Expectation Preference Optimization: Reliable Preference Estimation for Improving the Reasoning Capability of Large Language Models

Anonymous ACL submission

Abstract

Pairwise preference optimization, such as Direct Preference Optimization (DPO), was originally designed to align large language models (LLMs) with human value. It has recently been used to improve the supervised fine-tuning (SFT) performance of LLMs. Using pairs of single samples, DPO estimates the probability distribution of the preferences of picking one response over another. However, in tasks that involve more complicated preferences (e.g., reasoning tasks) than those in the human value alignment task, this sampling method is likely to bring deviations from the ground-truth distribution. To solve the problem, extra efforts (e.g., external annotations or amendment of the loss function) are often required. In this paper, we hypothesize that the preferences can be better estimated through a multi-sampling process. Accordingly, we propose an Expectation Preference Optimization (EPO) algorithm that takes pairs of sample groups, instead of pairs of single samples as in DPO, for preference learning. Compared to pairwise DPO, the proposed EPO tends to produce more reliable preference estimations. Applying different preference optimization methods in a self-training paradigm, we have conducted extensive experiments on various reasoning benchmarks. The results show that our EPO approach outperforms a range of baseline approaches in terms of zero-shot accuracy on all benchmarks.

1 Introduction

Large language models (LLMs), through supervised fine-tuning (SFT), have shown remarkable abilities on various reasoning tasks such as mathematical reasoning. However, it is well recognized that the effectiveness of SFT can reach its upper limit depending on the scale and quality of training samples, which are often limited and expensive to construct. Thus, an important question arises: *with the same SFT training data, how can we further improve the SFT performance?* To tackle the problem, pairwise preference optimization, which was originally developed to align with human values (e.g., harmlessness or honesty), has become a widely chosen solution.

Direct Preference Optimization (DPO) (Rafailov et al., 2024) is one of the most popular preference-based methods due to its simplicity and effectiveness compared to Reinforcement Learning with Human Feedback (RLHF) (Bai et al., 2022). DPO samples the preferred and dis-preferred responses once in one updating step on a prompt, and then uses the Bradley-Terry (BT) model to update the LLM with an implicit reward function that models the preference of picking the preferred sample over the dis-preferred one. As can be naturally applied in the self-improving approaches that alleviate the issue of data construction (Yuan et al., 2024; Sun et al., 2023), using DPO in reasoning tasks has shown a broad prospect.

The selection of pairwise training data is key in the utilization of DPO. The preferred and dis-preferred responses on a prompt represent an estimation of the correct preference, which in the training process guides the optimization direction (Rafailov et al., 2024). Different from the human value alignment task, in most reasoning tasks, the direction that the model needs to optimize can be more multifaceted. For example, in mathematical reasoning, the error of an answer can be attributed to various aspects, such as calculation, formula, and entity errors. Thus, directly using DPO on such reasoning tasks, especially when using correctness as the selection criterion for pairs of samples, would be insufficient to reflect the multifaceted nature of the reasoning tasks and result in poor performance (Lu et al., 2024; Lai et al., 2024). As shown in Fig. 1 (the red box on the left-hand side), sampling a pair of single responses for optimization, with one reporting the correct answer and the other on the opposite, may lead to a wrong direction of preference estimation that deviates from

the other correct responses (marked with crying faces).

Various approaches have been developed to solve this problem. Orca-Math (Mitra et al., 2024) applies preference optimization on a fine-tuned LLM using an augmented dataset that is constructed using GPT4 to select the pairs of responses, while Brain (Chen et al., 2024a) uses human annotations. DPOP (Pal et al., 2024) tries to solve the unstable optimization direction of pairwise optimization by enhancing the supervision of preferred ends in changing the loss function of DPO. Step-DPO (Lai et al., 2024) uses a large amount of sampling responses and boosts the training data into large step-level pairs. Iterative RPO (Yuanzhe Pang et al., 2024) uses a similar form of loss and applies it to a self-training structure. However, these methods do not fundamentally solve the problem of unstable preference modeling when facing complicated preferences.

In this paper, we explore a different perspective by *leveraging more samples in preference estimation.* Starting with the basic Bradley-Terry (BT) model, which is the basis of pairwise training, we hypothesize that the preferences in the BT model can be better estimated through a weighted multi-sampling process. Specifically, we assume that the preferences are not generated by the estimation of a single response, but by the expectation of the response sampling. Under this assumption, we propose an Expectation Preference Optimization (EPO) approach, a variant of DPO. EPO accepts group-wise preference samples, i.e., pairs of sample groups, for training, with a length limitation operation. EPO estimates the preference by calculating the weighted mean of each group. Our EPO shares the same objective with DPO and RLHF while overcoming the limitation of using only one preferred and one dis-preferred response each time. As shown in Fig. 1 (right-hand side), EPO makes it easier to produce proper preference estimations in reasoning tasks.

Utilizing the proposed EPO, we can simply use correctness (i.e., if the sampled responses answer the question correctly) as the signal for preference construction on reasoning tasks and boost the capability of LLMs yet bring no further human annotations. We apply a self-training algorithm that is detailed in Section 3.3 which requires no extra annotations and small cost on data preprocess. After SFT on a task-specific reasoning dataset, the target LLM generates responses for the input queries. Then we divide the responses for each query into two groups. Using EPO on these grouped responses, the optimization direction is estimated through multiple samples. Extensive experiments on various reasoning benchmarks (i.e. GSM8K (Cobbe et al., 2021), ARC (Clark et al., 2018), SocialQA (Amini et al., 2019), MathQA (Sap et al., 2019)) across different base LLMs (including Llama2-7B, Llama2-13B (Touvron et al., 2023), Qwen1.5-7B (Bai et al., 2023), Mistral-7B (Jiang et al., 2023)) show that our EPO constantly improves the performance of SFT models and outperforms other preference optimization baselines in the self-training framework.

2 Preliminaries

Given a large language model that is parameterized by θ, donated as π_θ, there are two categories of methods to improve its performance: fine-tuning-based and preference-optimization-based methods.

2.1 Fine-Tuning

SFT: Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, π_θ is finetuned with the cross-entropy loss following a typical chain-of-thought rationale y_i with respect to the input query x_i, resulting in π_θ^{SFT}.

