WPO: Enhancing RLHF with Weighted Preference Optimization

Wenxuan Zhou[†], **Ravi Agrawal, Shujian Zhang, Sathish Reddy Indurthi**
Sanqiang Zhao, Kaiqiang Song, Silei Xu, Chenguang Zhu
Zoom Video Communications

Abstract

Reinforcement learning from human feedback (RLHF) is a promising solution to align large language models (LLMs) more closely with human values. Off-policy preference optimization, where the preference data is obtained from other models, is widely adopted due to its cost efficiency and scalability. However, off-policy preference optimization often suffers from a distributional gap between the policy used for data collection and the target policy, leading to suboptimal optimization. In this paper, we propose a novel strategy to mitigate this problem by simulating on-policy learning with off-policy preference data. Our Weighted Preference Optimization (WPO) method adapts off-policy data to resemble on-policy data more closely by reweighting preference pairs according to their probability under the current policy. This method not only addresses the distributional gap problem but also enhances the optimization process without incurring additional costs. We validate our method on instruction following benchmarks including Alpaca Eval 2 and MT-bench. WPO not only outperforms Direct Preference Optimization (DPO) by up to 5.6% on Alpaca Eval 2 but also establishes a remarkable length-controlled winning rate against GPT-4-turbo of 76.7% based on Gemma-2-9b-it. We release the code and models at https://github.com/wzhouad/WPO.

1 Introduction

Large language models (LLMs; Ouyang et al. 2022; Achiam et al. 2023; Tunstall et al. 2023; Chung et al. 2024) have demonstrated remarkable capabilities in generating human-like responses. However, they still face challenges in scenarios demanding high standards of reliability, safety, and ethics. To address these challenges, reinforcement learning from human feedback (RLHF; Christiano et al. 2017; Ouyang et al. 2022; Glaese et al. 2022) is a promising approach to better align LLMs with human values.

Depending on how the outputs are generated, RLHF can be categorized into on-policy and off-policy settings. In the on-policy setting (Schulman et al., 2017; Yuan et al., 2024; Rosset et al., 2024; Wu et al., 2024), the policy model used to generate outputs is the same as the policy model being optimized. During this process, a policy model is first initialized from supervised finetuning (SFT). Then, a reward model (Schulman et al., 2017; Gao et al., 2023; Jiang et al., 2023) is obtained based on human (Schulman et al., 2017) or AI (Lee et al., 2023) feedback. Finally, the policy model samples outputs during training, which are then evaluated using the reward model. The policy model is optimized to improve the expected reward using training objectives such as Proximal Policy Optimization (PPO; Schulman et al. 2017) and Direct Preference Optimization (DPO; Rafailov et al. 2023). However, on-policy RL relies heavily on policy sampling during training and online rewards, which can incur high costs. In contrast, in the off-policy setting (Tunstall et al., 2023; Ivison et al., 2023), the outputs are generated from different models, and the policy model is optimized based on these data instead of its sampled outputs. Consequently, off-policy RL offers significant advantages in terms of cost and data efficiency and is easier to scale up.

Nevertheless, off-policy RL often shows worse performance than on-policy RL, due to the distributional gap between the policy used to collect data and the target policy being optimized, which leads to instability and inefficiency in training (Fujimoto et al., 2019; Kumar et al., 2019, 2020; Xu et al., 2024; Tang et al., 2024a; Tajwar et al., 2024). In off-policy preference optimization, the optimization is typically performed on preference data sampled from other models, and all the preference singles are equally treated. However, some preference data, distant from the current policy, are less informative

[†]Correspondence to <wenxuan.zhou@zoom.us>

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 8328–8340
November 12-16, 2024

