Reward Alignment Optimization: A Direct Point-wise Alignment Approach

Anonymous ACL submission

Abstract

Direct Alignment Algorithms (DAAs) such as DPO simplify RLHF by optimizing policies directly from preference pairs. However, the Bradley–Terry probability-gap objective can induce likelihood displacement and, under weak KL constraints, may even reduce the probability of preferred responses, while implicit rewards can be limited in generalizaiton.. We propose Reward Alignment Optimization (RAO), a point-wise direct alignment method that uses an explicit reward model to specify exact target generation probabilities and align the policy offline towards them. Our key insight is a theoretical principle we call "prefix consistency", which links the normalization terms of prompts that share a prefix. Leveraging this property, RAO decouples target reward differentials from bias terms, prevents decreasing preferred-response probabilities, and better exploits reward information both within and across prompts. Extensive experiments on multiple base LLMs show that RAO consistently outperforms existing DAAs while enabling controllable target probability distributions.

1 Introduction

Large Language Model (LLM) alignment is designed to enhance the model's ability to align with human values and preferences. A typical LLM alignment approach is Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022). RLHF relies on annotated preference data (i.e. positive and negative response pairs) to model human preferences through the Bradley-Terry (BT) model (Bradley and Terry, 1952). Despite its strong performance, the RLHF pipeline is complex, involving the training of multiple LLMs and sampling processes within the training loop. Therefore, simpler alignment methods, known as Direct Alignment Algorithms (DAAs), are becoming widely adopted (Gupta et al., 2025).

DAAs primarily include Direct Preference Optimization (DPO) (Rafailov et al., 2024) and various adaptations of it. DPO reparameterizes the reward function within the RLHF framework, showing that the target policy can be viewed as defining an implicit reward function. Pairwise BT training eliminates the normalization term in this implicit function, which cannot be calculated directly.

Although DPO shares the same optimization objective as RLHF and achieves comparable performance, the BT training paradigm (i.e., optimizing the probability gap) introduces several problems (Meng et al., 2024; Sharifnassab et al., 2024; Lin et al., 2024). First, it optimizes the implicit reward model differently from the explicit reward model, which can distort the generation distribution of the policy model (i.e., likelihood displacement). When a weaker KL constraint is used, DPO may simultaneously reduce the probabilities of preferred and dispreferred responses (Meng et al., 2024; Hong et al., 2024), which is undesirable. Current approaches tend to mitigate this issue by assigning different weights to the preferred and dispreferred responses in the training objective (Gupta et al., 2025; Hong et al., 2024). However, this breaks the consistency between the objectives of DPO and RLHF.

Second, the structure of the implicit reward model can limit the effectiveness of the BT training paradigm and exhibits limited generalization (compared to explicit reward model training in RLHF) (Lin et al., 2024). Meanwhile, the implicit reward model used in the BT training paradigm cannot explicitly represent the Chain-of-Thought process (Zhang et al., 2024). Consequently, its reward modelling capability is inadequate for complex tasks (i.e., reasoning tasks). Introducing an explicit reward model into the alignment process (Adler et al., 2024; Fisch et al., 2024) provides a potential solution. However, these works still face likelihood displacement and the limited capability

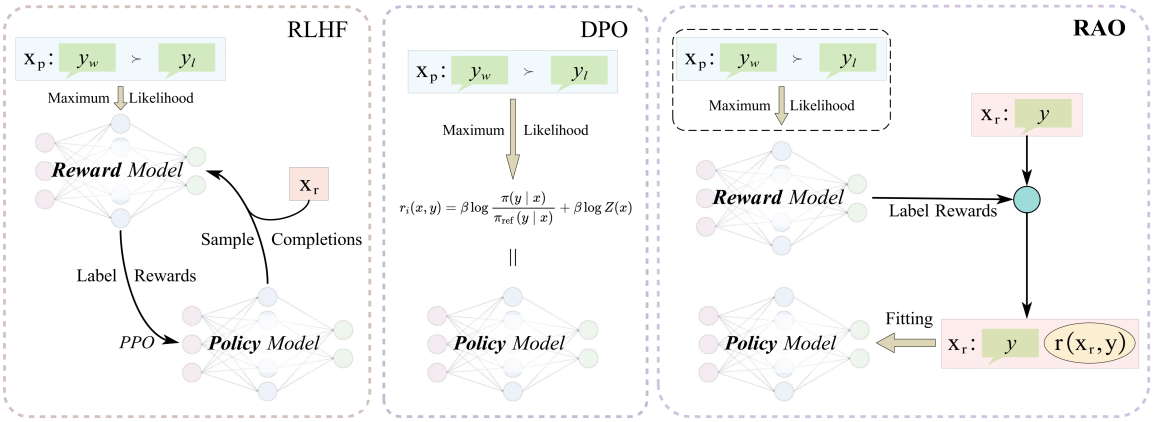


Figure 1: Differences of the proposed RAO from RLHF and DPO. RLHF trains a reward model using the BT model and applies PPO to optimize the policy model online. DPO uses the BT model to optimize the policy model offline. RAO uses a reward model (which may be trained by the BT model) to annotate the responses and offline distills the reward to the policy model.

of implicit reward models.

Unlike the pairwise BT model, reward alignment methods (Team et al., 2025; Richemond et al., 2024) provides the normalization term in the reward function, so that a target probability can be specified for the policy model. An MSE loss is then used to optimize the model toward this target. Determining the normalization value has given rise to different concrete methods. While some approaches (Team et al., 2025; Richemond et al., 2024) train an additional model to provide this value alongside the policy model, others (Su et al., 2025; Faye et al., 2025; Tang et al., 2025) estimate it from limited subsets of the training data (or via on-policy sampling). However, these methods either sacrifice the simplicity and theoretical consistency of DPO or fail to resolve the issues of the BT training paradigm (i.e., likelihood displacement).

