXiv:2402.10500*, 2024.

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

Miroslav Dudík, Katja Hofmann, Robert E Schapire, Aleksandrs Slivkins, and Masrour Zoghi. Contextual dueling bandits. In *Conference on Learning Theory*, pp. 563–587. PMLR, 2015.

Vikranth Dwaracherla, Xiuyuan Lu, Morteza Ibrahimi, Ian Osband, Zheng Wen, and Benjamin Van Roy. Hypermodels for exploration. *arXiv preprint arXiv:2006.07464*, 2020.

Vikranth Dwaracherla, Seyed Mohammad Asghari, Botao Hao, and Benjamin Van Roy. Efficient exploration for llms. In *International Conference on Machine Learning*, 2024.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

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

Javier González, Zhenwen Dai, Andreas Damianou, and Neil D Lawrence. Preferential bayesian optimization. In *International Conference on Machine Learning*, pp. 1282–1291. PMLR, 2017.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*, 2024.

Shengyi Huang, Michael Noukhovitch, Arian Hosseini, Kashif Rasul, Weixun Wang, and Lewis Tunstall. The n+ implementation details of rlhf with ppo: A case study on tl; dr summarization. *arXiv preprint arXiv:2403.17031*, 2024.

Natasha Jaques, Judy Hanwen Shen, Asma Ghandeharioun, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Shane Gu, and Rosalind Picard. Human-centric dialog training via offline reinforcement learning. *arXiv preprint arXiv:2010.05848*, 2020.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise comparison and generative fusion. In *Proceedings of the 61th Annual Meeting of the Association for Computational Linguistics (ACL 2023)*, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. Rewardbench: Evaluating reward models for language modeling, 2024.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 5 2023.

Xuheng Li, Heyang Zhao, and Quanquan Gu. Feel-good thompson sampling for contextual dueling bandits. *arXiv preprint arXiv:2404.06013*, 2024.

Chris Yuhao Liu and Liang Zeng. Skywork reward model series. https://huggingface.co/Skywork, September 2024. URL https://huggingface.co/Skywork.

Viraj Mehta, Vikramjeet Das, Ojash Neopane, Yijia Dai, Ilija Bogunovic, Jeff Schneider, and Willie Neiswanger. Sample efficient reinforcement learning from human feedback via active exploration. 2023.

Luckeciano C Melo, Panagiotis Tigas, Alessandro Abate, and Yarin Gal. Deep bayesian active learning for preference modeling in large language models. *arXiv preprint arXiv:2406.10023*, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

William Muldrew, Peter Hayes, Mingtian Zhang, and David Barber. Active preference learning for large language models. In *International Conference on Machine Learning*, 2024.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

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

Chao Qin, Zheng Wen, Xiuyuan Lu, and Benjamin Van Roy. An analysis of ensemble sampling. *Advances in Neural Information Processing Systems*, 35:21602–21614, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 37, 2023.

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

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3505–3506, 2020.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematics Society*, 58:527–535, 1952.

Daniel Russo. Simple bayesian algorithms for best arm identification. In *Conference on Learning Theory*, pp. 1417–1418. PMLR, 2016.

Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

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

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897. PMLR, 07–09 Jul 2015.

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

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

Richard S Sutton, Michael Bowling, and Patrick M Pilarski. The alberta plan for ai research. *arXiv preprint arXiv:2208.11173*, 2022.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Huasen Wu and Xin Liu. Double thompson sampling for dueling bandits. *Advances in neural information processing systems*, 29, 2016.

Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. *arXiv preprint arXiv:2405.00675*, 2024.

Tengyang Xie, Dylan J Foster, Akshay Krishnamurthy, Corby Rosset, Ahmed Awadallah, and Alexander Rakhlin. Exploratory preference optimization: Harnessing implicit q*-approximation for sample-efficient rlhf. *arXiv preprint arXiv:2405.21046*, 2024.

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

Fan Yang, Gabriel Barth-Maron, Piotr Stańczyk, Matthew Hoffman, Siqi Liu, Manuel Kroiss, Aedan Pope, and Alban Rrustemi. Launchpad: A programming model for distributed machine learning research. *arXiv preprint arXiv:2106.04516*, 2021a.