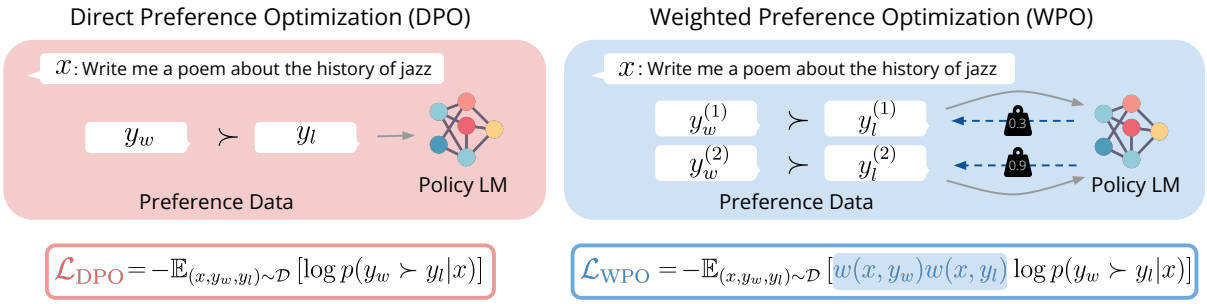

Figure 1: Overview of the Weighted Preference Optimization (WPO). Some notations are labeled along with corresponding components. Existing DPO directly optimizes the policy to best satisfy the preferences with off-policy data. In contrast, WPO adapts off-policy data to resemble on-policy data more closely by reweighting preference pairs according to their probability under the current policy.

for training, resulting in inefficient and suboptimal optimization.

In this paper, we propose simulating on-policy preference optimization with off-policy preference data, combining the efficiency of off-policy RL with the performance benefits associated with on-policy RL. Our method is motivated by the following conceptual data generation process. This process begins with transforming the existing preference dataset into a preference labeling function. We can then resample a new preference dataset through bootstrapping from the existing data. This process involves uniformly sampling inputs from the preference dataset and online sampling new pairs of outputs with the current policy model. Each pair is retained if it can be labeled by the labeling function; otherwise, it is rejected. We then perform DPO on the regenerated preference dataset. In practice, this bootstrapping process can be implemented with the Weighted Policy Optimization (WPO) objective, where different preference pairs are reweighted according to the joint probability of their outputs. We further devise a weighting alignment mechanism to ensure that all on-policy generated pairs are equally weighted. In this way, WPO can effectively mitigate the distribution gap during RL without incurring additional costs.

We evaluate WPO on instruction following benchmarks, including Alpaca Eval 2 (Li et al., 2023) and MT-bench (Zheng et al., 2023). In the off-policy setting based on Ultrafeedback (Cui et al., 2023), WPO improves the length-controlled winning rate against GPT-4-turbo on Alpaca Eval 2 by up to 14.9% over SFT model, outperforming DPO by up to 5.6%. Particularly, in the hybrid RL setting where the off-policy preference data is further enriched with on-policy outputs, WPO (Figure 1) achieves a new SOTA length-controlled winning rate of 76.7% on Alpaca Eval 2. Additionally, we find that WPO can be integrated into other loss functions for preference optimization and shows consistent improvements. Furthermore, we systematically compare the model performance in different RL settings. Our analysis reveals that the hybrid setting, which utilizes both on-policy and off-policy preference data, achieves the best results, and on-policy, dispreferred data is more important for preference optimization.

To summarize, our contributions are three-fold:

- We identify the distribution gap problem in off-policy preference optimization, and accordingly introduce a method to simulate on-policy RL using off-policy preference data.
- We propose the WPO objective, which reweights preference pairs based on their probabilities. This ensures that the most relevant and probable outputs are prioritized during optimization, mitigating the distribution gap and improving the effectiveness of the preference optimization.
- We conduct extensive instruction following benchmarks. Our results demonstrate that WPO significantly outperforms DPO and achieves new SOTA results on Alpaca Eval 2 in the hybrid RL setting.

2 Related Work

General alignment methods. The advancement of ChatGPT has propelled significant advancements in the field of large language models (LLMs). Notable models such as Zephyr (Tunstall et al., 2023) and GPT-4 (Achiam et al., 2023) have effectively demonstrated the application of techniques like reinforcement learning from human feedback (RLHF; Christiano et al. 2017; Ouyang et al. 2022;

Glaese et al. 2022) and direct preference optimization (DPO; Rafailov et al. 2023), highlighting their efficacy in achieving improved model alignment. These approaches, along with related methods such as sequence likelihood calibration (Zhao et al., 2023) and Generalized Preference Optimization (GPO) (Tang et al., 2024b), aim to refine the objectives of RLHF by clearly enhancing the distinction between more and less preferred outputs. Additionally, the introduction of the Direct Nash Optimization (DNO) algorithm by Rosset et al. (2024) represents a further innovation. This algorithm utilizes cross-entropy to assess the gap between actual and predicted win rates. Practical applications more frequently rely on the iterative framework of DPO (Xu et al., 2023). Yet, DPO often reveals a discrepancy between the output distributions produced by the policy and those in the preference dataset. To address this, we propose simulating on-policy reinforcement learning using off-policy data, thereby combining the benefits of on-policy RL with enhanced efficiency.

