Bootstrapping Language Models with DPO Implicit Rewards

Changyu Chen [*1] **Zichen Liu** [*2 3] **Chao Du** [2] **Tianyu Pang** [2] **Qian Liu** [2] **Arunesh Sinha** [4]
Pradeep Varakantham [1] **Min Lin** [2]

Abstract

Human alignment in large language models (LLMs) is an active area of research. A recent groundbreaking work, direct preference optimization (DPO), has greatly simplified the process from past work in reinforcement learning from human feedback (RLHF) by bypassing the reward learning stage in RLHF. DPO, after training, provides an implicit reward model. In this work, we make a novel observation that this implicit reward model can by itself be used in a bootstrapping fashion to further align the LLM. Our approach is to use the rewards from a current LLM model to construct a preference dataset, which is then used in subsequent DPO rounds. We incorporate refinements that debias the length of the responses and improve the quality of the preference dataset to further improve our approach. Our approach, named self-alignment with **D**PO **I**mpli**C**it r**E**wards (DICE), shows great improvements in alignment and achieves superior performance than Gemini Pro on AlpacaEval 2, reaching 27.55% length-controlled win rate against GPT-4 Turbo, but with only 8B parameters and no external feedback. Our code is available at https://github.com/sail-sg/dice.

1. Introduction

Direct preference optimization (DPO) (Rafailov et al., 2024b) presents a compelling alternative to reinforcement learning from human feedback (RLHF) in large language models (LLMs). By circumventing the complexity of learning a reward model from given human preferences, DPO is simpler to implement and train compared to the RLHF approaches. Importantly, DPO, once trained, implicitly specifies a reward model. Mathematically, the reward for a response $\mathbf{y}$ to the prompt $\mathbf{x}$ can be expressed in terms of the optimal policy $\pi^\star$ and the reference policy π_{ref}:

$$\hat{r}(\mathbf{x}, \mathbf{y}) = \beta \log \frac{\pi^\star(\mathbf{y}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})} + \beta \log Z(\mathbf{x}),$$

for parameter β and normalizing constant Z. Further, an *implicit reward* $r(\mathbf{x}, \mathbf{y}) = \beta[\log \pi^\star(\mathbf{y}|\mathbf{x}) - \log \pi_{\text{ref}}(\mathbf{y}|\mathbf{x})]$ is defined in DPO where the normalizing constant term can be ignored as it will be canceled out in the DPO objective, which only involves the difference of the rewards for the same prompt. In this work, we explore whether the above readily available implicit reward model after DPO training provides an opportunity to further improve the language model.

This paper answers the research question in the affirmative, by using the above implicit rewards in a *bootstrapping* fashion to further improve the LLM alignment with human preferences. Specifically, our approach follows the iterative DPO framework (Tran et al., 2023), where the implicit rewards are provided as the preference signals, as illustrated in Figure 1. We start with a model that has been through one round of DPO using human preference data. We then use the implicit rewards induced by itself to rank outputs from the current LLM, thereby, yielding a new preference dataset cheaply. We run DPO again with this newly generated preference dataset to obtain an updated LLM and then repeat the process. However, the above approach still needs further refinement to address practical issues. One is the known issue of length exploitation (Park et al., 2024) where LLMs generate long responses when the same content could be provided more succinctly. Another issue is that the implicit reward model is an approximate proxy for human preferences, hence relying on it strongly can result in corruption of the initial knowledge inbuilt into the LLM.

We address the length exploitation issue by length-regularized reward shaping, which discourages long responses from being preferred. To fix the overreliance on the implicit reward we use insights from continual learning (Rolnick et al., 2019) and replay high quality human preference data that was used in the first round of DPO (this round was before we start our bootstrapping).

[*]Equal contribution. Work done during Changyu Chen's internship at Sea AI Lab. [1]Singapore Management University [2]Sea AI Lab, Singapore [3]National University of Singapore [4]Rutgers University. Correspondence to: Chao Du <duchao@sea.com>, Arunesh Sinha <arunesh.sinha@rutgers.edu>, Pradeep Varakantham <pradeepv@smu.edu.sg>.

ICML 2024 Workshop on Models of Human Feedback for AI Alignment, Vienna, Austria. Copyright 2024 by the author(s).

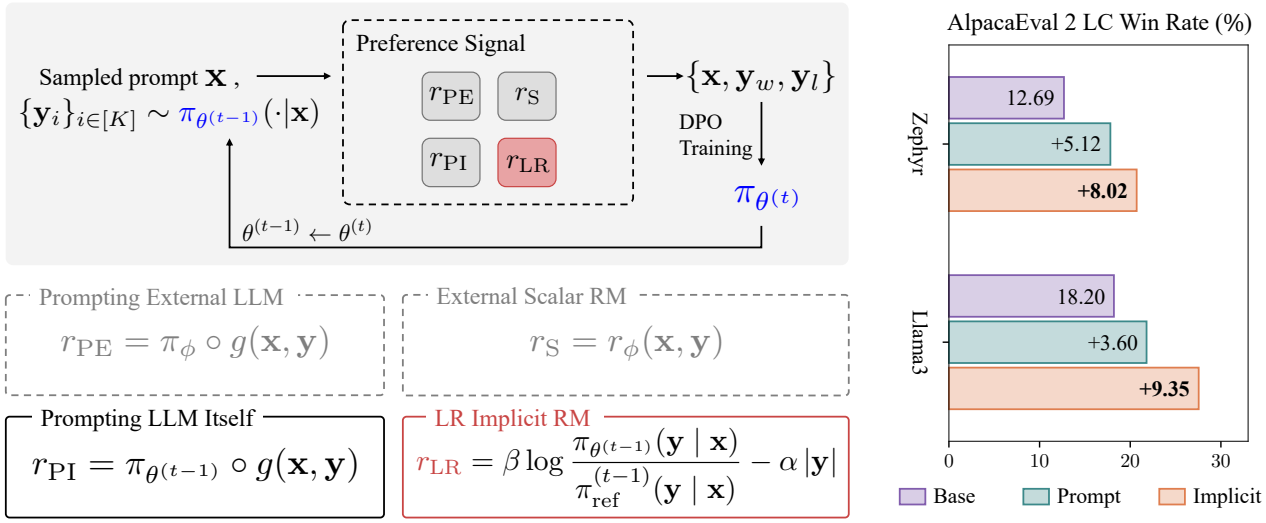

Figure 1: **(Left)** Iterative DPO (Tran et al., 2023) with various preference signals: under the iterative DPO framework, the policy model is iteratively trained on a newly generated preference dataset. This dataset can be constructed using various preference signals. A common source is a scalar reward model (RM) (Ouyang et al., 2022), denoted as r_ϕ. Alternatively, the dataset can be created by prompting a LLM to judge the responses. This LLM can either be an external model π_ϕ or the policy model itself from the previous iteration $\pi_{\theta^{(t-1)}}$. In this context, $\mathbf{x}$ and $\mathbf{y}$ are processed through a LLM-as-a-judge template $g(\cdot,\cdot)$ (Yuan et al., 2024). We propose utilizing the length-regularized (LR) implicit rewards introduced in Section 4.2, where $\pi_{\theta^{(t-1)}}$ and $\pi_{\mathrm{ref}}^{(t-1)}$ represent the policy model and reference model from the previous DPO iteration, respectively. Our experiment excludes approaches requiring external models, such as r_ϕ and π_ϕ, as they are beyond this work's scope. **(Right)** Our method, which leverages implicit rewards, further improves DPO-tuned models by a large margin, resulting in superior performance compared to the prompting counterpart.