Keming Yang, Zichen Liu, and Philip Cheng. MOSEC: Model Serving made Efficient in the Cloud, 2021b. URL https://github.com/mosecorg/mosec.

Yisong Yue, Josef Broder, Robert Kleinberg, and Thorsten Joachims. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.

Shenao Zhang, Donghan Yu, Hiteshi Sharma, Ziyi Yang, Shuohang Wang, Hany Hassan, and Zhaoran Wang. Self-exploring language models: Active preference elicitation for online alignment. *arXiv preprint arXiv:2405.19332*, 2024.

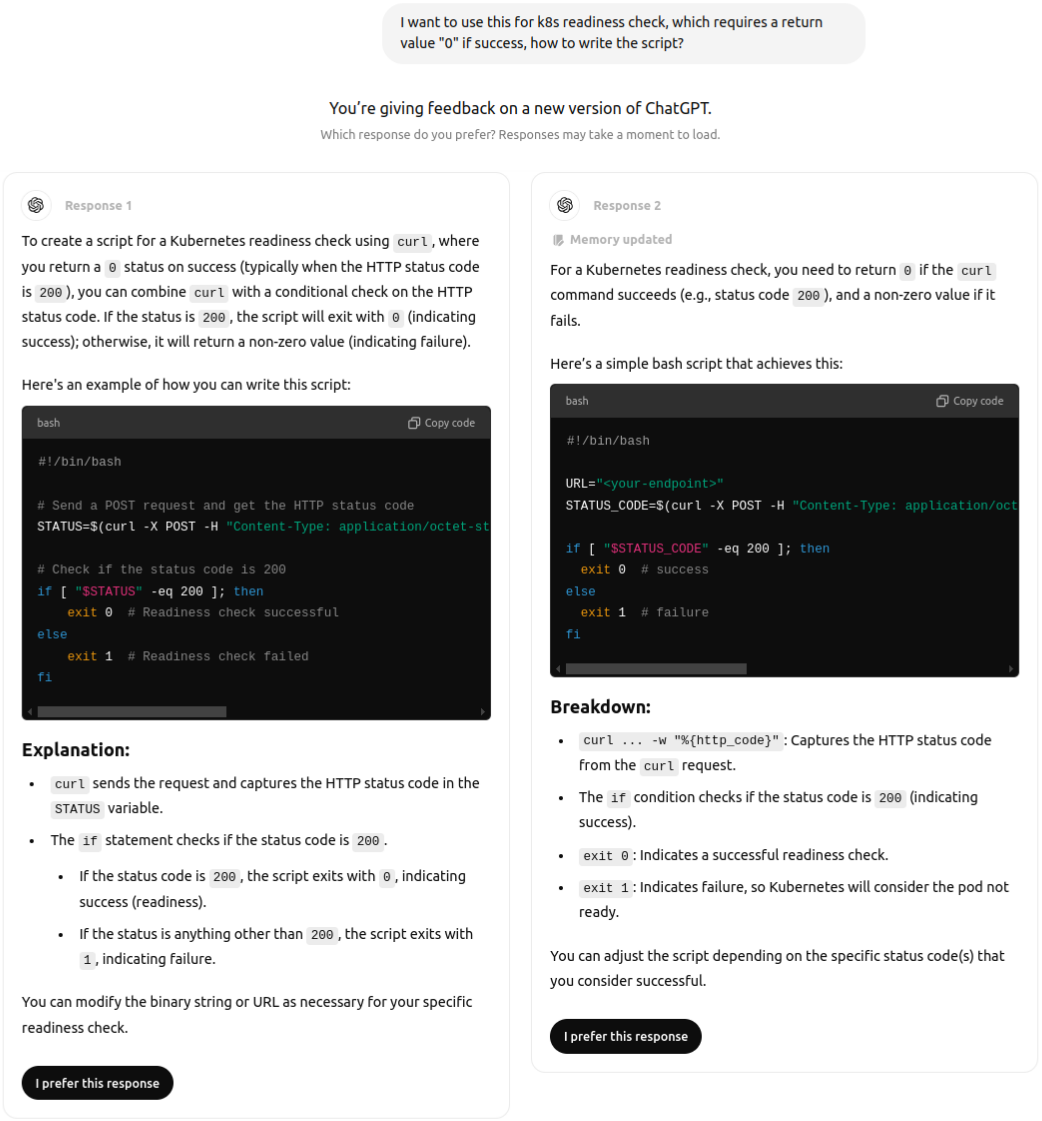
Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