On-policy reinforcement learning. Self-Play Fine-Tuning (Chen et al., 2024) operates under an iterative framework akin to DPO, utilizing human-labeled responses as "winners" and outputs from previous iterations as "losers" within each pairing. Similarly, Adversarial Preference Optimization (Cheng et al., 2023) incorporates contrastive losses, which obviate the need for direct feedback from annotators. This method introduces a token-level loss function known as Cringe Loss (Adolphs et al., 2022), which differentiates the correct subsequent token from a deliberately incorrect token from the vocabulary. Pairwise Cringe Loss (Xu et al., 2023) utilizes this cringe loss mechanism within a continuously improving iterative training framework. Moreover, the recent introduction of SAMI (Fränken et al., 2024) targets optimizing a lower bound on the conditional mutual information between prompts and responses through a contrastive estimation technique. In our approach, we adjust the importance of each pair in the training process by assigning greater weight to those pairs more likely to be sampled from the policy model, thus simulating on-policy reinforcement learning.

3 Method

In this section, we provide the theoretical background of RLHF and DPO in Section 3.1. We then introduce the distributional gap problem and propose the WPO method (Algorithm 1) in Section 3.2. Finally, we explore how to better simulate on-policy RL through weight alignment in Section 3.3.

Algorithm 1: Weighted Preference Optimization (WPO)

Input: Dataset ($\mathcal{D}$) with prompts and responses, policy LM π_θ, total number of iterations T, learning rate α_t,
for $t = 0$ to T **do**
  Sample a mini-batch of tuples (x, y_w, y_l) from $\mathcal{D}$,
  Calculate the alignment weight via Eq. (2),
  Compute $\mathcal{L}_{\text{WPO}}$ via Eq. (1),
  Update policy parameters θ using gradient descent: $\theta \leftarrow \theta - \alpha_t \nabla \theta(x, y_w, y_l, \theta)$.
end for

3.1 Preliminaries

RLHF (Schulman et al., 2017) aims to align a large language model with human preferences. Given a preference dataset $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$, in which y_w and y_l are a pair of outputs given prompt x sampled from a policy model, and y_w is favored over y_l as determined by human or AI annotators. This preference is modeled by a latent reward function $r^*(x, y)$, which scores on how well the candidate output y matches the input x. There are various ways to model the reward function, among which the Bradley-Terry (BT; Bradley and Terry 1952) model is most commonly used. The BT model assumes that the preference distribution is characterized by the following equation:

$$p(y_w \succ y_l|x) = \frac{\exp(r^*(x,y_w))}{\exp(r^*(x,y_w))+\exp(r^*(x,y_l))}.$$

The parameters of the reward function can be estimated based on maximum likelihood estimation, resulting in the reward model $\hat{r}(x, y)$. Then, we can use the fitted reward model to provide feedback to a large language model by optimizing the following objective:

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \left[\hat{r}(x, y) - \beta \log \frac{\pi_\theta(\cdot|x)}{\pi_{\text{ref}}(\cdot|x)}\right],$$

where β controls the deviation between the policy model π_θ and the reference model π_{ref}, which is usually initialized from the SFT model.

DPO. Direct optimization optimization (DPO; Rafailov et al. 2023) integrates the learning of the

reward function and the policy model to a unified objective. Specifically, suppose the optimal policy π^* is given, the corresponding reward r^* has a closed form:

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

where $Z(x)$ is the partition function. Applying this reparameterization to the BT model, we have:

$$p^*(y_w \succ y_l|x) = \sigma\left(\beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right).$$

We can then formulate a maximum likelihood estimation objective for the policy model π_θ on the preference dataset $\mathcal{D}$, resulting in the following training objective:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log p(y_w \succ y_l|x)\right].$$

Here, the loss is calculated based on a uniform sampling of the preference dataset. In practice, the y_w and y_l in the preference dataset may be generated either with the same policy model being optimized, which corresponds to an on-policy RL setting (Xu et al., 2023; Yuan et al., 2024; Rosset et al., 2024), or with other models (e.g., GPT-4; Achiam et al. 2023), corresponding to an off-policy RL setting (Tunstall et al., 2023; Ivison et al., 2023; Pal et al., 2024).