Our method, named self-alignment with **D**PO **I**mpli**C**it r**E**wards (DICE), significantly improves LLM alignment quality with different base models. On AlpacaEval 2, we achieve 8.02% length-controlled win rate improvement with the Zephyr-based model and 9.35% improvement with the Llama3-based model. Our best model outperforms Gemini Pro, but with only 8B parameters and does not require any in-house data or external reward model.

To summarize, our main contributions are as follows:

- We propose to utilize the implicit reward model readily available in a DPO-ed LLM. The implicit reward model enables us to evaluate the responses generated by the current policy and construct a preference dataset for future rounds of DPO without any external feedback;
- We propose to apply two techniques together with our above proposed approach, length-regularized reward shaping and experience replay;
- Empirical results show that our approach DICE enables significant improvement in alignment with different base models, and our best bootstrapped model achieves better performance than Gemini Pro but with only 8B parameters.

2. Related Work

Self-Improving Fine-Tuning. Many efforts have been made to investigate ways of fine-tuning language models without a large amount of human annotation (Huang et al., 2022; Li et al., 2023a; Sun et al., 2023; 2024; Yuan et al., 2024). Starting from an SFT model, Sun et al. (2023) collect the preference labels by prompting the SFT model itself to choose the preferred response given a principle and train a principle-driven reward model with these preference labels. Afterwards, they optimize the policy by PPO with their reward model. Yuan et al. (2024) construct the preference dataset by their own supervised fine-tuned model trained on instruction following data and evaluation fine-tuning data, followed by DPO training on the preference dataset. Our work differs from this line of work in the motivations and assumptions. This line of work aims to align a language model with a small amount of seed data by eliciting the internal knowledge that is learned during the pretraining phase of LLMs. In contrast, our goal is to further improve a DPO-ed model in a bootstrapping manner by utilizing its implicit rewards.

On-Policy Sampling in Preference Tuning. DPO and its variants are popular due to their simplicity in training and implementation. However, research indicates that the offline

nature of these direct alignment from preference (DAP) algorithms often prevents them from learning a good policy (Guo et al., 2024). Guo et al. (2024) have shown that the offline DPO will quickly overfit the preference dataset, while it performs much better and is more stable if online feedback can be provided to their on-policy samples[1]. Tajwar et al. (2024) systematically discussed the properties of different preference fine-tuning approaches and the role of on-policy sampling. Their actionable takeaways for practitioners tell on-policy sampling generally improves performance and efficiency. If not being able to perform the pure on-policy sampling, using the data that is closer to on-policy samples also helps as it can be seen in iterative DPO (Tran et al., 2023) and self-rewarding language models (Yuan et al., 2024). Similarly, our approach enables us to train the policy model with the preference data which is closer to on-policy samples than the offline dataset without any external reward models. We hypothesize this is one of the main gain sources of our approach.

DPO Implicit Rewards. Rafailov et al. (2024a) recently study DPO from a token-level MDP perspective, and reveal that DPO-ed models implicitly parameterize token-wise dense reward functions. They therefore conduct beam search using the implicit rewards to improve the inference quality. Our work is inspired by this observation but focuses on utilizing the DPO-ed model's implicit rewards to bootstrap itself for self-alignment. Another independent work to ours (Zhong et al., 2024) pretrains a DPO model to serve as a standalone dense reward generator for PPO training, instead of further improving the DPO model.

3. Preliminaries

We provide a brief review of the standard RLHF (Ouyang et al., 2022) and DPO algorithms (Rafailov et al., 2024b). Through the review, we demonstrate the implicit reward model induced by DPO, which will be used in our work.

In preference tuning, the preference data typically takes the form of pairwise preferences. Each prompt $\mathbf{x}$ is paired with two possible responses, $\mathbf{y}_1$ and $\mathbf{y}_2$. The human annotator (Christiano et al., 2017) or AI annotator (Lee et al., 2023) provides the preference feedback $o(\mathbf{y}_1 \succ \mathbf{y}_2|\mathbf{x}) \in \{0, 1\}$, indicating whether $\mathbf{y}_1$ is preferred over $\mathbf{y}_2$. The preferred response is denoted as $\mathbf{y}_w$, while the other is denoted as $\mathbf{y}_l$. A common assumption is that the ground-truth human preferences follow the Bradley-Terry model (Bradley & Terry, 1952). Based on this assumption, we can train a parameterized reward model $r_\phi(\mathbf{x}, \mathbf{y})$ using maximum likelihood:

$$\begin{aligned}&\mathcal{L}_R\left(r_\phi, \mathcal{D}\right) = \\ &-\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}}\left[\log\sigma\left(r_\phi\left(\mathbf{x},\mathbf{y}_w\right) - r_\phi\left(\mathbf{x},\mathbf{y}_l\right)\right)\right],\end{aligned} \tag{1}$$

where σ is the logistic function.

3.1. Reinforcement Learning from Human Feedback

The standard RLHF algorithm treats the LLM as a policy and optimizes the policy using the reward model r_ϕ. The optimization objective is represented by the following equation:

$$\begin{aligned}\max_{\pi_\theta}&\mathbb{E}_{\mathbf{x}\sim\mathcal{D},\mathbf{y}\sim\pi_\theta(\mathbf{y}|\mathbf{x})}\left[r_\phi(\mathbf{x},\mathbf{y})\right] \\ &-\beta\cdot\mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(\mathbf{y}|\mathbf{x})\|\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})\right],\end{aligned} \tag{2}$$

where $\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})$ denotes a reference distribution, and β is a hyper-parameter. This objective balances the maximization of the reward $r_\phi(\mathbf{x}, \mathbf{y})$ and divergence from the fixed reference distribution. The divergence term, given by the KL divergence (i.e., $\mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(\mathbf{y}|\mathbf{x})\|\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})\right]$) acts as a regularizer to prevent the policy π_θ from drifting too far away from the initial distribution $\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})$. This objective is then optimized using a general-purpose RL algorithm, such as PPO (Schulman et al., 2017).