In this paper, we aim to address these limitations from a new perspective by asking: **Can a DAA optimize the policy by directly specifying an accurate and reliable target generation probability of the response?** We observe that existing methods struggle to handle the normalization terms. Through theoretical analysis, we show that these terms exhibit a "prefix value consistency": for prompts that share the same prefix, their normalization values are related through the optimal policy, such that the normalization for one prompt can be derived from that of another. This implies that the normalization term can be determined by an invariant value and the optimal policy. Treating this value as a hyperparameter, we propose a Reward Alignment Optimization (RAO) algorithm, which uses an explicit reward model to optimize the policy offline via point-wise distillation while introducing no additional training or sampling cost compared to DPO. This derivation decouples the target reward differentials from the offsets in the optimization objective.

Compared to current DAAs, RAO leverages an explicit reward model (which estimates human preferences) to define an exact target probability for each response, compute the corresponding normalization terms, and optimize the policy using a point-wise MSE loss. RAO obtains several advantages. First, it naturally avoids the decrease in probability assigned to preferred responses. In practice, decoupling the target reward provides control over the policy model's target probability distribution and offers greater flexibility in adjusting optimization targets. Moreover, RAO makes better use of the explicit reward model than previous approaches by exploiting not only comparisons among responses for the same prompt but also reward information across different prompts. Meanwhile, the simplicity of DPO is preserved. Furthermore, RAO supports reward models in various forms (e.g., GenRM or rule-based reward).

In summary, the main innovation of the proposed RAO lies in a novel point-wise direct alignment algorithm that leverages an explicit reward model. The model decouples the target reward differentials from bias terms and more fully exploits reward information. Extensive experiments show that RAO substantially outperforms existing methods across a

range of base LLMs, including Llama3-8B (Dubey et al., 2024), Qwen2.5-7B (Yang et al., 2024), and EuroLLM-9B (Martins et al., 2024).

2 Preliminaries

Given a large language model parameterized by θ, donated as π_θ. The current alignment algorithms aim to optimize π_θ by learning from annotated preference pairs.

RLHF: RLHF (Bai et al., 2022) fits a reward model to pairwise samples of human preferences and then uses Proximal Policy Optimization (PPO) to optimize a language model policy to produce responses that are assigned a higher 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 i^{th} prompt, y_w^i and y_l^i respectively represent the preferred and dis-preferred responses to x_i. RLHF begins with modeling the probability of preferring y_w^i to y_l^i using the Bradley-Terry (BT) 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 (that is, the LLM classifier) that gives the estimation of y_i with respect to x_i according to human preference. Using maximum likelihood estimation to estimate the parameters of this function, we can optimize the classifier by the negative log-likelihood loss as below:

$$\begin{aligned}&\mathcal{L}_R\left(r_\phi, \mathcal{D}\right) = \\ &-\mathbb{E}_{\mathcal{D}}\left[\log\left(\sigma\left(r_\phi\left(x, y_w\right) - r_\phi\left(x, y_l\right)\right)\right)\right]\end{aligned} \quad (2)$$

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

$$\max_{\pi_\theta} \mathbb{E}\left[r_\phi(x, y)\right] - \beta \mathbb{D}_{\mathrm{KL}}\left[\pi_\theta \| \pi_{\mathrm{ref}}\right] \quad (3)$$

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

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

$$\begin{aligned}&\mathcal{L}_{\mathrm{DPO}}\left(\pi_\theta; \pi_{\mathrm{ref}}\right) = -\mathbb{E}_{(x, y_w, y_l) \sim D} \log \sigma \\ &\left(\beta \log \frac{\pi_\theta\left(y_w \mid x\right)}{\pi_{\mathrm{ref}}\left(y_w \mid x\right)} - \beta \log \frac{\pi_\theta\left(y_l \mid x\right)}{\pi_{\mathrm{ref}}\left(y_l \mid x\right)}\right)\end{aligned} \quad (4)$$

Notably, this optimization objective is based on a theoretical optimal π_θ beyond $r_U(x, y)$, which enables its equivalence with Eq.3.

3 Reward Alignment Optimization

This section presents the proposed Reward Alignment Optimization (RAO) algorithm. To guide the policy LLM toward the desired response probabilities, we derive our training objective from RLHF, following prior work (Rafailov et al., 2024), and introduce an explicit reward model. Using a "prefix consistency" criterion to determine the normalization term, RAO distills rewards from the explicit model into an implicit reward for the policy LLM.

3.1 Reward Model

RAO uses a reward model to distill rewards from an offline dataset into the policy LLM, encouraging the policy to approach the optimal solution of Eq. 3. Compared with DAA, which learns an implicit reward model via the Bradley-Terry (BT) training paradigm, RAO can rely on an explicit reward model with stronger generalization. In addition, our point-wise objective leverages reward relations across responses from different prompts, rather than the pair-wise formulation used in DAA.

Notably, RAO does not depend on a specific reward-model training procedure: it can optimize against GenRM or a rule-based reward model. In our experiments, however, we train the reward model following the standard RLHF setup, using a Bradley–Terry model over a pair-wise preference dataset (Rafailov et al., 2024), so that we can focus on evaluating the advantages of RAO in alignment optimization rather than the reward model itself. Specifically, we use Eq. 2 to train a neural reward model, where a classifier is applied to the hidden state of the last token produced by a pretrained LLM.

3.2 Reward Alignment Optimization

Starting from the RLHF objective, we follow prior work (Rafailov et al., 2024) and construct the im-

plicit reward function associated with the optimal policy $\hat{\pi}$ of Eq. 3 as follows:

$$r(x, y) = \beta \log \frac{\hat{\pi}(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x) \quad (5)$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y)\right)$ is the normalization term. Due to space limitations, we provide the detailed derivation in Appendix A.1.