3.2 Weighted Preference Optimization

DPO does not require actively generating new outputs from the current policy, making it more cost-effective and suitable for off-policy settings. However, DPO introduces a notable discrepancy between the distribution of outputs produced by the policy and those present in the preference dataset. This divergence can lead to less effective learning. To illustrate, consider two instances of preference data: $\left(x^{(1)}, y_w^{(1)}, y_l^{(1)}\right)$ and $\left(x^{(2)}, y_w^{(2)}, y_l^{(2)}\right)$, where the first tuple is sampled directly from the current policy model, while the second tuple is sampled from a different distribution from the current policy model. Despite this difference in sampling probability, DPO treats both instances equally in its loss calculation, ignoring the fact that the first tuple, representing a more probable output of the current policy, should ideally exert a greater influence on the optimization process. This oversight can lead to suboptimal performance, as DPO does not prioritize learning from the most representative or probable output of the policy model.

To address this issue, we propose to simulate on-policy RL using off-policy data, thereby being both fast and enjoying benefits from on-policy RL.

Theoretical derivation. To simulate on-policy RL, we first transform the (off-policy) preference dataset $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$ into the following preference labeling function:

$$f(x, y_1, y_2) = \begin{cases} y_1 \succ y_2, & (x, y_1, y_2) \in \mathcal{D} \\ y_2 \succ y_1, & (x, y_2, y_1) \in \mathcal{D} \\ \text{NA}, & \text{otherwise} \end{cases}$$

where we assume that the dataset contains no conflicting preferences, meaning that for any x, if $(x, y_1, y_2) \in \mathcal{D}$, then $(x, y_2, y_1) \notin \mathcal{D}$. We then conceptually generate a new preference dataset through a bootstrapping approach without actually carrying out the procedure. Suppose an input x is uniformly sampled from the original preference dataset, and then a pair of outputs y_1, y_2 is sampled with the current policy model. We retain the pair if it can be labeled by the labeling function, and otherwise reject the pair when $f(x, y_1, y_2) = \text{NA}$. If we sample for an infinite amount of times, according to the law of large numbers, the occurrence rate of a pair (x, y_w, y_l) would be proportional to $\pi_\theta(y_w|x)\pi_\theta(y_l|x)p(x)$. We then apply DPO to the newly generated preference dataset.

Practical implementation. The conceptual process above is equivalent to optimizing the following weighted preference optimization (WPO) objective, where different pairs in the original preference dataset are reweighed:

$$\mathcal{L}_{\text{WPO}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[w(x, y_w)w(x, y_l)\log p(y_w \succ y_l|x)\right], \quad (1)$$

where $w(x, y) = \pi_\theta(y|x)$ and is *detached* from back propagation. Through this process, we effectively adjust the importance of each pair in the training process, giving greater weight to those pairs that are more likely to be sampled from the policy model, thus simulating on-policy RL.

In language models where y_w and y_l are sequences of tokens, the product of the conditional probabilities $\pi_\theta(y_w|x) \cdot \pi_\theta(y_l|x)$ can be exceedingly small and exhibit high variance among different pairs. To address this, we utilize the length-normalized sequence probability as a weighting factor:

$$w(x, y) = \exp\left(\frac{1}{|y|}\sum_{t=1}^{|y|}\log \pi_\theta(y_t|x, y_{<t})\right),$$

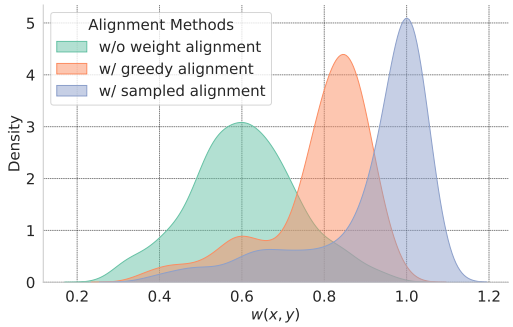

Figure 2: Weight distribution of outputs sampled using the policy model with different alignment methods.

where $|y|$ represents the number of tokens in the output.