3.2. Direct Preference Optimization

DPO (Rafailov et al., 2024b) starts with the same objective as Section 3.1. However, DPO derives an analytical closed-form relation between the reward and the resulting optimal policy. This relation can be used to reparameterize the ground truth reward in terms of the corresponding optimal policy. This reparameterized formulation can be substituted back into the reward optimization objective in Section 3, enabling direct training of the optimal model on the feedback data using the following objective:

$$\begin{aligned}&\mathcal{L}_{\mathrm{DPO}}\left(\pi_\theta;\pi_{\text{ref}}\right) = \\ &-\mathbb{E}_{(\mathbf{x},\mathbf{y}_w,\mathbf{y}_l)\sim\mathcal{D}}\Big[\log\sigma\Big(\beta\log\frac{\pi_\theta\left(\mathbf{y}_w|\mathbf{x}\right)}{\pi_{\text{ref}}\left(\mathbf{y}_w|\mathbf{x}\right)} \\ &\qquad -\beta\log\frac{\pi_\theta\left(\mathbf{y}_l|\mathbf{x}\right)}{\pi_{\text{ref}}\left(\mathbf{y}_l|\mathbf{x}\right)}\Big)\Big].\end{aligned} \tag{3}$$

In this context, the parameter β is the same as in Section 3.1, balancing the expected reward and divergence from the initial model. The DPO objective is particularly advantageous as it facilitates the recovery of the optimal model through a standard classification loss, without the need for on-policy sampling or extensive hyper-parameter tuning. Observe that Section 3.2 resembles the reward modeling objective in Section 3 under the parameterization

$$r(\mathbf{x},\mathbf{y}) = \beta\log\frac{\pi_\theta(\mathbf{y}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})}. \tag{4}$$

[1]On-policy samples: the data collected while following the current policy that is being optimized.

This reward function is commonly referred to as an "implicit reward" (Rafailov et al., 2024a; Zhong et al., 2024). Theorem 1 in Rafailov et al. (2024b) demonstrates that this parameterization of a reward model is indeed valid without loss of generality. If we substitute this form of $r_\theta(\mathbf{x}, \mathbf{y})$ into the RL objective in Section 3.1, we can derive the optimal solution in a closed form, which is π_θ. Consequently, once DPO optimization is completed, we obtain an "implicit reward model" as defined by Eq. (4).

4. Bootstrapping with DPO Implicit Rewards

DPO is an attractive alternative to RLHF as it largely simplifies the implementation and training process of language model alignment. However, recent evidences (Guo et al., 2024; Tran et al., 2023) have shown that continuing DPO training on a fixed offline dataset results in inferior policy, while the policy can be further improved if one can collect new responses generated by the updated policy and provide preference labels to perform another round of DPO training. This can be understood as being closer to the on-policy sampling, which is generally preferred in preference fine-tuning (Tajwar et al., 2024; Tang et al., 2024). In Section 4.1, we first outline the iterative DPO training framework adopted in this work, and provide a theoretical analysis similar to Xie et al. (2024) to show that learning with on-policy samples can be more effective than utilizing an offline dataset. Next, in Section 4.2, we introduce the proposed Length-Regularized Implicit Rewards, which augment the vanilla implicit rewards with a length-regularized reward shaping, to judge the on-policy sampled responses for constructing the preference dataset. Notably, by fine-tuning a DPO-ed LM on the constructed dataset, we essentially align it without relying on any external preference feedback (e.g., RLHF or RLAIF), hence in a *bootstrapping* fashion. Furthermore, to mitigate the potential catastrophic forgetting in the continual fine-tuning, we propose experience replay (Section 4.3) that mixes the generated data with the offline data for better performance. We refer to our method as iterative self-alignment (bootstrapping) with **D**PO **I**mpli**C**it r**E**wards (DICE).

4.1. Iterative DPO with On-policy Sampling

We employ the iterative DPO preference tuning framework, where we start from a base language model (a base policy) $\pi_{\theta^{(0)}}$ that is DPO-tuned from an initial reference model $\pi_{\text{ref}}^{(0)}$, commonly an SFT model. In each round $t \in \{1, 2, \dots\}$, we sample K on-policy responses from the latest policy $\pi_{\theta^{(t-1)}}(\cdot \mid \mathbf{x})$ given a prompt $\mathbf{x}$. We then label the response with the highest and the lowest implicit rewards as winning and losing responses respectively, thus constructing a new preference dataset $\mathcal{D}_t$. We further fine-tune the policy with DPO's objective (Section 3.2) to obtain the updated policy $\pi_{\theta^{(t)}}$ with reference model $\pi_{\text{ref}}^{(t)} = \pi_{\theta^{(t-1)}}$. This process is repeated to iteratively improve the language model.

To investigate the effect of on-policy sampling, we make a generalized notation for the sampling policy at the t-th round as $\pi^{(t)}$, and compare the on-policy sampling ($\pi^{(t)} = \pi_{\theta^{(t-1)}}$) with sampling from an offline dataset ($\pi^{(t)} = \pi_\mu$). For a prompt $\mathbf{x}$, we denote its optimal response as $\mathbf{y}^\star$ and suboptimal ones as $\mathcal{S} = \{\mathbf{y}_i^-\}$. We also denote its preferences sampled from $\pi^{(t)}$ as $(\mathbf{y}_w^{(t)}, \mathbf{y}_l^{(t)})$. By Section 3.2 and the definition of the logistic function, we can rewrite the DPO loss at round t for the sample $(\mathbf{x}, \mathbf{y}_w^{(t)}, \mathbf{y}_l^{(t)})$ as:

$$\mathcal{L}_{\text{DPO}}^{(t)}(\pi_{\theta^{(t)}}; \pi_{\text{ref}}^{(t)}) = -\log \frac{\pi_{\theta^{(t)}}\left(\mathbf{y}_w^{(t)} \mid \mathbf{x}\right)^\beta}{\pi_{\theta^{(t)}}\left(\mathbf{y}_w^{(t)} \mid \mathbf{x}\right)^\beta + \pi_{\theta^{(t)}}\left(\mathbf{y}_l^{(t)} \mid \mathbf{x}\right)^\beta R^\beta}, \tag{5}$$

where $R = \pi_{\text{ref}}^{(t)}(\mathbf{y}_w^{(t)} \mid \mathbf{x}) / \pi_{\text{ref}}^{(t)}(\mathbf{y}_l^{(t)} \mid \mathbf{x})$ is a constant. Observe that Section 4.1 can be minimized to zero by just minimizing $\pi_{\theta^{(t)}}(\mathbf{y}_l^{(t)} | \mathbf{x})$ to be zero, without minimizing the likelihood of other suboptimal responses $\pi_{\theta^{(t)}}(\mathbf{y}^- | \mathbf{x})$ for $\mathbf{y}^- \neq \mathbf{y}_l^{(t)}$. After t rounds of optimization, we are interested in the probability of outputting the optimal response:

$$\pi_{\theta^{(t)}}(\mathbf{y}^\star \mid \mathbf{x}) = 1 - \sum_{\mathbf{y}_i^- \in \mathcal{S}} \pi_{\theta^{(t)}}(\mathbf{y}_i^- \mid \mathbf{x}). \tag{6}$$