The normalization term $Z(x)$ depends on the prompt x, which implies that the implicit reward target requires the set $\mathbf{Z} = \{Z(x_1), Z(x_2), ..., Z(x_n)\}$. Noting that the reward assigned to the concatenated string "xy" is invariant to where we split it into x and y, we can derive a relationship between $Z(x, y)$ (i.e., $Z("xy")$ corresponding to $r(x, y)$) and $Z(x)$ as follows (see Appendix C for the detailed derivation):

$$\frac{Z(x, y)}{Z(x)} = \frac{\hat{\pi}(y \mid x)}{\pi_{ref}(y \mid x)} \quad (6)$$

We refer to this relationship as "prefix consistency", since it suggests that different normalization terms can be determined through a shared prefix. Based on this relationship, we introduce an imaginary universal prefix t_0 that is shared by every prompt x_i. We then define the normalization term $Z_0 = Z(t_0)$, whose definition is $Z_0 = \sum_y \pi_{\text{ref}}(y \mid t_0) \exp\left(\frac{1}{\beta} r(t_0, y)\right)$. This shows that the relationships among $\mathbf{Z}$ are governed by $\hat{\pi}$ and π_{ref}. Once we obtain $Z(x_i)$, we can optimize the policy using the MSE loss:

$$\mathcal{L}_{RAO}(\pi_\theta, r, Z; \mathcal{D}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[(r(x, y) - \beta \bar{r})\right]^2]$$
$$\bar{r} = \log \frac{\pi(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + Z_0 \frac{\hat{\pi}(x_i - t_0 \mid t_0)}{\pi_{ref}(x_i - t_0 \mid t_0)} \quad (7)$$

3.3 Optimization

RAO distills explicit rewards to improve the LLM policy. Following (Adler et al., 2024), we include more than one response per prompt during training to provide stronger supervision. Notably, the assumption of Z_0 introduces a universal prefix t_0 that is shared by every prompt x_i; consequently, RAO requires prompts to start with the same token. This condition is typically satisfied in practice because most LLM templates fix the initial token (e.g., "⟨|im_start|⟩" or "User").

Algorithm 1: Reward Alignment Optimization

Input: SFT model π_θ, Reward model r, Training Data $\mathcal{D}$, Norm Value Z_0 with t_0, Training Epochs T, Learning Rate η

Output: Optimized Policy $\hat{\pi}_\theta$

$\pi_{ref} \longleftarrow \pi_\theta$;
foreach *Epoch t=1, 2, ..., T* **do**
  Get a batch of samples $\mathcal{D}_\sqcup \subset \mathcal{D}$;
  foreach $(x_i, y_i^1, y_i^2, ...) \in \mathcal{D}_\mathcal{B}$ **do**
    $\mathcal{L}_\mathcal{T} \longleftarrow 0$;
    $Z_i \longleftarrow \frac{\pi_\theta(x_i - t_0 | t_0)}{\pi_{ref}(x_i - t_0 | t_0)} Z_0$;
    Detach Z_i ;
    foreach y_i^j **do**
      $r_j \longleftarrow \beta \log \frac{\pi_\theta(y_i^j | x_\mathcal{T})}{\pi_{\text{ref}}(y_i^j | x_i)} + \beta \log Z_i$;
      $\mathcal{L}_\mathcal{T} \longleftarrow \mathcal{L}_\mathcal{T} + \left(r(x_i, y_i^j) - r_j\right)^2$;
    $\pi_\theta \longleftarrow \pi_\theta - \eta \nabla \left(\frac{\mathcal{L}_\mathcal{T}}{|\text{the number of } y \text{ for } x_i|}\right)$;
$\hat{\pi}_\theta \longleftarrow \pi_\theta$;

Theorem 3.1. *Suppose a reward model $r(x, y)$ and a certain dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$, infinite various $\bar{r}_r$ can be constructed ensuring: 1. $r(x_i, y_i) = \bar{r}_r(x_i, y_i)$ for $x_i, y_i \in \mathcal{D}$. 2. For all x_1, y_1, x_2, y_2 in the language space, $[r(x_1, y_1) - r(x_2, y_2)][\bar{r}_r(x_1, y_1) - \bar{r}_r(x_2, y_2)] > 0$*

Why Z_0 can be regarded as a hyperparameter? The actual value of Z_0 cannot be computed from its definition because the space of possible y is unbounded. Therefore, in RAO we treat Z_0 as a hyperparameter. As shown by our derivation in App. D (proving Thm. 3.1), multiple reward functions can correspond to different Z_0 values while inducing the same reward values on the observed data. Consequently, Z_0 can be viewed as a property of an equivalent class of reward functions that does not change the optimization objective (i.e., the reward distribution over the given responses). In practice, Z_0 acts as a knob that shifts reward magnitudes in RAO; we analyze this effect in Sec. 4.5. During optimization, we approximate $\hat{\pi}$ with π_θ to maintain consistency with the optimal solution of RAO. Our experiments further confirm that this approximation does not harm convergence.

Algorithm 1 summarizes the RAO optimization procedure. RAO optimizes the policy's implicit reward objective while treating the normalization term Z_i as a constant (i.e., detaching Z_i from the computation graph). After forming the target for $\log \hat{\pi}(y \mid x)$, RAO trains the policy using an MSE loss, following (Fisch et al., 2024).

3.4 The Interpretation of RAO

Our RAO uses Eq. 6 to construct an approximation to the normalization term in Eq. 5, and optimizes the policy with an MSE loss. By substituting the expression derived from Eq. 6 into Eq. 5 (via $Z(t_0)$), we obtain the following form:

$$r(x, y) = \beta \log \frac{\hat{\pi}(t_0 \mid (x, y) - t_0)}{\pi_{\text{ref}}(t_0 \mid (x, y) - t_0)} + \beta \log Z(t_0) \tag{8}$$

Eq. 8 can be viewed as a special case of Eq. 5 under this substitution. Importantly, in Algo. 1 we treat Z_i as a constant, so it does not contribute gradients. Because prompt-generation probabilities fall within the optimization scope, this yields an optimization procedure that differs from directly optimizing Eq. 8.