3.3 Weight Alignment

The objective of our weighting strategy is to simulate on-policy RL, where outputs are weighted according to how closely they align with on-policy behavior. For outputs generated by the current policy model, we expect their weights to be uniformly 1, while outputs that deviate from this on-policy behavior should receive smaller weights. However, due to the varying levels of confidence that LLMs exhibit across different inputs (Si et al., 2023; Xiong et al., 2024), even outputs generated by the policy model may sometimes be assigned low weights. This introduces an unintended bias where some on-policy outputs receive lower weights purely because of lower model confidence based on the input, disrupting the uniformity we aim to achieve. Figure 2 shows the weight distribution of sampled outputs based on prompts from Ultrafeedback and the Mistral-sft-beta model, in which we observe significant variability in $w(x, y)$. To address this and ensure equal weighting of these outputs, we propose to align the weights in WPO.

A direct method is to adjust the weights above by the sequence probability of the on-policy outputs sampled from the policy model. However, generating outputs during training is computationally expensive, and hence, we explore approximation methods for this alignment. Instead of using weights of the whole sequences as reference, we operate at the token level and adjust the probability of output tokens according to the token distribution in the policy model, based on the current subsequence. We propose two ways to achieve the alignment.

Greedy alignment. In this approach, we adjust the weights based on greedy decoding by comparing the probability of the current token with that of the most probable token in the vocabulary. Specifically, we adjust weights based on the maximum token probability among the set of all tokens in the subsequence, defined as:

$$w(x, y) = \exp\left(\frac{1}{|y|}\sum_{t=1}^{|y|}\log\frac{\pi_\theta(y_t|x, y_{<t})}{\max_{v\in\mathcal{V}}\pi_\theta(v|x, y_{<t})}\right),$$

where $\mathcal{V}$ represents the set of all tokens in the language model.

Sampled alignment. In this approach, we adjust weights based on outputs that are randomly sampled from the policy model at a temperature of 1.0. Since the probability for each token v is computed as $\pi_\theta(v|x, y_{<t})$, the expected probability of a randomly sampled token would be $\sum_{v\in\mathcal{V}}\pi_\theta(v|x, y_{<t})^2$, and the calibrated weights are then given by:

$$w(x, y) = \exp\left(\frac{1}{|y|}\sum_{t=1}^{|y|}\log\frac{\pi_\theta(y_t|x, y_{<t})}{\sum_{v\in\mathcal{V}}\pi_\theta(v|x, y_{<t})^2}\right). \quad (2)$$

We use sampled alignment as the default alignment method in WPO due to its superior performance, as confirmed in Section 4.2. Additionally, in Figure 2, sampled alignment leads to a more concentrated weight distribution of outputs from the policy model, thereby better simulating on-policy RL.

4 Experiment

In this section, we outline our experimental settings (Section 4.1) and present the main results along with ablation studies (Section 4.2). We then compare different RL settings (Section 4.3). Additional analysis of WPO is provided in Appendix A.

4.1 Experimental Settings

Model configurations. Our methods are implemented based on the official code of zephyr[1]. For Mistral-base, we adopt the official hyperparameters from zephyr. Specifically, we use the SFT checkpoint of zephyr[2] as our SFT model. Training is conducted over a single epoch with a batch size of 128, a learning rate of 5e-7, a warm-up phase for 10% of the training, and a cosine decay schedule. We set β to 0.01 for both DPO and WPO. For Llama-3-Instruct, we perform a hyperparameter search within the range recommended by Meng