Eq. (6) allows us to reveal the deficiency of training on a fixed offline dataset. If there exists a suboptimal response $\mathbf{y}^- \in \mathcal{S}$ that lies in the high likelihood region of $\pi_{\theta^{(t)}}$, say $\pi_{\theta^{(t)}}(\mathbf{y}^- \mid \mathbf{x}) \geq p$ for some $p \in [0, 1]$ that is close to 1, and $\mathbf{y}^-$ is never sampled from $\pi^{(t)} = \pi_\mu$ thus not optimized as $\mathbf{y}_l^{(t)}$ during all t rounds, we have:

$$\pi_{\theta^{(t)}}(\mathbf{y}^\star \mid \mathbf{x}) \leq 1 - \pi_{\theta^{(t)}}(\mathbf{y}^- \mid \mathbf{x}) \leq 1 - p. \tag{7}$$

Since π_μ is zero except points that appear in $\mathcal{D}_{\text{offline}}$, it is highly likely to find such a $\mathbf{y}^-$ not being sampled from π_μ (hence never optimized) and therefore $\pi_{\theta^{(t)}}(\mathbf{y}^\star \mid \mathbf{x})$ can be very low with a large p.

On the other hand, conducting on-policy sampling can alleviate the "never-sampled" issue and promote convergence to the optimal policy. This is because whenever $\pi_{\theta^{(t-1)}}(\mathbf{y}_i^- \mid \mathbf{x})$ is high, it is likely to sample $\mathbf{y}_i^-$ from $\pi^{(t)} = \pi_{\theta^{(t-1)}}$ and thus it can be optimized such that $\pi_{\theta^{(t)}}\left(\mathbf{y}_l^{(t)} = \mathbf{y}_i^- \mid \mathbf{x}\right) \approx 0$. In this way, the subtrahend of Eq. (6) is decreased per round, hence we can gradually improve the language model policy towards the optimal policy.

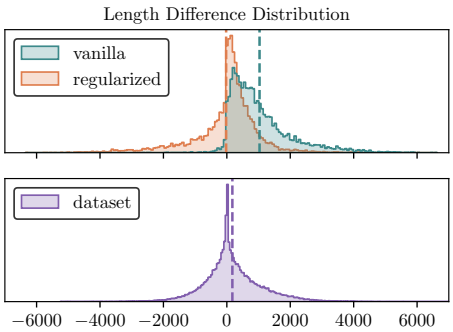

Figure 2: Distribution of the length difference between the winning and losing examples ($|\mathbf{y}_w|-|\mathbf{y}_l|$). **(Top)** Distribution of the first round on-policy generated dataset. With LR reward shaping defined by Eq. (8), the length bias is mitigated and the length difference becomes more evenly distributed. The average length difference decreases from 1031 to -21 by setting α=0.023. **(Bottom)** Distribution of the high quality UltraFeedback preference dataset (Cui et al., 2023) is almost unbiased.

4.2. Length-Regularized Implicit Rewards

It is a known issue in the literature that preference tuning may introduce length bias (or length exploitation) (Park et al., 2024) , which is likely caused by the fact that the preference labels collected from human annotators favor more verbose responses. This problem is further compounded by the iterative self-alignment scheme such as the one in Yuan et al. (2024), because the generated responses that are long and preferred will be reinforced in the next round of DPO, leading the language model to generate increasingly longer responses.

Inevitably, the vanilla DPO implicit rewards as in Eq. (4) would also exhibit length bias when generating preference dataset. In Figure 2, we show the distribution of the difference in string length of the winning and losing responses. We can see from the top figure that vanilla implicit rewards yield a skewed distribution (in green), with an average length difference 1031. In stark contrast, the length difference of a high quality preference dataset is almost normally distributed (as in the bottom figure). This observation motivates us to debias the distribution induced by vanilla implicit rewards so as to mitigate the length exploitation. We resort to reward shaping (Sutton & Barto, 2018) for this purpose. In particular, we introduce a length-regularized (LR) reward shaping term in the implicit reward that penalizes the length of the response to obtain the shaped reward:

$$r_{\mathrm{LR}}(\mathbf{x}, \mathbf{y}; \alpha) = \beta \log \frac{\pi_\theta(\mathbf{y} \mid \mathbf{x})}{\pi_{\mathrm{ref}}(\mathbf{y} \mid \mathbf{x})} - \alpha\,|\mathbf{y}|\,, \tag{8}$$

where α controls the penalty strength and $|\mathbf{y}|$ is the string length of the response. Based on the shaped rewards, we can construct many versions of the preference dataset $\mathcal{D}(\alpha)$, following the principle that the response with the higest $r_{\mathrm{LR}}(\mathbf{x}, \mathbf{y}_i; \alpha)$ is labeled as $\mathbf{y}_w$ and the one with the lowest reward is labeled as $\mathbf{y}_l$. To find the most suitable α such that $\mathcal{D}(\alpha)$ is (approximately) unbiased, we optimize α with the objective to minimize the average absolute difference in response length:

$$\alpha^\star = \arg\min_\alpha \mathbb{E}_{(\mathbf{y}_w, \mathbf{y}_l)\sim\mathcal{D}(\alpha)} \left(\Big| |\mathbf{y}_w| - |\mathbf{y}_l| \Big| \right). \tag{9}$$

We can employ any black-box optimizer to solve this non-differentiable objective function. In this work we find a simple random search suffices, and its solution effectively transforms the dataset into a more evenly distributed one (shown in the orange curve in the top of Figure 2). We will output $\mathcal{D}(\alpha^\star)$ for the next round of DPO training. Details of the optimization can be found in Appendix B.

Importantly, despite the resemblance of Eq. (8) to Park et al. (2024) where they incorporate the token length as a regularizer in the training objective, our reward shaping is conducted during the dataset construction stage, thereby avoiding the need for expensive hyper-parameter tuning.

4.3. Experience Replay

Though DICE enables us to learn from the response of the current policy, we know that the implicit reward model from the DPO training is not a perfect proxy for human preferences. Solely relying on the implicit reward model may result in forgetting the knowledge inbuilt in the initial policy at the first DPO stage. Inspired by the technique of experience replay (Rolnick et al., 2019) in continual learning for preventing catastrophic forgetting and making a good balance between old and new data, we propose to use a mixture of the generated data and the offline preference dataset. While offline preference data is considered to be high-quality, it is off-policy samples; the generated data is closer to the on-policy samples, but the imperfect implicit reward model may introduce noise in labeling the preference. Combining the two can make for a good balance.