Since Eq. 8 shares the same optimal policy as RAO, it provides intuition for how hyperparameters shape the training signal. In particular, β controls the scale of reward differences in the target: smaller β leads to larger gaps among reward targets, consistent with (Rafailov et al., 2024). Moreover, Z_0 acts as an offset on the rewards: decreasing Z_0 shifts all reward targets upward. This helps RAO avoid simultaneous decreases in generation probabilities.

4 Experiments

We experiment with our RAO based on the below pretrained LLMs: Llama3-8B (Dubey et al., 2024), Qwen2.5-7B (Bai et al., 2023), and EuroLLM-9B (Martins et al., 2024). In this section, our aim is to present the advantages of our RAO versus other direct alignment baselines. We start from the base models and finetune them to gain the instruction-following capability. Reward models are trained on a pairwise preference dataset. Then, we use the reward models to annotate the rewards of this preference dataset, and RAO is used to optimize the finetuned LLMs. Notably, we keep sampling two responses each prompt in order to keep the scale of training data the same as RAO and all baselines. Notably, we put our Implementation Details in the App. B

Model Setting	Small		Large	
	Loss	Acc	Loss	Acc
RM-Base	0.0621	0.975	0.0539	0.982
RM-SFT	0.0463	0.979	0.035	0.988
DPO-Implicit	0.2039	0.9521	0.2463	0.9660

Table 1: The reward model training results.

4.1 Datasets and Evaluations

We follow the typical training pipeline of Zephyr (Tunstall et al., 2023) and SimPO (Meng et al., 2024) to select datasets. For the supervised fine-tuning phase, we apply the UltraChat-200k dataset (Ding et al., 2023) to train our base models. Notably, we optimize the base models utilizing the multi-turn dialogue templates of their chat derivatives. For reward model training and alignment optimization, we apply the UltraFeedback dataset (Cui et al., 2023). This approach provides a high level of reproducibility. We present an introduction of these datasets in the Appendix E.

For evaluation benchmarks, we apply the widely used benchmarks for general instruction-following capability: Alpaca-Eval2 (Dubois et al., 2024) and MT-Bench (Zheng et al., 2024). These benchmarks evaluate the LLM's versatile conversational capabilities utilizing different queries. Alpaca-Eval2 constructs its 805 queries from 5 datasets, and MT-Bench contains 80 queries sampled from 8 different categories. Both benchmarks rely on an LLM-as-judge evaluating methods. Notably, we use GPT-4 (Achiam et al., 2023) as the annotator for them. For Alpaca-Eval2, we present the results of win rate (WR) and length-controlled win rate (LC), which reflect the evaluation results eliminating the effect of model verbosity over a base response, which is sampled from GPT-4 Turbo (Achiam et al., 2023). For MT-Bench, we report the average overall score calculated based on the judgment of GPT-4.

4.2 Baselines

We compare our RAO with different direct alignment algorithm baselines. Except for the widely used and introduced **DPO** and **RLHF**, **IPO** (Azar et al., 2024) constructs a general preference learning structure objective, deriving from which DPO is a special case, bypasses the BT modelization assumption for preferences, and utilizes an MSE loss. **SimPO** (Meng et al., 2024) uses the average log probability of a sequence as the implicit reward and introduces a target reward margin in the DPO objective. **Robust Preference Optimization (RPO)**

(Fisch et al., 2024) introduces an explicit reward model to distill the reward gaps to the policy model. Notably, we use the same reward model to address the reward gaps as our RAO uses. We only use one reward model in RPO to ensure the fairness of our RAO and RPO. **Cal-DPO** (Xiao et al., 2024) introduces IPO-like calibration loss to DPO loss to solve likelihood displacement. **DRO** introduces extra value model to present the normalization term. Notably, except for IPO, all the above methods do not share the same optimal solution consistency as DPO and RAO with RLHF. We report the hyperparameter search area of each baseline in Appendix F.

4.3 Reward Model

Our RAO doesn't specify the approach of the reward model used to give the reward. Here we present a demonstrative reward model training process. We utilize the Bradley-Terry model to train an explicit reward model that gives a reward score through a randomly initialized classifier on the hidden state of the last token of a pretrained model's output. To compare the performances of explicit reward models initialized with the base model the SFT model and the implicit reward model indicated in Eq. 5, we utilize all the preference pairs in UltraFeedback (regarded as "large" setting) or 10000 pairs randomly sampled from it (regarded as "small" setting) either to train the reward models based on Llama3. Taking the loss of training ends and the metrics of reward accuracy (i.e., the accuracy of the reward model gives a larger reward to preferred responses than dispreferred ones) on the test set of UltraFeedback, we present the results in Tab. 1.

We can observe that the explicit reward model initialized by the SFT model performs best among the three. The either explicit model shows an apparent advantage to the implicit model. This indicates the benefits of using an explicit reward model for alignment as our RAO. Following the results, we train the reward model of Qwen2.5 and EuroLLM using their SFT model instead of directly using the base model. Specifically, we use the same reward model for RAO, DRO and RLHF.

4.4 Main Results

The main results of our experiments are presented in Tab. 2. Remarkably, while all the direct alignment baselines optimize the SFT model to a better conversational capability, referring to the benchmarks, RAO outperforms all the baselines in all settings except SimPO on EuroLLM-9B on the Alpaca-Eval 2 win-rate metric. This illustrates the advantages of RAO over current alignment methods. Notably, RAO achieves an 82.83% increase over the SFT model and a 5.04% increase over RPO, which performs best among the baselines in the Alpaca-Eval 2 win rate metric based on Llama3-8B, and this advantage comes to 73.47% and 12.31% on the length-controlled win rate. For Qwen2.5-7B, RAO gains 14.79% and 14.98% advantages compared to the best baseline on win rate and length-controlled win rate of Alpaca-Eval 2. For EuroLLM-9B, RAO gains a 6.29% advantage on the length-controlled win rate.