RFT: Rejection Sampling Fine-Tuning (RFT) (Yuan et al., 2023) is a training method where π_θ is fine-tuned on its own correct generations. After SFT on $\mathcal{D}$, π_θ^{SFT} obtains the ability to perform zero-shot chain-of-thought rationales. Thus we can sample M candidate rationales $\hat{y_{i,1}}, \hat{y_{i,2}}, \cdots \hat{y_{i,M}}$ for each query x_i. All the rationales together are denoted as $\hat{\mathcal{D}} = \left\{(x_i, \hat{y}_{i,j})_{j=1}^M \mid (x_i, y_i) \in \mathcal{D}\right\}$.

Utilizing a filtering method (e.g. reward model annotation), we can construct $\hat{\mathcal{D}}_{RFT}$ as a subset of $\hat{\mathcal{D}}$. The outcome π_θ^{RFT} is trained on the augmented dataset $\mathcal{D} \cup \hat{\mathcal{D}}_{RFT}$ based on π_θ.

2.2 Preference-Optimization

RLHF: RLHF (Bai et al., 2022) fits a reward model to pairwise samples of human preferences and then uses Reinforcement Learning to optimize a language model policy to produce responses that are assigned high reward without drifting excessively far from the original model. Consider an annotated dataset of pairwise samples $\mathcal{D}_p = \left\{x_i, y_w^i, y_l^i\right\}_{i=1}^N$, where x_i denotes the ith prompt, y_w^i and y_l^i respectively represent the preferred and dis-preferred responses to x_i. RLHF begins by modeling the probability of preferring y_w^i to y_l^i using the Bradley-Terry

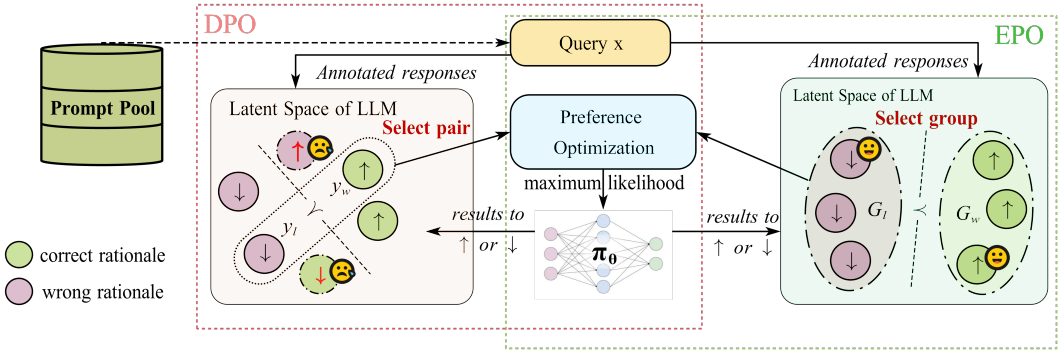


Figure 1: In the latent space of the target LLM, DPO chooses a pair of samples using correctness as the signal. In more complicated case, as shown in the figure, DPO can result in a wrong estimation of the preference and drive the LLM to a wrong reward updating direction (i.e., increased reward to the wrong samples and decreased to the correct samples). On the opposite, EPO considers multi-sampling and can provide a more reliable optimizing direction.

model (Bradley and Terry, 1952), which appoints the following probabilistic form:

$$p\left(y_w^i \succ y_l^i \mid x\right) = \sigma\left(r\left(x_i, y_w^i\right) - r\left(x, y_l^i\right)\right) \quad (1)$$

where σ represents the logistic function and $r(x_i, y_i)$ corresponds to a reward function r_ϕ (i.e., LLM classifier) that gives the estimation of y_i with respect to x_i according to human preference.

Then the target model π_θ can be trained by the feedback from the learned reward function. In general, we formulate the following optimization target for this learning process:

$$\max_{\pi_\theta} \mathbb{E}\left[r_\phi(x, y)\right] - \beta \mathbb{D}_{\mathrm{KL}}\left[\pi_\theta(y \mid x) \| \pi_{\mathrm{ref}}(y \mid x)\right] \quad (2)$$

where β is a parameter controlling the deviation of the target model π_θ from the status when the training starts.

DPO: DPO (Rafailov et al., 2024) shows the possibility of keeping the same optimization target as RLHF without explicitly training a reward function and the implementation of RL. The loss function of DPO is presented as below:

$$\mathcal{L}_{\mathrm{DPO}}\left(\pi_\theta; \pi_{\mathrm{ref}}\right) = -\mathbb{E}_{(x, y_w, y_l) \sim D} \log \sigma \left(\beta \log \frac{\pi_\theta\left(y_w \mid x\right)}{\pi_{\mathrm{ref}}\left(y_w \mid x\right)} - \beta \log \frac{\pi_\theta\left(y_l \mid x\right)}{\pi_{\mathrm{ref}}\left(y_l \mid x\right)}\right) \quad (3)$$

Notably, this optimization objective is based on a theoretical optimal π_θ beyond $r_U(x, y)$, which enables its equivalence with Eq.2.

3 Expectation Preference Optimization

3.1 An Analysis of Pairwise Preference Optimization

Taking DPO as an example, Pairwise Preference Optimization methods accept one preferred sample and one dis-preferred sample as the unit to calculate the loss for updating the reward function. Considering an ideal reward function $\hat{r}(x, y)$ reflects the ground-truth preference, let us assume a sampling of four responses $\{y_{\alpha 1}, y_{\alpha 2}, y_{\beta 1}, y_{\beta 2}\}$ with respect to the query x, where $\hat{r}(x, y_{\alpha i}) > \hat{r}(x, y_{\beta i})$ holds. When an initial reward function r_ϕ^t is optimized on $(y_{\alpha 1}, y_{\beta 1})$, the optimization directions of $y_{\alpha 2}$ and $y_{\beta 2}$ are not restricted to follow the ground-truth. The updated r_ϕ^{t+1} may give a wrong estimation $r_\phi^{t+1}(x, y_{\alpha 2}) < r_\phi^{t+1}(x, y_{\beta 2})$ while correctly estimating the training pair as $r_\phi^{t+1}(x, y_{\alpha 1}) > r_\phi^{t+1}(x, y_{\beta 1})$, and vice versa.

The trigger for this issue is that the sampling of $(y_{\alpha 1}, y_{\beta 1})$ with respect to the prompt x may be away from the ground-truth preference distribution. Accordingly, the optimization of r_ϕ^t gives wrong guidance on $y_{\alpha 2}$ and $y_{\beta 2}$. When the purpose of training is to align with humans, the inconsistency of preference estimation is not so prominent (compared to reasoning tasks), so the problem is less significant. However, the reasoning tasks present a different situation. For example, in math reasoning tasks such as GSM8K, LLMs can make mistakes for many reasons (e.g., equation calculation errors, incorrect understanding of problems, etc.) and the estimates from different aspects are not independent. Thus the true preference distribution is complicated and varies with the target LLM.