Our complete algorithm is summarized in Algorithm 1. During each iteration t, we generate K responses $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K$ from the policy $\pi_{\theta^{(t-1)}}$ for each prompt $\mathbf{x}$ (Line 3). The preference dataset $\mathcal{D}(\alpha)$ is constructed based on the LR rewards r_{LR}, where the response with the highest LR reward is labeled $\mathbf{y}_w$ and the one with the lowest reward is labeled $\mathbf{y}_l$. In this process, $\pi_{\theta^{(t-1)}}$ serves as the target policy, and $\pi_{\mathrm{ref}}^{(t-1)}$ serves as the reference policy. At Line 4, we construct the debiased dataset $\mathcal{D}(\alpha^\star)$ by minimizing the average absolute difference in response length, referring to Eq. (9). Subsequently, a mixed dataset $\mathcal{D}_t$ is created by sampling γ proportion of data from $\mathcal{D}_{\mathrm{offline}}$ and $(1-\gamma)$

Algorithm 1 Bootstrapping with DPO Implicit Rewards (DICE)

1: **Input:** prompt set $\mathcal{X}$ extracted from $\mathcal{D}_{\text{offline}}$, initial DPO-tuned policy $\pi_{\theta^{(0)}}$, initial reference policy $\pi_{\text{ref}}^{(0)}$, number of generated samples K, regularization weight β, experience replay weight $\gamma \in (0, 1)$
2: **for** $t = 1, 2, \ldots$ **do**
3: Generate responses by sampling $\mathbf{x} \sim \mathcal{X}$ and $\mathbf{y}_{1:K} \sim \pi_{\theta^{(t-1)}}$;
4: Create preference dataset $\mathcal{D}(\alpha^\star)$ by optimizing Eq. (9);
// $\mathcal{D}(\alpha)$ is constructed based on LR rewards $r_{\text{LR}}(\mathbf{x}, \mathbf{y}_k; \alpha)$, $k \in [K]$;
// Evaluate $r_{\text{LR}}(\mathbf{x}, \mathbf{y}_k; \alpha)$ given $\pi_{\theta^{(t-1)}}$ and $\pi_{\text{ref}}^{(t-1)}$ as the target policy and reference policy
5: Create the mixed dataset via experience replay $\mathcal{D}_t = \{(\mathbf{x}^i, \mathbf{y}_w^i, \mathbf{y}_l^i)\}_{i \in [N]}$;
// Sample $(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l) \sim p_{\mathcal{D}_t}$, $p_{\mathcal{D}_t} = (1-\gamma) p_{\mathcal{D}(\alpha^\star)} + \gamma p_{\mathcal{D}_{\text{offline}}}$
6: Optimize π_θ according to DPO loss, Section 3.2:

$$\theta^{(t)} \leftarrow \arg\min_{\theta} -\mathbb{E}_{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l) \sim \mathcal{D}_t} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(\mathbf{y}_w|\mathbf{x})}{\pi_{\theta^{(t-1)}}(\mathbf{y}_w|\mathbf{x})} - \beta \log \frac{\pi_\theta(\mathbf{y}_l|\mathbf{x})}{\pi_{\theta^{(t-1)}}(\mathbf{y}_l|\mathbf{x})} \right) \right];$$

7: Assign $\pi_{\text{ref}}^{(t)} \leftarrow \pi_{\theta^{(t-1)}}$;
8: **end for**

proportion of data from $\mathcal{D}(\alpha^\star)$ (Line 5). DPO training is then conducted on $\mathcal{D}_t$ using $\pi_{\theta^{(t-1)}}$ as both the initial policy and the reference policy, resulting in the updated policy $\pi_{\theta^{(t)}}$ (Line 6). Finally, at Line 7, we set $\pi_{\theta^{(t-1)}}$ as the new reference policy $\pi_{\text{ref}}^{(t)}$.

5. Experiments

This section empirically investigates DICE. Our findings highlight several key points: (1) DICE significantly improves the model performance on the widely used leaderboard AlpacaEval 2.0 (Li et al., 2023b), increasing length-controlled win rate by up to 9.35%; (2) our best model bootstrapped from a 8B base model (`Llama-3-8B-DPO`) achieves a better performance than Gemini Pro without any extra human annotations (or external reward model) other than the preference dataset that is used in the initial DPO training of base model; (3) the two proposed techniques in Sections 4.2 and 4.3 are shown to be critical for DICE.

5.1. Experiment Setup

Base Models and Datasets. In this paper, we adopt `Llama-3-8B-DPO`[2] and `zephyr-7B-beta`[3] as our base models. Both models are trained following the pipeline of Zephyr (Tunstall et al., 2023). `Llama-3-8B-DPO` is trained based on meta-llama/Meta-Llama-3-8B[4], developed by Meng et al. (2024). `zephyr-7B-beta` is fine-tuned based on mistralai/Mistral-7B-v0.1[5]. We adopt UltraFeedback preference dataset (Cui et al., 2023) and randomly sample a subset of around 10k preference pairs as the offline dataset $\mathcal{D}_{\text{offline}}$. Our experiment aims to show how much the language model can improve from a DPO-ed model and a subset of the preference dataset that was used to conduct the initial DPO training.

Response Generation and Dataset Construction. At the beginning of each round, we first sample responses from the current policy, with temperature $T = 0.9$, $p = 1.0$ for the Llama3 setting and $T = 0.7$, $p = 0.9$ for the Zephyr setting. We sample with different random seeds to get $K = 16$ diverse responses for each prompt. We then reward each prompt-response pair by the implicit reward model (Eq. (4)) and incorporate length-regularized reward shaping (Eq. (8)) to get the debiased dataset $\mathcal{D}(\alpha^\star)$ with the optimal regularization strength $\alpha^\star$. The final dataset is a mixture of $\mathcal{D}(\alpha^\star)$ and $\mathcal{D}_{\text{offline}}$.

Training Details. All experiments are conducted on 8 Nvidia A100 GPUs. For DICE, we trained two rounds in total. In each round, we train the model for 300 steps on a preference dataset with 9.6k preference pairs (either a solely generated dataset, or a mixture of the generated dataset and the offline preference dataset). The global training batch size is set to 32 and the learning rate is 5e-7 with a constant schedule and a warm-up of 50 steps. We hypertune β for each method and report their best performance. For our approach, we additionally hypertune the experience replay ratio γ.

Baselines. We evaluate the following baseline methods that are applicable to the setting of this paper:

[2] https://huggingface.co/princeton-nlp/Llama-3-Base-8B-SFT-DPO

[3] https://huggingface.co/HuggingFaceH4/zephyr-7b-beta

[4] https://huggingface.co/meta-llama/Meta-Llama-3-8B

[5] https://huggingface.co/mistralai/Mistral-7B-v0.1

Table 1: AlpacaEval 2.0 evaluation of various models (detailed in Baselines in Section 5.1) in terms of both normal win rates (WR) and length-controlled win rates (LC) in percentage (%).