A cursory examination reveals that our RAO has an obvious outperformance over all the direct alignment baselines across all tasks. Such a pattern underscores the effectiveness of RAO in improving LLM's ability in preference learning. RAO not only introduces an explicit reward model that has a better generalization capability to the alignment training but also provides a more stable training target using point-wise loss and prevents the continual decrease of preferred response probabilities.

4.5 Analysis

We present here a detailed analysis of our RAO controls and the alignment process of the policy. As shown in Fig. 2, we conclude:

- RAO utilizes point-wise loss to optimize the policy model. It sets a target for the chosen reward of the policy model; thus, we can observe from Fig. 2(a) that the rewards of both chosen and rejected rewards are symmetrically separated from each other while keeping a clear, stable mean value. This mean value is the Z_0 value set to be stable in the training process. While Z_0 grows larger, the value drops.

- From another perspective, the effect of Z_0 and β in RAO is more clearer in Fig. 2(b). While Z_0 grows larger, the chosen reward of the training end decreases. While β grows smaller, this decreasing trend becomes slower. It can be inferred that when Z_0 is sufficiently larger, the chosen reward can be smaller than utilizing DPO.

- As for the gap between chosen rewards and rejected rewards in the training ends, β can

Methods	Llama3-8B			Qwen2.5-7B			EuroLLM-9B		
	AlpacaEval 2		MT-Bench	AlpacaEval 2		MT-Bench	AlpacaEval 2		MT-Bench
	WR(%)	LC(%)		WR(%)	LC(%)		WR(%)	LC(%)	
SFT	3.35	5.82	5.0	5.41	10.69	5.7	4.11	7.81	5.3
DPO	18.32	17.63	6.5	18.12	23.16	**6.8**	12.52	16.02	6.0
RLHF	16.08	15.97	6.3	16.78	19.54	6.5	12.71	15.85	6.0
IPO	14.92	15.24	6.1	13.25	14.47	6.4	11.38	11.98	5.8
RPO	18.52	19.24	**6.6**	17.74	22.14	6.6	14.24	14.59	6.1
Cal-DPO	18.51	17.02	6.5	18.27	22.09	6.7	13.25	16.72	6.1
DRO	13.25	13.38	6.2	15.35	17.58	6.5	12.33	14.69	6.0
SimPO	18.42	19.97	**6.6**	17.32	23.28	6.7	**14.92**	16.53	**6.2**
RAO	**19.51**	**21.94**	**6.6**	**20.82**	**26.04**	**6.8**	14.11	**17.64**	**6.2**

Table 2: Main Results on UltraFeedback Dataset. The best performance is in bold.

Methods	MMLU	GSM8K	ARC-Easy	ARC-Hard	MathQA	SocialQA	Avg.
SFT	63.81	25.84	52.82	48.29	26.73	50.25	44.62
DPO	64.88	24.84	49.37	39.25	**28.88**	37.45	41.45
RLHF	63.82	25.77	56.72	50.17	26.81	53.48	46.92
IPO	63.25	28.96	60.29	45.30	27.03	40.78	44.27
RPO	64.32	26.37	58.33	51.50	26.29	52.20	46.50
Cal-DPO	64.68	27.25	58.39	51.27	27.01	52.85	46.91
DRO	63.25	25.17	56.85	50.02	26.52	53.61	45.90
SimPO	63.47	25.02	44.57	36.6	25.42	36.83	38.65
RAO	**65.25**	**31.72**	**69.49**	**55.38**	27.19	**54.95**	**50.66**

Table 3: Overall result in the Downstream Tasks for Models trained on UltraFeedback. The best performance is in bold.

have a significant effect. While β drops, this gap grows rapidly. One of our RAO's main effectiveness is decoupling the reward gap and the mean value of the alignment target. It can be seen in Fig. 2(c) that Z_0 has little effect on the reward gap.

- From Fig. 2(d) we can observe that the performance of the alignment algorithm is affected by the combination of other factors. Neither reward gap nor the chosen reward can reflect the final performance independently.