3.2 Expectation Preference Optimization

Aiming to solve the aforementioned problem brought by the single sampling of preference distribution in the reasoning tasks, we propose an Expectation Preference Optimization (EPO) algorithm starting from the RLHF pipeline. As we have previously mentioned, the reward modelling phase of RLHF is based on the BT model. After a single sampling of response pair (y_1, y_2) for a prompt x, we can annotate the responses using human labellers or some stronger LLMs. As the preferences are presented as $y_w \succ y_l \mid x$ where $y_w, y_l \in \{y_1, y_2\}$ we can optimize a reward function through Eq. 1.

By estimating preferences through multi-sampling, which results in a group of responses $\{y_i\}_{i=1}^N$ for a prompt x, we present the group-wise preference form $G_w \succ G_l \mid x$ where $G_w, G_l \subseteq \{y_i\}_{i=1}^N$. In general, G_w represents the preferred group and G_l represents the dis-preferred group. We assume that the reward level of G_w and G_l is the expectation for all rewards in the group:

$$r^*(x, G) = \mathbb{E}_{y_i \sim G}[r(x, y_i)] \tag{4}$$

Thus the Bradley-Terry model can be rewritten as:

$$\begin{aligned} &p^*\left(G_w \succ G_l \mid x\right) = \\ &\sigma\left(\mathbb{E}_{G_l}[r(x, y_i)] - \mathbb{E}_{G_w}[r(x, y_i)]\right) \end{aligned} \tag{5}$$

EPO objective. Following the derivation process of DPO, we can construct the reward function under the optimal solution to Eq. 2 as follows:

$$r(x, y) = \beta \log \frac{\hat{\pi}(y \mid x)}{\pi_{\text{ref}}\left(y \mid x\right)} + \beta \log Z(x) \tag{6}$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right)$ represents a partial function referring to the previous work (Peters and Schaal, 2007; Rafailov et al., 2024). Using this re-parameterization of $r(x, y)$, Eq. 5 can be formed as below using the optimal solution.

$$\begin{aligned} &p^*\left(G_w \succ G_l \mid x\right) = \sigma(\beta P_{G_l} - P_{G_w}) \\ &P_G = \mathbb{E}_G[\log \frac{\pi\left(y_i \mid x\right)}{\pi_{\text{ref}}\left(y_i \mid x\right)})] \end{aligned} \tag{7}$$

Due to space limitation, we present our proof and detailed deriving process in the Appendix.

We can now formulate a minimum loss function for the target model π_θ through this preference function:

$$\mathcal{L}_R\left(r_\phi, \mathcal{D}\right) = -\mathbb{E}_{(x, G_w, G_l) \sim \mathcal{D}}[log\sigma(P)] \tag{8}$$

While the sampling model (reference model) provides the group result (i.e. G_w, G_l), we regard the $\pi_{\text{ref}}\left(y_i \mid x\right)$ as the probability of y_i in the expectation. In practice, this means that the response with higher probability have a higher impact on the overall optimization direction. Thus, the loss function of EPO can be derived as:

$$\begin{aligned} &\mathcal{L}_R\left(r_\phi, \mathcal{D}\right) = -\mathbb{E}_{(x, G_w, G_l) \sim \mathcal{D}} \\ &[log\sigma\left(\beta f(G_w, \pi, \pi_{ref}) - \beta f(G_l, \pi, \pi_{ref})\right)] \\ &f(G, \pi, \pi_{ref}) = \frac{\sum_{y_i \in G} \pi_{ref}(y_i|x)^\gamma \log \frac{\pi(y_i|x)}{\pi_{\text{ref}}(y_i|x)}}{\sum_{y_i \in G} \pi_{ref}(y_i|x)^\gamma} \end{aligned} \tag{9}$$

Notably, this method only calculates an approximate expectation, as the sum of probabilities is not 1. Thus we introduce a smoothing coefficient $0 < \gamma \leq 1$, to avoid weights with large variants caused by incomplete calculation of expected deviations.

A further interpretation of EPO. We here present a brief analysis of EPO. The objective function of EPO is derived from RLHF, which means that we share the same overall optimal solution with RLHF and DPO. As we estimate the preferences through a multi-sampling assumption, EPO has a more reliable implicit reward function compared to the pair-wise DPO, especially in reasoning tasks with complicated preferences. EPO drives the target LLM to have higher probabilities of generating responses in the preferred group and lower probabilities of generating responses in the dispreferred group, while ensuring the responses with higher probabilities affect more on the optimization. Notably, when the sampling number of G_l and G_w is 1, EPO becomes a typical DPO algorithm. Theoretically, in random sampling, the larger the sampling number, the more accurate the estimation of preferences in line with the ground-truth distribution.

Length Limitation Operation. After the brief analysis of the EPO loss function, we introduce an additional module to the EPO algorithm. Previous work (Wang and Zhou, 2024) indicates that the beginning tokens affect most of the decoding (generating) process of an LLM. Considering the subsequent tokens of the responses could adversely

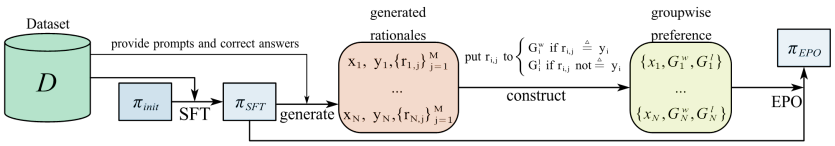


Figure 2: Overview of self-improving approach with EPO

impact the coherence of the model in the optimizing process, especially the dis-preferred responses, we aim to increase the stability of the EPO optimization process by limiting the length of samples.

Specifically, we truncate the responses in G_l and G_w and ensure that the length of responses is smaller than a preset threshold. Knowing that this truncation drops some information from the supervised data, we will analyze the effect of this operation in our experiments.

3.3 Self-improve Training approach With EPO

As EPO can provide reliable preference estimation, we can simply use correctness (i.e., whether the answer of sampled response is the same as the answer of the target) as signals and boost the capability of LLM on the datasets that contains verifiable answers (e.g., math datasets). We design a self-improve training approach, which is presented in Fig 2.