Method	zephyr-7B-beta			Llama-3-8B-DPO		
	LC	WR	Avg. Len.	LC	WR	Avg. Len.
Base	12.69	10.71	1444	18.20*	15.50*	-
Offline DPO Iter 1	13.40	11.10	1544	20.22	18.33	1750
Offline DPO Iter 2	4.96	5.47	2178	21.04	19.21	1761
Offline DPO (w/ new ref) Iter 1	13.40	11.10	1544	22.29	19.96	1775
Offline DPO (w/ new ref) Iter 2	4.58	5.27	2296	22.50	20.18	1737
LLM-as-a-Judge Iter 1	17.81	15.36	2572	20.30	21.31	2029
LLM-as-a-Judge Iter 2	14.14	17.89	2916	21.80	22.42	2051
DICE Iter 1	19.03	17.67	1848	25.08	25.77	2201
DICE Iter 2	**20.71**	**20.16**	1879	**27.55**	**30.99**	2427

* We note that the results of Llama-3-8B-DPO base are obtained from Meng et al. (2024).

- Offline DPO: continue conducting DPO training with the offline preference dataset.
- Offline DPO w/ new ref: similar to offline DPO but we assign the current policy as the new reference model, while we use a fixed reference model in Offline DPO. This corresponds to $\gamma = 1$.
- DPO with prompted rewards: similar to Self-rewarding LM (Yuan et al., 2024), where they prompt the LLM itself as a preference judge to construct new preference pairs and iteratively fine-tune the LLM with the DPO algorithm. In their case, the judge capability is learned by supervised fine-tuning on an evaluation fine-tuning dataset. We exploit our base model to perform LLM-as-a-Judge directly as it can follow the judge instructions well. In the experiment, we call it *LLM-as-a-Judge*. The LLM-as-a-Judge prompt template can be found in Appendix A.

Evaluation. We evaluate our method by AlpacaEval 2.0 (Li et al., 2023b). AlpacaEval 2.0 is an LLM-based automatic evaluation benchmark. It employs AlpacaFarm (Dubois et al., 2023) as its prompts set composed of general human instructions. The model responses and the reference responses generated by GPT-4-Turbo are fed into a GPT-4-Turbo-based annotator to be judged. We follow the standard approach and report both the win rate (WR) and the Length-Controlled win rate (LC) (Dubois et al., 2024) over the reference responses.

5.2. Main Results

DICE Effectively Improves a DPO-ed Model. With only a DPO-ed model and a preference dataset that was used to train this model, one can choose to further improve the current policy via Offline DPO, or construct a new preference dataset via LLM-as-a-Judge. In Table 1, we compare the performance of the model fine-tuned by DICE in two rounds with the base model and other baselines. It shows all methods can improve the LC win rate on AlpacaEval 2.0 over the base model while DICE leads to the most significant enhancement in both Zephyr and Llama3 settings, increasing by **8.02**% and **9.35**% respectively. We found that the LLM-as-a-Judge leads to good performance in the Zephyr setting while it has minor improvement in the Llama3 setting. We hypothesize this may be caused by the coarse rewards which are not able to provide effective preference signals when responses are of high quality (the prompt template requires LLM judge to provide a discrete score from 0 to 5, referring to Appendix A). We note that training on a fixed offline dataset for multiple rounds leads to even worse performance than the base model, possibly due to the increased data staleness and overfitting.

Compare with the models on the leaderboard. Compared

Table 2: AlpacaEval 2.0 leaderboard results.

Model	LC	WR
GPT-4 0613	30.18	15.76
Mistral Medium	28.61	21.86
Claude 2	28.15	17.19
DICE-Llama3 8B Iter 2	27.55	30.99
Snorkel (Mistral-PairRM-DPO)	26.39	30.22
Gemini Pro	24.38	18.18
Mixtral 8×7B v0.1	23.69	18.26
Llama 3 8B Instruct	22.92	22.57
GPT-3.5 Turbo 0613	22.35	14.10
Tulu 2+DPO 70B	21.24	15.98
DICE-Zephyr 7B Iter 2	20.71	20.16
GPT-3.5 Turbo 1106	19.30	9.18
Llama-3-8B-DPO	18.20	15.50
Vicuna 33B v1.3	17.58	12.71
zephyr-7B-beta	12.69	10.71

Table 3: DPO rewards are compatible with other direct preference optimization algorithms.

Method	LC	WR	Avg. Len.
Offline DPO	13.40	11.10	1544
DICE-IPO	18.51	19.49	2729
DICE-KTO	14.88	12.16	1541
DICE-Hinge	15.92	15.57	1970
DICE-DPO	19.03	17.67	1848

Table 4: Effects of the Length Regularized Weight.

γ	α	LC	WR	Avg. Len.
0.0	0.000	13.32	15.37	2570
	0.023 ($\alpha^\star$)	18.88	19.31	2109
	0.046 ($2\alpha^\star$)	14.40	9.08	876
0.5	0.000	15.92	19.08	2600
	0.023 ($\alpha^\star$)	19.03	17.67	1848
	0.046 ($2\alpha^\star$)	14.91	10.80	1185

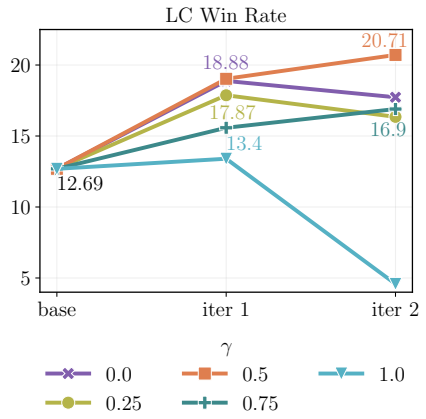

Figure 3: AlpacaEval 2.0 LC Win Rate across different experience replay ratio (γ) in the Zephyr setting. The highest LC Win Rate is reported via text.

with the models on the public leaderboard shown in Table 2, DICE-Llama3 8B performs better than the official instruct version of Llama3 by a non-trivial margin, 4.63%. Regarding the closed-source models, it achieves better performance than Gemini Pro with only 8B parameters and does not require any in-house data or external reward model.