A Supplementary Materials

Table 2: A summary of prior work. π_θ denotes the proposal policy that is continuously updated based on newly collected preference annotations, while π_β denotes a fixed proposal policy. The strategy that encompasses active exploration, online interaction, and π_θ offers the best sample efficiency within the CDB framework. Notably, only the three methods listed at the bottom of the table utilize this strategy, and we include these as baselines in our experiments.

	Method	Exploration		Interaction			Proposal Policy	
		Active	Passive	Online	Iterative	Offline	π_θ	π_β
RL Optimizer	Christiano et al. (2017)		✓		✓	✓	✓	
	Stiennon et al. (2020)		✓		✓	✓	✓	
	Bai et al. (2022)		✓		✓	✓	✓	
	Ouyang et al. (2022)		✓		✓	✓	✓	
Direct Optimizer	Rafailov et al. (2023)		✓			✓	✓	
	Azar et al. (2024)		✓			✓	✓	
	Ethayarajh et al. (2024)		✓			✓	✓	
	Wu et al. (2024)		✓			✓	✓	
	Meng et al. (2024)		✓			✓	✓	
	Dong et al. (2024)		✓		✓		✓	
	Guo et al. (2024)		✓	✓			✓	
	Mehta et al. (2023)	✓		✓				✓
	Das et al. (2024)	✓		✓				✓
	Melo et al. (2024)	✓		✓				✓
	Dwaracherla et al. (2024)	✓		✓				✓
	Zhang et al. (2024)	✓		✓			✓	
	Xie et al. (2024)	✓		✓			✓	
	Muldrew et al. (2024)	✓		✓			✓	



I want to use this for k8s readiness check, which requires a return value "0" if success, how to write the script?

You're giving feedback on a new version of ChatGPT.

Which response do you prefer? Responses may take a moment to load.

To create a script for a Kubernetes readiness check using `curl`, where you return a `0` status on success (typically when the HTTP status code is `200`), you can combine `curl` with a conditional check on the HTTP status code. If the status is `200`, the script will exit with `0` (indicating success); otherwise, it will return a non-zero value (indicating failure).

Here's an example of how you can write this script:

```bash
#!/bin/bash

# Send a POST request and get the HTTP status code
STATUS=$(curl -X POST -H "Content-Type: application/octet-st

# Check if the status code is 200
if [ "$STATUS" -eq 200 ]; then
    exit 0  # Readiness check successful
else
    exit 1  # Readiness check failed
fi
```

Explanation:

- `curl` sends the request and captures the HTTP status code in the `STATUS` variable.
- The `if` statement checks if the status code is `200`.
  - If the status code is `200`, the script exits with `0`, indicating success (readiness).
  - If the status is anything other than `200`, the script exits with `1`, indicating failure.

You can modify the binary string or URL as necessary for your specific readiness check.

For a Kubernetes readiness check, you need to return `0` if the `curl` command succeeds (e.g., status code `200`), and a non-zero value if it fails.

Here's a simple bash script that achieves this:

```bash
#!/bin/bash

URL="<your-endpoint>"
STATUS_CODE=$(curl -X POST -H "Content-Type: application/oct

if [ "$STATUS_CODE" -eq 200 ]; then
  exit 0  # success
else
  exit 1  # failure
fi
```

Breakdown:

- `curl ... -w "%{http_code}"`: Captures the HTTP status code from the `curl` request.
- The `if` condition checks if the status code is `200` (indicating success).
- `exit 0`: Indicates a successful readiness check.
- `exit 1`: Indicates failure, so Kubernetes will consider the pod not ready.

You can adjust the script depending on the specific status code(s) that you consider successful.

Figure 8: ChatGPT system asks for users' preference feedback to strategically explore better answers. In this case, algorithms should be designed around the objective of minimizing cumulative regret, because the quality of the dueling responses generated by the system affects user experience.