We start with access to a base LLM π_{init} and data of a verifiable task $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$. First, we give the model the ability to follow and generate rational instructions by applying SFT to it. The fine-tuned model is denoted as π_{SFT}. Then we generate M different responses for every query in $\mathcal{D}$. We denote all the generated responses (R_i) with the original responses y_i as $\mathcal{D}_{aug} = \{x_i, y_i, R_i\}_{i=1}^{N}$ where $R_i = \{r_{i,j}\}_{j=1}^{M}$.

In the next step, we generate the group-wise preference data from $\mathcal{D}_{aug}$ using the correctness of generated responses in R_i as the annotation signal. Specifically, if a response reports the same answer as the typical rationale, it is put in G^w and it is put in G_l while it reports a different answer (which means it is wrong). The constructed training data are presented as follows:

$$\mathcal{D}_{EPO} = \{x_i, G_i^w, G_i^l\}_{i=1}^{N'} \tag{10}$$

where $G_i^w \cup G_i^l = R_i \cup \{y_i\}$. Notably, we construct the preference groups on R_i combining with y_i, thus for each prompt x the number of candidates' correct responses is always greater than 1. As the wrong response of a query does not always exist in the sampling, we drop the triplets in $\mathcal{D}_{aug}$ whose R_i contains all correct responses.

Applying EPO algorithm on π_{SFT} with $\mathcal{D}_{EPO}$, we can obtain the resultant LLM denoted as π_{EPO}. In general, π_{EPO} is optimized based on the supervising information of base dataset $\mathcal{D}$ (i.e. the correct answer), and the self-improve training ensures that the model can have better performance on the fine-tuning dataset.

4 Experiments

We evaluate the effectiveness of our EPO on two representative reasoning tasks: arithmetic reasoning and commonsense reasoning. We test four different LLMs: Llama3-8B (Dubey et al., 2024), Llama2-13B (Touvron et al., 2023), Qwen2.5-7B (Yang et al., 2024) and Mistral-7B (Jiang et al., 2023) as our base LLM model. We mainly evaluate the performance of EPO in the self-improving scenario. Notably, we put our Implementation Details in the Appendix D.4.

4.1 Datasets and Preprocessing

The experiments are carried out on two arithmetic reasoning datasets and three commonsense reasoning datasets.

GSM8K. GSM8K (Cobbe et al., 2021) has been adopted as a benchmark for the mathematical reasoning skills of LLMs. It contains 7,473 training and 1,319 test problems, and each sample is paired with a rationale that clearly states the final answer.

MetaMath$_s$. MetaMath (Yu et al., 2023) is a popular augmentation of GSM8K and MATH (Hendrycks et al., 2020). It contains 240K augmented samples based on GSM8K and 155K samples based on MATH. Notably, for lighter response generation, we only take 80K augmented GSM8K samples for training. The subset is denoted as MetaMath$_s$.

AI2 Reasoning Challenge (ARC). ARC (Clark et al., 2018) consists of two subsets: ARC-Easy

Table 1: The False Positive Situation.

N	5	10	20	30	50
All Positive	22800	45660	79508	118978	198047
False Positive	804	1634	4252	6921	11884
Proportion	3.52%	3.58%	5.35%	5.82%	6.00%

and ARC-Challenge. To obtain the rationales of the queries for SFT, we apply a strong LLM (i.e., Yi-Chat-34B (Young et al., 2024)) to generate typical answers. Using the prompt presented in the Appendix, we generate a rationale ending with an answer statement for each query. After filtering the rationales with wrong answers and incorrect format, we construct an SFT training set with 1599 samples from ARC-Easy and another with 793 samples from ARC-Challenge. They are then applied in the first SFT phase of the approach. For the generation phase, we use the original training set.

MathQA. MathQA (Amini et al., 2019) contains 29837 training samples and 2985 test samples. Each sample contains a math query, four candidate results, a rationale, and a correct answer. We manually add the answer statements at the end of the rationales for SFT.

SocialIQA. Social IQA (Sap et al., 2019) has 33410 training samples, each containing a query and 3-5 candidate results without rationales, as well as 2224 test samples. We utilize the same method we use in constructing the ARC SFT dataset to generate rationales. Notably, we generate 23624 samples with one correct rationale each.

4.2 Baselines

In the experiments, we compare the proposed self-training EPO method (i.e. SFT + EPO) with various existing self-training approaches. They are: **SFT**, (SFT +) **RFT**, (SFT +) **DPO**, (SFT +) **DPO**$_{batch}$, (SFT +) **RPO**, and (SFT +) **Step-DPO**. We present the detailed introduction in the Appendix D.1.

4.3 Analysis of Misclassification Situation

Since our experiments are based on the rule-based verifier to label the correctness of the sampled responses, there could be misclassified samples. For True Negative samples, we consider that it either gives no answer at the end of the responses or gives the wrong formatted answers, which is not the behavior we want the model to learn. For the analysis of False Positive samples, we utilize DPSK-Distill-Qwen-32B(Guo et al., 2025) to annotate whether the positive sampled responses are true positive using the prompt given in Appendix A. From Tab. 1 we can observe that although the proportion of false positive samples increases with the increase of N, it only hovers around a 5% proportion of all positive samples. This can indirectly confirm the effectiveness of our method.

4.4 Main Results

The main results of our experiments are presented in Tab. 2 and Tab. 3. Remarkably, on the GSM8K benchmark, EPO achieves a 5.43% increase over the SFT model in accuracy on the GSM8K dataset and 3.29% based on the Metasub$_s$ dataset for Llama2-13B. This improvement comes to 2.64% and 2.05% for Qwen2.5-7B. As for the Commonsense tasks, EPO brings an increase of 3.58% for Llama3-*B on SocialIQA, 4.47% for Mitral-7B on ARC-Easy, 6.94% for Llama2-13B on ARC-Challenge, and 6.29% for Mistral-7B on MathQA.

A cursory examination reveals that our EPO consistently outperforms all the preference optimization baselines across all tasks. Such a pattern underscores the effectiveness of EPO in improving LLM's ability in reasoning tasks. The DPO baselines can eventually damage the performance of the model, and this happens more frequently in mathematical reasoning. The DPO$_{batch}$ method also shows an unstable effect compared to the DPO, while it can bring a slight improvement in many cases. RPO, compared to the former two, shows a more stable improvement effect. However, our EPO provides a more reliable preference estimation and constantly brings better performance improvements.