5.3. DICE Is Compatible with Other Direct Alignment from Preference Algorithms

Though DICE works best with DPO as it makes the iterative training possible (because the implicit reward model for the next round can be naturally derived using the updated policy), we would like to check if the dataset generated by DICE can also improve the base model with other DAP algorithms. In the Zephyr setting, we tune the policy model using DICE-generated dataset (at the first round) with KTO (Ethayarajh et al., 2024), IPO (Azar et al., 2024), and Hinge loss proposed in Zhao et al. (2023). The training follows the protocol described in Section 5.1. The results in Table 3 show that all DAP algorithms benefit from the newly generated data by the current policy and DPO implicit reward model with LR reward shaping, demonstrating LC win rates higher than the offline DPO. Meanwhile, DICE works best with DPO while IPO is the runner-up. It should be noted that DICE-IPO has a serious length exploitation issue even on the dataset where LR reward shaping has been applied.

5.4. Ablation Study

In this section, we investigate the effects of LR reward shaping and experience replay.

Effects of LR reward shaping. LR reward shaping (Eq. (8)) penalizes responses for being too verbose, and guides the construction of a debiased dataset with the optimal penalty strength $\alpha^\star$ found by optimizing Eq. (9). To validate the effectiveness of the propose LR reward shaping as well as the α-searching procedure, we run experiments in the Zephyr setting with different mixture ratios ($\gamma = 0$ and $\gamma = 0.5$) and ablate three design choices: (1) no LR reward shaping ($\alpha = 0$); (2) LR reward shaping with penalty strength found by Eq. (9), i.e., $\alpha = \alpha^\star = 0.023$; (3) LR reward shaping with slightly larger penalty strength ($\alpha = 2\alpha^\star$). Results are presented in Table 4. For all values of the γ, we observe that $\alpha = 0.0$ does lead to serious length exploitation. So, even if the policy can get a high win rate, it will suffer in the LC win rate due to the length exploitation issue (responses with longer average length get a lower LC win rate), e.g., the low LC win rate with $\gamma = 0.5, \alpha = 0.0$. In contrast, a larger α seemingly mitigates the length exploitation issue even better, but it may adversely affect the response quality. Our proposed approach of finding $\alpha^\star$ using the objective of minimizing the absolute difference of response length does provide the best performance.

Effects of experience replay. The experience replay results in a new mixed dataset in which γ fraction of the data is from the offline dataset and $(1-\gamma)$ fraction of the data is from the generated dataset. E.g., we use data only from the generated dataset if $\gamma = 0.0$. We run experiments in the Zephyr setting with $\gamma \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$ and conduct DICE in total of two rounds. From the results shown in Figure 3, we find $\gamma = 0.5$ provides the best performance. The results satisfy our expectations. With only offline preference data, the DPO optimizes the current policy with off-policy samples that are further away from its distribution. With only its own generated data, the model may keep reinforcing its current "belief", potentially leading to catastrophic forgetting. An

intermediate value $\gamma = 0.5$ finds a good balance. Additional results in the Llama3 setting can be found in Appendix C.

6. Limitations and Future Work

Limitations. One of the primary limitations of our work is the reliance on the DPO training prior to bootstrapping. If the implicit reward model is not well-trained, it can lead to a collapse of the training pipeline. This challenge is not unique to our approach; the classic RLHF pipeline is also struggling when its reward model is not well-trained. Another limitation is the lack of continued improvement over many iterations. Similar to other research, such as the work by (Yuan et al., 2024), which enhances the policy model without an external reward model, we did not observe continuous improvement in our model beyond three iterations. This issue highlights an open question within this field regarding the iterative enhancement of policy models.

Future Work. Future research could explore the rewarding capabilities of models trained using other DPO variants, such as KTO and IPO. Investigating whether these variants can offer general rewards similar to those provided by DPO-ed policy would be valuable. Another promising direction involves developing methods that enable continuous improvement of the policy model over iterations without degradation. Additionally, investigating a theoretical understanding of the policy learned through self-bootstrapping could provide deeper insights into the mechanics of our approach and facilitate further advancements.

7. Conclusion

In this paper, we introduce DICE, a novel approach that leverages the implicit reward model from DPO to further align LLMs with human preferences. Our method stands out in the current landscape of LLM alignment research, as it uses the implicit reward model to iteratively refine the policy model. Empirical results demonstrate that DICE surpasses the closed-source language models such as Gemini Pro, despite utilizing a model with only 8B parameters. This efficiency in parameter usage does not come at the cost of additional human annotations or external reward models, underscoring the practicality and accessibility of our approach.

Impact Statement

Due to its simplicity in implementation and its ability to enhance the policy model effectively, our method has the potential to be misused, for instance, in generating harmful content. The ease of use and the capability to produce stronger policy models might attract malicious actors who seek to exploit these features for unethical purposes. It is crucial to address these concerns by developing safeguards and ethical guidelines to prevent misuse.

References

Azar, M. G., Guo, Z. D., Piot, B., Munos, R., Rowland, M., Valko, M., and Calandriello, D. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pp. 4447–4455. PMLR, 2024.

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

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 2017.

Cui, G., Yuan, L., Ding, N., Yao, G., Zhu, W., Ni, Y., Xie, G., Liu, Z., and Sun, M. Ultrafeedback: Boosting language models with high-quality feedback, 2023.

Dubois, Y., Li, X., Taori, R., Zhang, T., Gulrajani, I., Ba, J., Guestrin, C., Liang, P., and Hashimoto, T. B. Alpacafarm: A simulation framework for methods that learn from human feedback, 2023.

Dubois, Y., Galambosi, B., Liang, P., and Hashimoto, T. B. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

Ethayarajh, K., Xu, W., Muennighoff, N., Jurafsky, D., and Kiela, D. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Guo, S., Zhang, B., Liu, T., Liu, T., Khalman, M., Llinares, F., Rame, A., Mesnard, T., Zhao, Y., Piot, B., et al. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*, 2024.

Huang, J., Gu, S. S., Hou, L., Wu, Y., Wang, X., Yu, H., and Han, J. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*, 2022.

Lee, H., Phatale, S., Mansoor, H., Lu, K., Mesnard, T., Bishop, C., Carbune, V., and Rastogi, A. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Li, X., Yu, P., Zhou, C., Schick, T., Zettlemoyer, L., Levy, O., Weston, J., and Lewis, M. Self-alignment with instruction backtranslation. *arXiv preprint arXiv:2308.06259*, 2023a.

Li, X., Zhang, T., Dubois, Y., Taori, R., Gulrajani, I., Guestrin, C., Liang, P., and Hashimoto, T. B. Alpacaeval: An automatic evaluator of instruction-following models, 2023b.

Meng, Y., Xia, M., and Chen, D. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Park, R., Rafailov, R., Ermon, S., and Finn, C. Disentangling length from quality in direct preference optimization. *arXiv preprint arXiv:2403.19159*, 2024.

Rafailov, R., Hejna, J., Park, R., and Finn, C. From r to q^*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024a.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024b.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. Experience replay for continual learning. *Advances in neural information processing systems*, 2019.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Sun, Z., Shen, Y., Zhang, H., Zhou, Q., Chen, Z., Cox, D., Yang, Y., and Gan, C. Salmon: Self-alignment with principle-following reward models. *arXiv preprint arXiv:2310.05910*, 2023.