[1] https://github.com/huggingface/alignment-handbook

[2] https://huggingface.co/HuggingFaceH4/mistral-7b-sft-beta

	Method	Mistral-Base (7B)				Llama-3-Instruct (8B)			
		Alpaca Eval 2.0		MT-bench		Alpaca Eval 2.0		MT-bench	
		Len-control. Win Rate	Win Rate vs GPT-4	Avg. Score	Win Rate vs DPO	Len-control. Win Rate	Win Rate vs GPT-4	Avg. Score	Win Rate vs DPO
	SFT	9.5	5.8	6.64	-	26.0	25.3	7.97	-
Off-policy	ORPO	14.7	12.6	7.32	-	-	-	-	-
Off-policy	KTO	14.9	12.3	7.36	-	-	-	-	-
Off-policy	SimPO	21.5	21.4	7.32	-	-	-	-	-
Off-policy	DPO	20.6 (0.7)	18.6 (1.0)	7.36 (0.04)	50 (0)	28.2 (0.5)	24.0 (0.5)	8.10 (0.05)	50 (0)
Off-policy	WPO	**24.4 (1.4)**	**23.7 (2.1)**	**7.37 (0.10)**	**60.1 (4.7)**	**33.8 (1.3)**	**31.0 (1.8)**	**8.14 (0.05)**	**58.1 (3.4)**
Hybrid	DPO	37.9 (1.2)	40.3 (1.1)	7.14 (0.41)	50 (0)	44.2 (1.2)	48.6 (1.0)	8.16 (0.10)	50 (0)
Hybrid	WPO	42.0 (1.7)	46.2 (2.3)	**7.38 (0.08)**	56.4 (4.6)	45.8 (1.3)	50.0 (1.1)	**8.18 (0.22)**	54.8 (2.2)
Hybrid	+ *Ultrafeedback*	**43.1 (1.1)**	**49.6 (1.2)**	7.23 (0.19)	**58.8 (4.5)**	**48.6 (1.3)**	**52.1 (1.2)**	8.14 (0.10)	**55.1 (2.4)**

Table 1: Alpaca Eval 2.0 and MT-bench results. We report the average and standard deviation of the results from 5 runs of different random seeds. Scores that are underlined denote statistically significant gains ($p < 0.05$).

et al. (2024). Our final hyperparameters are a learning rate of 1e-6, two training epochs, and β of 0.01 for both DPO and WPO. For all training configurations, we conduct training for 5 runs with different random seeds and report both the average results and their standard deviation.

Training data. We perform RLHF in off-policy and hybrid settings. In the *off-policy* setting, we use the binarized Ultrafeedback dataset[3](Cui et al., 2023), which compromises 63k preference pairs sampled from models other than our SFT model, such as GPT-4 and Llama-2 (Touvron et al., 2023). In the *hybrid* setting, we follow the approach in DNO (Rosset et al., 2024), using data generated from both the policy model and other models. Specifically, we sample 5 outputs from the SFT model based on prompts from Ultrafeedback and add another output generated by `gpt-4-turbo`. We employ top-p sampling with $p = 0.95$ and a temperature of 0.7. Preference annotations are produced using `gpt-4-turbo` with additive scoring prompt. For each prompt, we select outputs scoring 5 or 6 as y_w and then choose a random output with a score at least one point lower as y_l. If such a pair cannot be found, the prompt is not used. This data construction step produces a smaller preference dataset, so we further employ the + *Ultrafeedback* setting, where we add the missing prompts back using the preference pairs from Ultrafeedback.

Evaluation. We evaluate the models on Alpaca Eval 2 and MT-bench. Alpaca Eval 2 is an automated metric that measures LLMs' alignment with human preferences using 805 representative instructions. For each instruction, the evaluated model's response and `gpt-4-turbo`'s response are compared head-to-head using an auto-evaluator. The win rate is the probability that the auto-evaluator prefers the evaluated model's responses. Alpaca Eval 2 also introduces a length-controlled win rate (Dubois et al., 2024) to address the length bias of `gpt-4-turbo`. We follow the generation configurations in Tunstall et al. (2023) for Mistral models and in Zheng et al. (2024) for Llama-3 models.

MT-bench is an LLM-based automated evaluation metric comprising 80 challenging questions. We report results using two scoring methods. In the single answer grading approach, the auto-evaluator (`gpt-4-0613`) assigns scores from 1 to 10 to responses, and we report the average scores. In the pairwise comparison approach, the evaluator (`gpt-4-0613`) compares two responses to decide which is better or if it's a tie (recorded as 0.5 in win rate). The pairwise method can detect more subtle differences between responses than single answer grading. We use the official generation configurations in MT-bench.