4.5 Further Analysis

4.5.1 Analysis of Generation Parameters and Length Limitation

Effect of sampling temperature and length limitation. We analyze the effect of sampling temperature in the generation phase and the length limitation operation in the training phase. Fig. 3(a) shows the effectiveness of length limitation in contributing to the optimization stability. For GSM8K datasets, limiting the length of participation in the responses to the interval between 10 and 20 can result in better performance. As the sampling temperature grows, the peak is gradually moving to the right. We consider this effect to be due to the increasing variety of responses that would decrease the instability of responses.

Effect of sampling number and length limitation. We analyze the effect of sampling number

Table 2: Overall results on the math tasks in comparison with four base models. We report the accuracy of CoT Pass@1 greedy sampling. The best performance is in bold and the second-best is underlined.

Base Model	Datasets	SFT Result	Post Methods					
			RFT	DPO	DPO_{batch}	RPO	Step-DPO	EPO
Llama3-8B	GSM8K	50.03	53.27	50.83	49.07	51.85	51.70	**53.92**
	$MetaMath_s$	77.25	76.02	75.37	76.12	79.02	79.78	**81.03**
Llama2-13B	GSM8K	49.27	47.99	48.47	48.53	50.09	51.83	**54.70**
	$MetaMath_s$	69.82	68.38	67.39	68.46	71.19	70.27	**73.11**
Qwen2.5-7B	GSM8K	75.59	73.02	73.85	72.93	76.02	76.25	**78.23**
	$MetaMath_s$	82.03	81.32	81.19	80.37	81.85	82.24	**84.08**
Mistral-7B	GSM8K	41.84	41.74	39.57	38.89	41.48	43.25	**45.40**
	$MetaMath_s$	70.05	70.15	68.01	68.29	71.72	71.29	**74.72**

Table 3: Overall results on the Commonsense tasks in comparison with 4 base models. We report the accuracy of CoT Pass@1 greedy sampling. The best performance is in bold and the second-best is underlined.

Base Model	Datasets	SFT Result	Post Methods					
			RFT	DPO	DPO_{batch}	RPO	Step-DPO	EPO
Llama3-8B	ARC-Easy	81.31	81.24	83.52	81.45	82.73	82.92	**84.10**
	ARC-Challenge	52.98	**56.56**	54.77	55.02	54.88	53.05	55.74
	MathQA	52.16	53.75	51.29	50.77	52.75	52.03	**55.37**
	SocialIQA	75.17	71.82	77.39	76.58	77.12	75.47	**78.75**
Llama2-13B	ARC-Easy	82.28	82.07	82.74	82.93	83.20	83.31	**84.35**
	ARC-Challenge	57.93	62.62	61.60	62.07	63.99	64.72	**64.87**
	MathQA	44.62	**47.07**	38.22	43.37	45.31	45.93	46.91
	SocialIQA	74.14	74.55	78.50	77.58	77.36	77.46	**79.86**
Qwen2.5-7B	ARC-Easy	91.03	89.30	90.52	90.33	91.86	91.97	**92.15**
	ARC-Challenge	84.55	83.92	85.49	86.14	84.72	86.49	**87.28**
	MathQA	67.67	68.25	66.92	67.64	68.75	68.30	**68.96**
	SocialIQA	77.02	76.84	77.31	76.32	77.95	78.37	**78.94**
Mistral-7B	ARC-Easy	74.47	72.83	74.83	75.05	78.30	78.33	**78.94**
	ARC-Challenge	60.45	62.71	63.84	60.03	62.97	63.45	**64.73**
	MathQA	52.09	52.36	50.83	51.95	55.70	57.92	**58.38**
	SocialIQA	74.10	74.37	76.30	75.58	76.15	75.33	**78.05**

in the generation phase and the length limitation operation in the training phase. As shown in Fig. 3(b), with the increase of the sampling number, the performance increases for the length limitation of less than 20. This result indicates that our EPO estimates the preference distribution more accurately as the number of samples increases. When the length limitation is increased, this benefit becomes unstable.

4.5.2 Effect of EPO from the Training Set Perspective

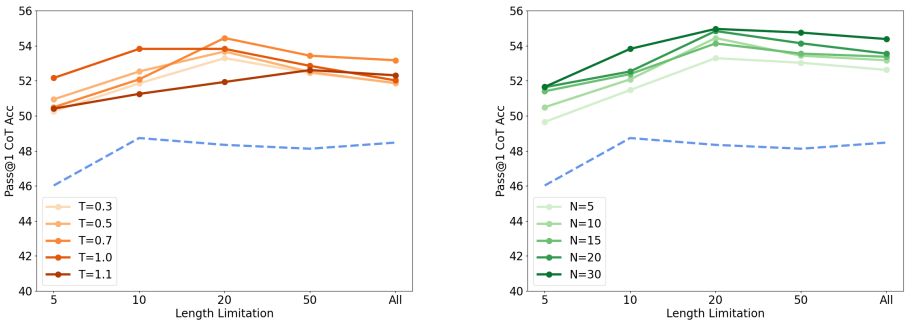
Considering that all the self-improving methods can more effectively utilize the training set compared to simple SFT, we analyze the performance of our EPO in comparison with baselines from the perspective of the training set. We apply an N=5 inference on GSM8K for each trained model with different methods. Taking the leftmost bar (SFT) in Fig. 4 as the reference, we can observe that EPO increases the probability of the model responding correctly (i.e., increased number of the "5" segments and decreased number of the "0" segments) most. In fact, EPO drives the increase of the number of all-correct generations from 2441 to 3253, while DPO and RPO even drive it to decrease.

4.5.3 Effects of Sampling Distribution on Training Result

Table 4: Effect of sampling distribution on DPO. "Highest / Lowest Prob" represents the selection of the responses with the highest / lowest probabilities

Base Model	Random	Highest Prob	Lowest Prob
Llama3-8B	54.05	53.25(-0.80)	51.37(-2.68)
Llama2-13B	54.96	54.61(-0.35)	50.58(-4.38)

As we utilize the expectation of a sampling process to estimate the preference in EPO, the sampling distribution (i.e. the samples in groups) can affect the final optimization direction. Here we present an analysis of the choice of responses for EPO. Firstly, we apply an N=30 generation on GSM8K with T=0.7. Then we present three different methods to select 15 responses for each prompt: randomly selecting, selecting the responses with the highest probabilities, and selecting the responses with the lowest probabilities. We perform this analysis on two base LLMs: Llama3-*B and Llama2-13B. As shown in Table 4, the randomly selecting approach presents the best performance, and selecting with the lowest probabilities shows a poor performance. This implies that when se-



(a) (b)

Figure 3: Analysis of hyperparameters. The analysis experiments are conducted on GSM8K for Llama2-13B. The sampling number for the experiments in (a) is set to 10, and the temperature for the experiments in (b) is set to 0.7. The blue dashed line represents the performance of DPO utilizing the length-limitation method.