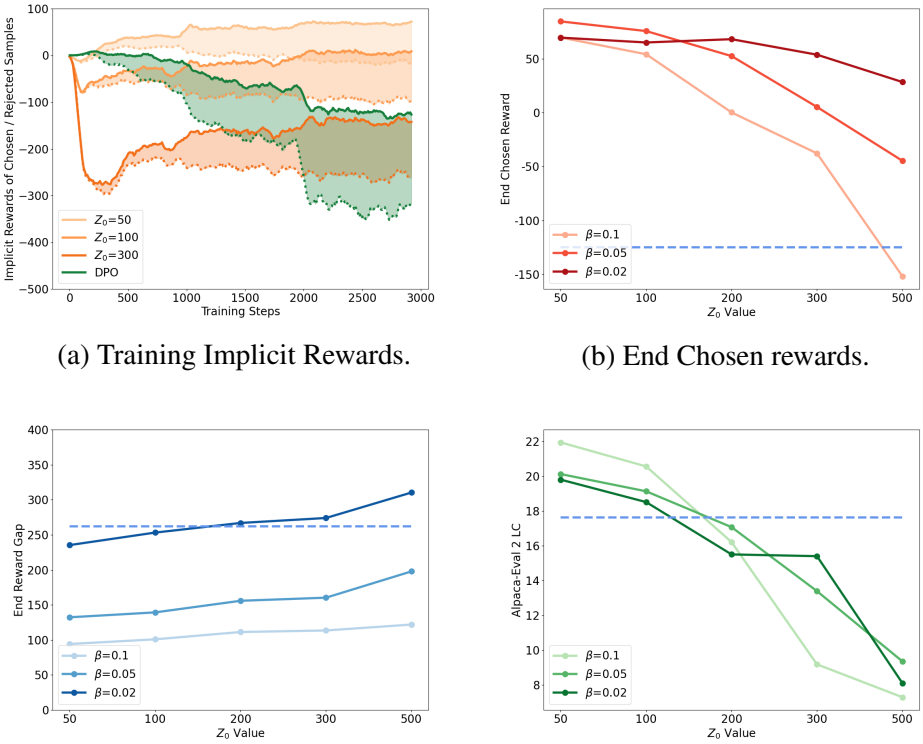
4.6 Downstream Tasks Evaluation

To examine how exactly the models perform in different fields, we evaluate all the models reported in Tab. 2 which is based on Llama3-8B to various downstream tasks. Specifically, we include the MMLU (Hendrycks et al., 2020), GSM8K (Cobbe et al., 2021), ARC-Easy and Challenge (Clark et al., 2018), MathQA (Amini et al., 2019), and SocialQA (Sap et al., 2019). As reported in (Meng et al., 2024), several direct alignment algorithms may RAOp the model's performance in reasoning tasks. Thus, we mainly choose the reasoning tasks in our evaluation and the widely used MMLU. Notably, except for MMLU, all the tasks are evaluated through the CoT Pass@1 zero-shot setting. We set the sampling temperature to 0.0 to adopt the greedy sampling method.

The results are presented in Tab. 3. We can observe that RAO performs better than all the baselines. While alignment methods as DPO and SimPO obviously drop the model's reasoning capabilities, RAO does not decrease the ability of the SFT model and instead improves the reasoning ability of the model through alignment. We infer that some baselines dropping the model's reasoning capability may be caused by the significant decrease of preferred response probabilities that the alignment methods do to the policy model. While "heavily" optimizing the model to align with human preference, the training process overfits the model and weakens its generalization ability. As a side note, the downstream tasks of the models trained by Cal-DPO and RLHF did not show significantly inferior performance compared to SFT. This proves the advantages of RAO.

5 Related Work

Large language models (LLMs) have shown great zero-shot and few-shot performance (Brown et al., 2020; Chowdhery et al., 2023; Radford et al., 2019).



(a) Training Implicit Rewards.

(b) End Chosen rewards.

(c) End Reward gap.

(d) End Performances.

Figure 2: Analysis of RAO training process. The analysis experiments are conducted on Llama3-8B under different hyperparameters. The blue dashed line represents the performance of DPO. The dotted line in (a) indicates the reject samples' reward and the solid line indicates the chosen samples' reward.

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 simple and computationally lightweight method with no need for additional reward function training.

Various further methods have been proposed to improve DPO. ORPO (Hong et al., 2024) and SimPO (Meng et al., 2024) focus on regularization of sequence length to tackle the issue that DPO tends to increase the response length of the policy model. For solving the likelihood displacement problem (discussed in Sec. 1), DPOP (Pal et al., 2024), KTO (Ethayarajh et al., 2024), and DPO-Shift(Yang et al., 2025) aim to lower the preferred response probabilities by increasing the weight of the preferred term in the training objective. However, these methods break the theoretical basis of DPO and obtain uncertain gains.

Exploring the point-wise optimization based on directly optimize the policy through the origin implicit reward function has raised several works. K1.5 (Team et al., 2025) and DRO (Richemond et al., 2024) train an extra value function to provide the normalization term. DGRO (Su et al., 2025) and RPO (Faye et al., 2025) present this term by an incomplete statistic. These works face the instability of the estimation and either drop the simplicity of DPO or facing the same problem of BT training paradigm.

Our RAO proposes a point-wise direct alignment method that better utilizes the reward model information and brings a strengthened control over the optimization objective.

6 Conclusion

In this paper, we propose a Reward Alignment Optimization (RAO) method that utilizes a point-wise target for aligning the model.

Compared to the existing direct alignment approaches. RAO prevents the policy model from dropping the generation probability and utilises a direct target to lead the optimization while introducing no extra training parameters. Experimental results on various reasoning tasks and datasets demonstrate the superior performance of our RAO which consistently outperforms a wide range of baseline approaches.

7 Limitations

Our paper presents a simple and effective method to align the LLMs to human performances. We present our experiments based on a typical trained Bradley-Terry model using exactly the same data used for alignment optimization. It would be better to discuss more about the reward models and do a more comprehensive experiment about the number of responses for each prompt used in the optimization as RAO doesn't restrict to the pairwise training structure.

8 Discussion of Ethical Considerations

Our proposed methods are used to improve the capabilities of LLMs. Using RAO 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 (Llama3-8B, Qwen2.5-7B, EuroLLM-9B) and the datasets (UltraChat, UltraFeedBack, 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 AI Assistants (i.e., Grammarly) only for check the words and polish sentences.

References

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

Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan Cohen, et al. 2024. Nemotron-4 340b technical report. *arXiv preprint arXiv:2406.11704*.

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*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

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*.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. 2024. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*.

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

Bilal Faye, Hanane Azzag, and Mustapha Lebbah. 2025. Value-free policy optimization via reward partitioning. *arXiv preprint arXiv:2506.13702*.

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*.

Adam Fisch, Jacob Eisenstein, Vicky Zayats, Alekh Agarwal, Ahmad Beirami, Chirag Nagpal, Pete Shaw, and Jonathan Berant. 2024. Robust preference optimization through reward model distillation. *arXiv preprint arXiv:2405.19316*.

Aman Gupta, Shao Tang, Qingquan Song, Sirou Zhu, Jiwoo Hong, Ankan Saha, Viral Gupta, Noah Lee, Eunki Kim, Jason Zhu, et al. 2025. Alphapo–reward shape matters for llm alignment. *arXiv preprint arXiv:2501.03884*.

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*.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189.

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*.

Yong Lin, Skyler Seto, Maartje Ter Hoeve, Katherine Metcalf, Barry-John Theobald, Xuan Wang, Yizhe Zhang, Chen Huang, and Tong Zhang. 2024. On the limited generalization capability of the implicit reward model induced by direct preference optimization. *arXiv preprint arXiv:2409.03650*.