Sun, Z., Shen, Y., Zhou, Q., Zhang, H., Chen, Z., Cox, D., Yang, Y., and Gan, C. Principle-driven self-alignment of language models from scratch with minimal human supervision. *Advances in Neural Information Processing Systems*, 36, 2024.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tajwar, F., Singh, A., Sharma, A., Rafailov, R., Schneider, J., Xie, T., Ermon, S., Finn, C., and Kumar, A. Preference fine-tuning of llms should leverage suboptimal, on-policy data. *arXiv preprint arXiv:2404.14367*, 2024.

Tang, Y., Guo, D. Z., Zheng, Z., Calandriello, D., Cao, Y., Tarassov, E., Munos, R., Ávila Pires, B., Valko, M., Cheng, Y., and Dabney, W. Understanding the performance gap between online and offline alignment algorithms. *arXiv preprint arXiv: 2405.08448*, 2024.

Tran, H., Glaze, C., and Hancock, B. Iterative dpo alignment. Technical report, Snorkel AI, 2023.

Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., von Werra, L., Fourrier, C., Habib, N., Sarrazin, N., Sanseviero, O., Rush, A. M., and Wolf, T. Zephyr: Direct distillation of lm alignment, 2023.

Xie, Y., Goyal, A., Zheng, W., Kan, M.-Y., Lillicrap, T. P., Kawaguchi, K., and Shieh, M. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.

Yuan, W., Pang, R. Y., Cho, K., Sukhbaatar, S., Xu, J., and Weston, J. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Zhao, Y., Joshi, R., Liu, T., Khalman, M., Saleh, M., and Liu, P. J. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

Zhong, H., Feng, G., Xiong, W., Zhao, L., He, D., Bian, J., and Wang, L. Dpo meets ppo: Reinforced token optimization for rlhf. *arXiv preprint arXiv: 2404.18922*, 2024.

A. Prompts Used by LLM-as-a-Judge

LLM-as-a-Judge Prompt Template

Review the user's question and the corresponding response using the additive 5-point scoring system described below. Points are accumulated based on the satisfaction of each criterion:

- Add 1 point if the response is relevant and provides some information related to the user's inquiry, even if it is incomplete or contains some irrelevant content.
- Add another point if the response addresses a substantial portion of the user's question, but does not completely resolve the query or provide a direct answer.
- Award a third point if the response answers the basic elements of the user's question in a useful way, regardless of whether it seems to have been written by an AI Assistant or if it has elements typically found in blogs or search results.
- Grant a fourth point if the response is clearly written from an AI Assistant's perspective, addressing the user's question directly and comprehensively, and is well-organized and helpful, even if there is slight room for improvement in clarity, conciseness or focus.
- Bestow a fifth point for a response that is impeccably tailored to the user's question by an AI Assistant, without extraneous information, reflecting expert knowledge, and demonstrating a high-quality, engaging, and insightful answer.

User: {instruction}
Response: {response}

After examining the user's instruction and the response, provide brief step-by-step justifications and conclude with a score between 0 to 5. You must follow the format below:

Evaluation: <evaluation>
Score: <score>

Figure 4: We follow the prompt template used by Yuan et al. (2024) to use LLMs to judge model responses and construct paired dataset for further preference tuning.

B. Optimization for Length Regularized Reward Shaping

We solve the objective in Eq. (9) using a simple Bayesian optimization toolkit based on Gaussian process[6]. The objective landscape with respect to α is depicted in Figure 5, where we compare the proposed implicit reward model with the LLM-as-a-Judge reward model. With our length-reguarlized implicit rewards, the optimizer is able to find the optimal solution quickly that debiases the length difference of the winning and losing responses. For LLM-as-a-Judge rewards, the optimal solution is obtained with $\alpha = 0$, hence we do not explicitly debias the dataset for all the experiments.

C. Extended Ablation Study

In the Llama3 setting, we also conduct a coarse sweeping for the experience ratio γ, and present the AlpacaEval 2.0 LC win rate in Figure 6 for two self-alignment rounds. We observe similar trends to those in the Zephyr setting, which further justify the effectiveness of the proposed experience replay: it helps to keep a balance between the more on-policy generated data and the curated offline data. The best identified value of the mixture ratio is $\gamma = 0.1$.

[6] https://scikit-optimize.github.io/stable/modules/generated/skopt.gp_minimize.html

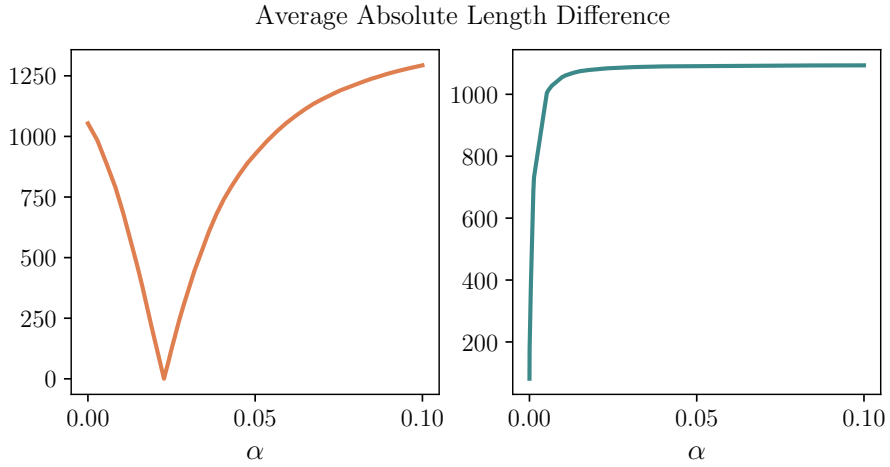

Figure 5: The objective landscape for (**Left**) our implicit reward model and (**Right**) the LLM-as-a-Judge reward model.

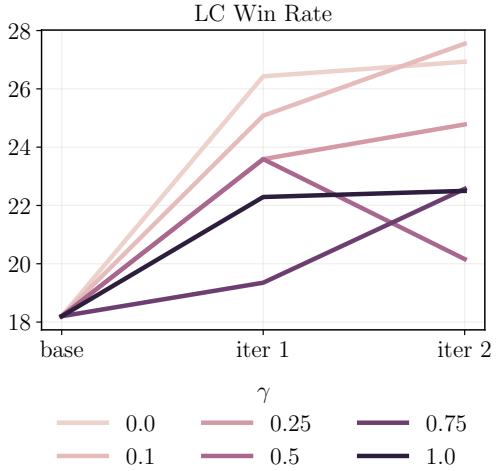

Figure 6: AlpacaEval 2.0 LC Win Rate across different experience replay ratio (γ) in the Llama3 setting.