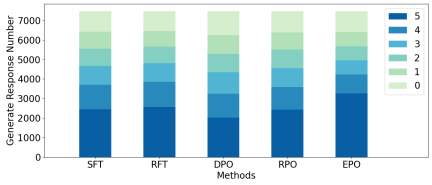


Figure 4: We calculate the number of correct responses for each query in an N=5 generation for each method on GSM8K, using Llama2-13B as base LLM. The different colors reflect different numbers of correct responses. The length of the bar represents the number of prompts.

lecting sample groups, it is necessary to follow a true distribution that guides a correct optimization direction, otherwise optimization deviations may occur, leading to poor performance.

5 Related Work

Despite the success of instruction tuning on LLMs which has shown a great zero-shot performance (Chung et al., 2024; Mishra et al., 2021; Sanh et al., 2021), preference optimization has demonstrated its great effectiveness in aligning LLMs with humans (Bai et al., 2022). As reinforcement Learning with Human Feedback (RLHF) (Bai et al., 2022) is a complex and often unstable procedure (Pal et al., 2024), DPO (Rafailov et al., 2024) has been proposed as a more stable and computationally lightweight algorithm with no need for extra reward function training.

Reasoning ability is important for LLMs in practice. Let us take mathematical reasoning as an example. To make a stronger math-reasoning model, previous studies have focused on training the base model on larger datasets of better quality (Yuanzhe Pang et al., 2024; Yu et al., 2023). However, it is well-recognized that creating large-scale and better-quality training samples is challenging and expensive.

The use of preference learning to improve the LLM's reasoning ability has attracted increasing attention, while also facing certain problems. DPOP (Pal et al., 2024) enhances the supervision of the positive end in DPO by adjusting the loss function. Iterative RPO (Yuanzhe Pang et al., 2024) presents a similar loss function in a self-improving scenario without the SFT phase. Step-DPO (Lai et al., 2024; Lu et al., 2024) takes extra effort to create step-wise paired data and utilizes methods that are similar to vanilla DPO. However, these methods do not solve the problem of preference estimation of pair-wise optimization, thus gaining little improvement.

6 Conclusions and Future Work

In this paper, we propose an Expectation Preference Optimization (EPO) method that accepts pairs of response groups for preference learning. Compared to the existing pairwise preference optimization approaches, our EPO method can more reliably estimate the preference distribution, especially when facing complicated reasoning tasks. We further design a self-improving framework, in which EPO can be effectively leveraged to improve the reasoning ability of LLMs. Experimental results on various reasoning tasks and datasets demonstrate the superior performance of our EP

For future work, we plan to explore other reasonable methods (e.g., adding weights on responses) to better estimate the preferences based on EPO.

7 Limitations

Our paper presents a simple and practical method to improve the capability of LLMs in any reasoning task. However, the theory of EPO is not confined to reasoning tasks. Our intuition is to replace a single sample with an expectation in the Bradley-Terry model. Thus EPO can also used in alignment tasks. However, we have not found a proper way to calculate the expectation in alignment tasks since in reasoning tasks the answer to a query is binary (i.e., correct or incorrect) while it is not in alignment tasks. Finding a proper method to calculate the expectation in alignment tasks can be a more comprehensive demonstration of the superiority of EPO theory.

8 Discussion of Ethical Considerations

Our proposed methods are used to improve the capabilities of LLMs. Though we mainly utilize it in reasoning tasks, it can also be used in other tasks which depends on the purpose of its user. On the other hand, using EPO training LLMs may cause an environmental impact as all other training methods do.

For the permissions of our used artifact, each of our used models (Llama2-13B, Llama2-7B, Mistral-7B, Qwen1.5-7B) and the datasets (GSM8K, ARC, MathQA) are open-sourced and can be found from Github or Huggingface. Secondly, all the models can not be used commercially.

We utilize all the models and datasets consistent with their intended use. We do not provide extra data. Our construction of self-training data using the LLMs presents the answers to the datasets, which is the purpose LLMs are designed.

The datasets we used contain no information that names or uniquely identifies individual people or offensive content.

We use Generative AI only for writing correction.

References

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

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

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

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

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yezeng Chen, Zui Chen, and Yi Zhou. 2024a. Brain-inspired two-stage approach: Enhancing mathematical reasoning by imitating human thought processes. *arXiv preprint arXiv:2403.00800*.

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

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind

Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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

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

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

Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. 2024. Towards analyzing and understanding the limitations of dpo: A theoretical perspective. *arXiv preprint arXiv:2404.04626*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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

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

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457*.

Hyeonbin Hwang, Doyoung Kim, Seungone Kim, Seonghyeon Ye, and Minjoon Seo. 2024. Self-explore to avoid the pit: Improving the reasoning capabilities of language models with fine-grained rewards. *arXiv preprint arXiv:2404.10346*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Dahyun Kim, Yungi Kim, Wonho Song, Hyeonwoo Kim, Yunsu Kim, Sanghoon Kim, and Chanjun Park. 2024. sdpo: Don't use your data all at once. *arXiv preprint arXiv:2403.19270*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

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

Zimu Lu, Aojun Zhou, Ke Wang, Houxing Ren, Weikang Shi, Junting Pan, and Mingjie Zhan. 2024. Step-controlled dpo: Leveraging stepwise error for enhanced mathematical reasoning. *arXiv preprint arXiv:2407.00782*.

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

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *arXiv preprint arXiv:2402.14830*.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*.

Jan Peters and Stefan Schaal. 2007. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pages 745–750.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

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

Zhiqing Sun, Yikang Shen, Hongxin Zhang, Qinhong Zhou, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. 2023. Salmon: Self-alignment with principle-following reward models. *arXiv preprint arXiv:2310.05910*.

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

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Y Wu, and Zhifang Sui. 2023. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *CoRR, abs/2312.08935*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *arXiv preprint arXiv:2402.10200*.