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*.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. Generative reward models. *arXiv preprint arXiv:2410.12832*.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, and Chase Blagden1 Nathan Lile1 Louis Castricato. Generative reward models-a unified approach to rlhf and rlaif.

Pedro Henrique Martins, Patrick Fernandes, João Alves, Nuno M Guerreiro, Ricardo Rei, Duarte M Alves, José Pombal, Amin Farajian, Manuel Faysse, Mateusz Klimaszewski, et al. 2024. Eurollm: Multilingual language models for europe. *arXiv preprint arXiv:2409.16235*.

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*.

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

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.

Noam Razin, Sadhika Malladi, Adithya Bhaskar, Danqi Chen, Sanjeev Arora, and Boris Hanin. 2024. Unintentional unalignment: Likelihood displacement in direct preference optimization. *arXiv preprint arXiv:2410.08847*.

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

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*.

ByteDance Seed, Yufeng Yuan, Yu Yue, Mingxuan Wang, Xiaochen Zuo, Jiaze Chen, Lin Yan, Wenyuan Xu, Chi Zhang, Xin Liu, et al. 2025. Seed-thinking-v1. 5: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*.

Arsalan Sharifnassab, Saber Salehkaleybar, Sina Ghiassian, Surya Kanoria, and Dale Schuurmans. 2024. Soft preference optimization: Aligning language models to expert distributions. *arXiv preprint arXiv:2405.00747*.

Xuerui Su, Liya Guo, Yue Wang, Yi Zhu, Zhiming Ma, Zun Wang, and Yuting Liu. 2025. Dgro: Enhancing llm reasoning via exploration-exploitation control and reward variance management. *arXiv preprint arXiv:2505.12951*.

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*.

Yunhao Tang, Taco Cohen, David W Zhang, Michal Valko, and Rémi Munos. 2025. Rl-finetuning llms from on-and off-policy data with a single algorithm. *arXiv preprint arXiv:2503.19612*.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

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*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

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*.

Teng Xiao, Yige Yuan, Huaisheng Zhu, Mingxiao Li, and Vasant G Honavar. 2024. Cal-dpo: Calibrated direct preference optimization for language model alignment. *arXiv preprint arXiv:2412.14516*.

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*.

Xiliang Yang, Feng Jiang, Qianen Zhang, Lei Zhao, and Xiao Li. 2025. Dpo-shift: Shifting the distribution of direct preference optimization. *arXiv preprint arXiv:2502.07599*.

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.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

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 Deriving the optimal solution of RLHF

A.1 Proof for optimal solution of RLHF

We construct our proof following the previous works(Peters and Schaal, 2007; Rafailov et al., 2024). From Eq. 3, our optimizing target is:

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi}[r(x,y)] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi(y \mid x)\|\pi_{\mathrm{ref}}(y \mid x)\right] \tag{9}$$

Notably, we can derive that:

$$\begin{aligned}
&\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi}[r(x,y)] - \beta\mathbb{D}_{\mathrm{KL}}\left[\pi(y \mid x)\|\pi_{\mathrm{ref}}(y \mid x)\right] \\
&\quad= \max_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[r(x,y) - \beta\log\frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}\right] \\
&\quad= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} - \frac{1}{\beta}r(x,y)\right] \\
&\quad= \min_{\pi} \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y \mid x)}{\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right)} - \log Z(x)\right]
\end{aligned} \tag{10}$$

where we define as :

$$Z(x) = \sum_{y} \pi_{\mathrm{ref}}\left(y \mid x\right)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{11}$$

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

$$\hat{\pi}(y \mid x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)\exp\left(\frac{1}{\beta}r(x,y)\right) \tag{12}$$

As is a probability distribution which holds $\sum_y \pi^*(y \mid x) = 1$. Using the $Z(x)$, we can re-organize the Eq. 9 as:

$$\begin{aligned}
&\min_{\pi}\mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi(y|x)}\left[\log\frac{\pi(y \mid x)}{\hat{\pi}(y \mid x)}\right] - \log Z(x)\right] = \\
&\min_{\pi}\mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{D}_{\mathrm{KL}}\left(\pi(y \mid x)\|\hat{\pi}(y \mid x)\right) - \log Z(x)\right]
\end{aligned} \tag{13}$$

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 situations where two distributions are equal. Thus, we have the optimal solution:

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

B Implement Details

The experiments are carried out on 16 A100-80G GPUs with a Linux system. For all baselines and RAO, we search the hyperparameters as we present the details in the Appendix F. For the SFT phase, we train two epochs in each setting and report the performance of the best checkpoint. We train for three epochs for the alignment phase and take the same approach. We use *Pytorch*[1] and *Huggingface*[2] as tools for the implementation. For alignment, we apply experiments based on *trl*[3]. All the generations during the evaluation process 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

C Deriving the Equation 6

We here present our derivation process for Eq. 6. Let's start from Eq. 5. Assuming a reward process on the concatenated response (y_1, y_2) to the prompt x (i.e., the response of x is: y_1 followed by y_2), the reward of the implicit reward function is:

$$r(x, (y_1, y_2)) = \beta \log \frac{\hat{\pi}\left((y_1, y_2) \mid x\right)}{\pi_{\text{ref}}\left((y_1, y_2) \mid x\right)} + \beta \log Z(x) \tag{15}$$