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

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. 2024. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. 2024. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

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

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

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *arXiv e-prints*, pages arXiv–2404.

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

A Used Prompt

A.1 Prompt for Yi to generate rationales

user: Please answer the following single-choice question by presenting the thinking process and presenting the answer. 1. The question has an answer. 2. The thinking process part is a coherent paragraph. 3. Present the answer in the end of the response which is in the format of T̈he answer is A/B/C/D.:

Question:

[present question here]

Choice:

[present choice here]

assistant:

A.2 Prompt for base models to generate CoT answer for GSM8K

Below is an instruction that describes a task.

"Write a response that appropriately completes the request.

Instruction:

[present query here]

Response:

A.3 Prompt for base models to generate CoT answer for Commonsense choosing task

Below is an instruction that describes a task.

Write a response that appropriately completes the request.

Instruction:

Pick the most correct option to answer the following question.

[present question here]

A.[present choice here]

B.[present choice here]

C.[present choice here]

D.[present choice here]

Response:

A.4 Prompt for analysis the False Positive Samples

You are an accurate answer evaluator. Your task is to determine whether a candidate answer is genuinely correct based on the question I provide and the reference answer. Key notes:

1. The reference answer is always correct, and the candidate answer to be evaluated will always have the correct final result.

2. You must evaluate whether the reasoning process of the candidate answer is correct.

3. The candidate answer does not need to match the reference answer verbatim—it only needs to be logically self-consistent.

4. If the candidate answer contains calculation errors, formula mistakes, or flawed logic (even if the final result matches the reference answer), it must be judged as incorrect.

5. Format your response strictly as:

{"conclusion": "correct/incorrect"}

Question

[present question here]

Reference Answer

[present reference answer here]

Candidate Answer

[present answer here]

B The Time Cost of EPO

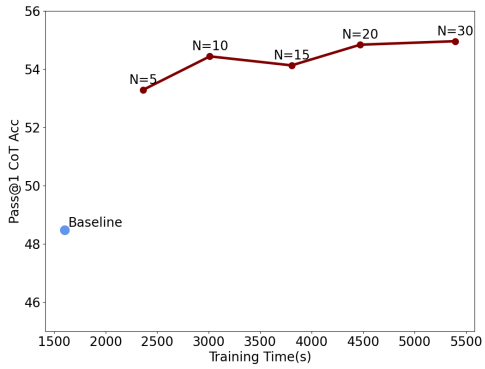


Figure 5: Analysis of training cost of EPO and baseline (i.e. DPO) under different N along with their performance.

The training cost involves time cost and memory costs. For the former, taking the sample of 20 responses per prompt, EPO requires the LLM to process an input that is 10 times larger than other methods (20 to 2). Benefiting from CUDA's parallel strategy for tensors, the extra time cost we need to bear is smaller than the linear estimation. For the latter, the extra GPU memory cost by a larger input tensor is much smaller than that is required for LLM training.

We present the relevance of training costs and the performance of our EPO. As it is shown in Fig 5, EPO's training time is less than 3 times of the other methods (while N is less than 30), while requiring a small amount of extra GPU memory.

C Proof for optimal solution to EPO

C.1 Proof for optimal solution to EPO

We construct our proof following the previous works(Peters and Schaal, 2007; Rafailov et al., 2024). From Eq. 2, our optimizing target is:

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi}[r(x,y)] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi(y \mid x)\|\pi_{\mathrm{ref}}(y \mid x)\right] \tag{11}$$

Notably, we can derive as:

$$\begin{aligned}
&\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi}[r(x,y)] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi(y \mid x)\|\pi_{\mathrm{ref}}(y \mid x)\right] \\
&\quad = \max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[r(x,y) - \beta\log\frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}\right] \\
&\quad = \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} - \frac{1}{\beta}r(x,y)\right] \\
&\quad = \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y \mid x)}{\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]
\end{aligned} \tag{12}$$

where we define as :

$$Z(x) = \sum_{y} \pi_{\mathrm{ref}}\left(y \mid x\right)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{13}$$

Notably, $Z(x)$ is a function of only x and π_{ref}. We can additionally define:

$$\pi^*(y \mid x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{14}$$

As is a probability distribution which holds $\sum_y \pi^*(y \mid x) = 1$. Using the $Z(x)$, we can re-organize the Eq. 11 as:

$$\begin{aligned}
&\min_{\pi}\mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y \mid x)}{\pi^*(y \mid x)}\right] - \log Z(x)\right] = \\
&\min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{D}_{\mathrm{KL}}\left(\pi(y \mid x)\|\pi^*(y \mid x)\right) - \log Z(x)\right]
\end{aligned} \tag{15}$$

Since $Z(x)$does not depend on π, the optimal solution is achieved by the policy that minimizes the first term. The KL divergence is minimized in the situation where two distributions are equal. Thus we have the optimal solution:

$$\pi(y \mid x) = \pi^*(y \mid x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{16}$$

C.1.1 Deriving the EPO Objective Under the Bradley-Terry Model

To derive the EPO objective under the Bradley-Terry preference model, we have the origin Bradley-Terry Model:

$$p^*\left(G_w \succ G_l \mid x\right) = \frac{1}{1 + \exp\left(\mathbb{E}_{y_i\sim G_l}\left[r\left(x, y_i\right)\right] - \mathbb{E}_{y_i\sim G_w}\left[r\left(x, y_i\right)\right]\right)} \tag{17}$$

In Eq. 6, we have:

$$r(x,y) = \beta\log\frac{\hat{\pi}(y \mid x)}{\pi_{\mathrm{ref}}\left(y \mid x\right)} + \beta\log Z(x) \tag{18}$$