Then, we can observe this equation from another perspective. Rewarding the prompt of x, y_1, y_2 as (x, y_1) and the response is y_2, the the reward of the implicit reward function is:

$$r((x, y_1), y_2)) = \beta \log \frac{\hat{\pi}\left(y_2 \mid (x, y_1)\right)}{\pi_{\text{ref}}\left(y_2 \mid (x, y_1)\right)} + \beta \log Z(x, y_1) \tag{16}$$

since $r(x, (y_1, y_2)) = r((x, y_1), y_2))$, we can derive from combining Eq. 15 and Eq. 16 into:

$$\begin{aligned} \beta \log \frac{\hat{\pi}\left(y_2 \mid (x, y_1)\right)}{\pi_{\text{ref}}\left(y_2 \mid (x, y_1)\right)} + \beta \log Z(x, y_1) &= \beta \log \frac{\hat{\pi}\left((y_1, y_2) \mid x\right)}{\pi_{\text{ref}}\left((y_1, y_2) \mid x\right)} + \beta \log Z(x) \\ \log \frac{\hat{\pi}\left(y_2 \mid (x, y_1)\right)}{\pi_{\text{ref}}\left(y_2 \mid (x, y_1)\right)} - \log \frac{\hat{\pi}\left((y_1, y_2) \mid x\right)}{\pi_{\text{ref}}\left((y_1, y_2) \mid x\right)} &= \beta \log Z(x) - \log Z(x, y_1) \\ \frac{Z(x, y_1)}{Z(x)} &= \frac{\hat{\pi}(y_1 \mid x)}{\pi_{ref}(y_1 \mid x)} \end{aligned} \tag{17}$$

Thus, we derive Eq. 6.

D Proof of Thm. 3.1

Suppose a reward model $r(x, y)$ and a response dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$. We can construct a reward function as:

$$\bar{r}_{r,\alpha_1,\alpha_2,\mathcal{D}}(x_i, y_i) = \begin{cases} \alpha_1 \left[r(x_i, y_i) - \delta_{\mathcal{D}}^{max}\right] + \delta_{\mathcal{D}}^{max} & \text{if } r(x_i, y_i) > \delta_{\mathcal{D}}^{max} \\ \delta_{\mathcal{D}}^{min} - \alpha_2 \left[\delta_{\mathcal{D}}^{min} - r(x_i, y_i)\right] & \text{if } r(x_i, y_i) < \delta_{\mathcal{D}}^{min} \\ r(x_i, y_i) & \text{if } \delta_{\mathcal{D}}^{min} < r(x_i, y_i) < \delta_{\mathcal{D}}^{max} \end{cases} \tag{18}$$

where

$$\begin{aligned} \delta_{\mathcal{D}}^{max} &= max\left(\{r(x_i, y_i) | (x_i, y_i) \in \mathcal{D}\}\right) \\ \delta_{\mathcal{D}}^{min} &= min\left(\{r(x_i, y_i) | (x_i, y_i) \in \mathcal{D}\}\right) \end{aligned} \tag{19}$$

With different setting of α_1 and α_1 ensuring $\alpha_1, \alpha_2 > 0$, different reward model $\bar{r}(r, \alpha_1, \alpha_2, \mathcal{D})$ can be constructed. Each constructed reward function $\bar{r}$ satisfies that the reward sorting of $\bar{r}$ is the same with r and the reward value of the responses in $\mathcal{D}$ is invariable (i.e., $\bar{r}_{r,\alpha_1,\alpha_2,\mathcal{D}}(x_i, y_i) = r(x_i, y_i)$). This proves the Theorem. 3.1. Specifically, different α_1 and α_1 lead to different Z_0, thus regarding Z_0 as a hyperparameter doesn't detract from the theoretical basis.

E Datasets

•UltraChat-200k is a multi-turn instructional conversation dataset that contains 207,865 conversations for training. UltraChat-200k is designed by a principle that attempts to capture the breadth of interactions that a human might have with an AI assistant and then employs meta-information, in-context expansion, and iterative prompting to scale up the number of instructions. The constructors use LLMs only to generate the conversations.

•UltraFeedback is a large-scale, high-quality, and diversified AI feedback dataset, which contains over 1 million GPT-4 feedback for user-assistant conversations from various aspects. It is constructed beyond a compiled diverse array of over 60,000 instructions and 17 models from multiple sources and then utilizes GPT-4 for annotation with a bunch of techniques to alleviate annotation biases and improve feedback

quality to the greatest extent. Notably, we utilize binary preferences from UltraFeedback by selecting the highest mean score as the preferred response and one of the remaining three at random as dispreferred, referring to (Tunstall et al., 2023). The total number of data pairs for training is 61,135.

F HyperParameter Search

Table 4: Hyperparameter search range.

Methods	Search Range
DPO	$\beta \in \{0.05, 0.1, 0.5, 1.0\}$ $lr \in \{1e-7, 2e-7, 5e-7, 1e-6\}$
SLiC-HF	$\lambda \in \{0.05, 0.1, 0.5, 1.0, 5, 0\}$ $lr \in \{1e-7, 2e-7, 5e-7\}$
IPO	$\beta \in \{0.05, 0.1, 0.5, 1.0\}$ $lr \in \{1e-7, 2e-7, 5e-7, 1e-6\}$ $\alpha \in \{0.25, 0.5, 1, 2\}$
ORPO	$\tau \in \{0.01, 0.05, 0.1, 1.0\}$
SimPO	$\beta \in \{1.0, 2.0, 2.5\}$ $\gamma \in \{0.3, 0.5, 0.7, 1.0, 1.5\}$
RPO	$\beta \in \{0.05, 0.1, 0.5, 1.0\}$
RLHF	$lr \in \{1e-7, 2e-7\}$ $\beta \in \{0.05, 0.1\}$ clip ratio $= 0.2$
RPO	$lr \in \{1e-7, 2e-7\}$ $\tau = 1.0$
RAO	$\beta \in \{0.05, 0.1, 0.5, 1.0\}$ $lr \in \{1e-7, 2e-7, 5e-7, 1e-6\}$ $Z_0 \in [-50, 500]$

Notably, we are referring to the papers (Rafailov et al., 2024; Meng et al., 2024; Hong et al., 2024; Azar et al., 2024; Zhao et al., 2023) to set the search ranges. For our RAO, we report the results under learning rate 5e-7, beta 0.1 with Z_0 50 in our paper (which is also mentioned in the Experiments Section in the paper).