Substituting Eq. 18 into Eq. 17, we can get:

$$
\begin{aligned}
p^*\left(G_w \succ G_l \mid x\right) &= \frac{1}{1+\exp\left(\mathbb{E}_{y_i \sim G_l}\left[r\left(x, y_i\right)\right]-\mathbb{E}_{y_i \sim G_w}\left[r\left(x, y_i\right)\right]\right)} \\
&= \frac{1}{1+\exp\left(\mathbb{E}_{y_i \sim G_l}\left[\beta \log \frac{\hat{\pi}(y_i|x)}{\pi_{\text{ref}}\left(y_i|x\right)}+\beta \log Z(x)\right]-\mathbb{E}_{y_i \sim G_w}\left[\beta \log \frac{\hat{\pi}(y_i|x)}{\pi_{\text{ref}}\left(y_i|x\right)}+\beta \log Z(x)\right]\right)} \\
&= \frac{1}{1+\exp\left(\mathbb{E}_{y_i \sim G_l}\left[\beta \log \frac{\hat{\pi}(y_i|x)}{\pi_{\text{ref}}\left(y_i|x\right)}\right]-\mathbb{E}_{y_i \sim G_w}\left[\beta \log \frac{\hat{\pi}(y_i|x)}{\pi_{\text{ref}}\left(y_i|x\right)}\right]\right)} \\
&= \sigma\left(\mathbb{E}_{y_i \sim G_l}\left[\beta \log \frac{\hat{\pi}\left(y_i \mid x\right)}{\pi_{\text{ref}}\left(y_i \mid x\right)}\right]-\mathbb{E}_{y_i \sim G_w}\left[\beta \log \frac{\hat{\pi}\left(y_i \mid x\right)}{\pi_{\text{ref}}\left(y_i \mid x\right)}\right]\right)
\end{aligned}
\tag{19}
$$

Which leads to Eq. 7.

D Implementation Details

D.1 Baselines

In this section, we present the details of the baselines we used compared to EPO. Notably, we are using different training methods in the self-training scenario. Thus all of our baselines start from the **SFT** model:

SFT presents the π_{SFT} which is the LLM fine-tuned on typical rationales for specific tasks. It is used as the initialization of each self-training method below and our EPO.

Beyond the **SFT** model, we utilize several self-training methods that do not introduce additional supervising information as our EPO does. The below methods are all beyond **SFT** model and the inference responses $\hat{\mathcal{D}}$ sampled from **SFT** model and the certain dataset:

(SFT +) **RFT** presents the model fine-tuned on the correct generated responses based on π_{SFT}, referring to the RFT method. Notably, we get a subset of $\hat{\mathcal{D}}$ using the correction of responses as the filtering signal, denoted as $\hat{\mathcal{D}}_{RFT}$. **RFT** are fine-tuned on $\mathcal{D} \cup \hat{\mathcal{D}}_{RFT}$(Yuan et al., 2023). This method stands for the performance of fine-tuning in the self-improving scenario.

(SFT +) **DPO** presents the fine-tuned model using typical DPO on the pair-wise preference samples which are randomly chosen once for each prompt. Notably, we sample one correct response and one incorrect response for each prompt in $\mathcal{D} \cup \hat{\mathcal{D}}$(Yuan et al., 2023) randomly. Then we apply DPO to this dataset. It has the same optimizing steps as our EPO.

(SFT +) **DPO**$_{batch}$ presents the model using DPO training on pairs selected as many as possible to the prompt (while ensuring the single utilization of each response) in G_l and G_w for each prompt. Notably, for each prompt in $\mathcal{D} \cup \hat{\mathcal{D}}$, we sample $min(Num_{right}, Num_{wrong})$ preference pairs as Num_{right} and Num_{wrong} represent the number of correct and incorrect responses. It shows the performance of using batched DPO compared to EPO.

(SFT +) **RPO** represents the model using the RPO algorithm (combining DPO loss with an NLL loss on the preferred response) on the pair-wise preference samples same as SFT + DPO. Notably, the RPO objective is represented as:

$$
\mathcal{L}_{RPO} = -\log \sigma\left(\beta \log \frac{M_\theta\left(c_i^w, y_i^w \mid x_i\right)}{M_t\left(c_i^w, y_i^w \mid x_i\right)}-\beta \log \frac{M_\theta\left(c_i^l, y_i^l \mid x_i\right)}{M_t\left(c_i^l, y_i^l \mid x_i\right)}\right)-\alpha \frac{\log M_\theta\left(c_i^w, y_i^w \mid x_i\right)}{\left|c_i^w\right|+\left|y_i^w\right|}
$$

(SFT +) **Step-DPO** represents the model using the Step-DPO algorithm(Lai et al., 2024) on the step-level pair-wise preference samples. We construct the step-level samples for each wrong responses using in (SFT +) DPO.

D.2 Hyperparameters

For the **SFT** training setups, we train SFT models using the following hyperparameters: learning rate of 2e-5, batch size of 64, max sequence length of 2048, and cosine learning rate schedule with 10% warmup steps for 3 epochs. All the models are trained with an Adam optimizer (Kingma and Ba, 2017). This setting is also the same for **RFT**.

For the preference optimization **DPO**, **DPO**$_{batch}$, **RPO** and **EPO**. We apply a search on the learning rate, training epoch, and additional hyperparameters. The search range is presented as below:

D.3 Search range of Baselines

Table 5: Hyperparameter search range.

Methods	Search Range
DPO	$\beta \in [0.05, 0.1, 0.5, 1.0]$ $lr \in [1e-7, 2e-7, 5e-7, 1e-6]$
DPO$_{batch}$	$\beta \in [0.05, 0.1, 0.5, 1.0]$ $lr \in [1e-7, 2e-7, 5e-7, 1e-6]$
RPO	$\beta \in [0.05, 0.1, 0.5, 1.0]$ $lr \in [1e-7, 2e-7, 5e-7, 1e-6]$ $\alpha \in [0.25, 0.5, 1, 2]$
EPO	$\beta \in [0.05, 0.1, 0.5, 1.0]$ $lr \in [1e-7, 2e-7, 5e-7, 1e-6]$ $\gamma \in [0.1, 0.2, 0.5, 1.0]$

Notably, we are referring the papers (Rafailov et al., 2024; Yuanzhe Pang et al., 2024; Meng et al., 2024) to set the search ranges. The length limitation of EPO is tuned from 5 to 100.

D.4 Implement Details

The experiments are carried out on 16 A100-80G GPUs with a Linux system. For all methods, we search the hyperparameters as we present the details in the Appendix. We train 3 epochs in each setting and report the performance of the best checkpoint. For the response generation phase in the self-improving scenario, we use the sample number N = 20 with temperature T = 0.7 following (Yuanzhe Pang et al., 2024). We use *Pytorch*[1] and *Huggingface*[2] as tools for the implementation. For preference optimization, we run our experiments based on *trl*[3]. All the generations were done using *vllm* (Kwon et al., 2023)[4]. The code will be released on GitHub[5].

[1]https://pytorch.org/
[2]https://huggingface.co/
[3]https://github.com/huggingface/trl
[4]https://github.com/vllm-project/vllm
[5]http://github.com/xxxxxx