4.2 Main Results and Ablation

WPO consistently and significantly outperforms DPO and its variants. The main results are shown in Table 1. We include the results of different preference optimization algorithms such as DPO, ORPO (Hong et al., 2024), KTO (Ethayarajh et al., 2024), and SimPO (Meng et al., 2024) on the two benchmarks. For ORPO, KTO, and SimPO, we report the evaluation results of their official model checkpoints on Mistral-base.[4] We find that WPO

[3] https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized

[4] We do not include their results on Llama-3-Instruct in the off-policy setting as the official checkpoints are unavailable. Reproducing these methods requires extensive hyperparameter

Method	Alpaca Eval 2.0		MT-bench
	Len-control. Win Rate	Win Rate vs GPT-4	Win Rate vs DPO
WPO *w/ sampled align.*	24.4	23.7	60.1
WPO *w/ greedy align.*	23.0	21.4	57.9
WPO *w/o align.*	22.0	20.3	54.4
DPO	20.6	18.6	50

Table 2: Ablation of weight alignment methods on Mistral-base in the off-policy setting. sampled alignment, the default weight alignment method, yields the best results.

generally outperforms DPO in all settings and also outperforms all its variants on Mistral-base in the off-policy setting. Particularly, when trained with the Llama-3-Instruct model and the hybrid *+Ultrafeedback* setting, WPO achieves a new state-of-the-art length-controlled win rate of 48.6% against GPT-4-turbo on Alpaca Eval 2. These results highlight the effectiveness of WPO. Additionally, while DPO underperforms compared to SimPO, it still demonstrates competitive results, providing a solid basis for WPO.

Varied separation of benchmarks. On MT-bench, the average score does not effectively distinguish the performance of different models. Additionally, we observe variability in the average MT-bench score. Even when using GPT-4 to score the same outputs with a temperature of 0, the score can vary by up to 0.1 at different times. Given the clearer separation in our experiments and the greater alignment with human evaluations, as shown in the original paper (Zheng et al., 2023), we consider pairwise win rate to be a more suitable metric for assessing different alignment methods. Therefore, we use it for MT-bench in the following part of the paper.

Sampled weight alignment works the best. Table 2 shows the results of WPO with different weight alignment methods on Mistral-base in the off-policy setting. We observe that sampled alignment outperforms other variations on both benchmarks, while greedy sampling outperforms w/o alignment. We also find that the ranking of performance matches the ranking of concentration levels in the weight distribution shown in Figure 2. This indicates that weight alignment enables a more effective simulation of on-policy RL, leading to improved performance.

WPO also improves other loss functions for preference optimization. It is important to note that, in addition to DPO, there are other loss functions for aligning LLMs. Since WPO works by weighing preference data and is independent to the loss function being used, it can be easily integrated into them. We investigate whether the integration of WPO enhances the performance of other loss functions. Existing losses can be categorized into those using paired preference data and those utilizing unpaired preference data. For losses using paired data, we weigh each pair similarly to DPO. For losses using unpaired data, we weigh each output y independently with $w(x, y)$ and normalize the weights so that the total weights of favored outputs and disfavored outputs are both 1 within the batch. This normalization ensures a balance between favored and disfavored outputs in the loss. In this study, we considered IPO (Azar et al., 2024) and SimPO for alignment with paired data, and KTO for alignment with unpaired data. The results on Mistral-base in the off-policy setting, shown in Table 3, indicate that integrating WPO leads to improved results for all loss functions. This demonstrates that WPO provides universal improvements across different loss functions for preference optimization.

Method	Alpaca Eval 2.0		MT-bench
	Len-control. Win Rate	Win Rate vs GPT-4	Win Rate vs Baseline
IPO	25.0	21.2	50
SimPO	21.5	21.4	50
KTO	14.9	12.3	50
WPO_{IPO}	29.4	25.7	54.1
$\text{WPO}_{\text{SIMPO}}$	21.9	24.6	52.5
WPO_{KTO}	21.1	20.3	60.0

Table 3: Results of WPO with different loss functions for preference optimization on Mistral-base in the off-policy setting, which show that incorporating WPO leads to consistent improvements.

Better base and reward models yield stronger results. To further enhance our model, we investigate using better base models and reward models. Specifically, we adopt the Gemma-2-9b-it (Team et al., 2024) as the base model. In a setup similar to our hybrid approach, we sample five outputs from Gemma and one additional output from `gpt-4-turbo`. To rank these outputs, we apply ArmoRM (Wang et al., 2024a,b) and use the best and worst outputs to form preference pairs. We use the same set of training hyperparameters used in Llama-3-Instruct. The model finetuned using WPO achieves a length-controlled win rate of 76.7% and a win rate of 77.8% on Alpaca Eval 2, demonstrat-

searches, which may not yield the optimal hyperparameter values for a fair comparison.

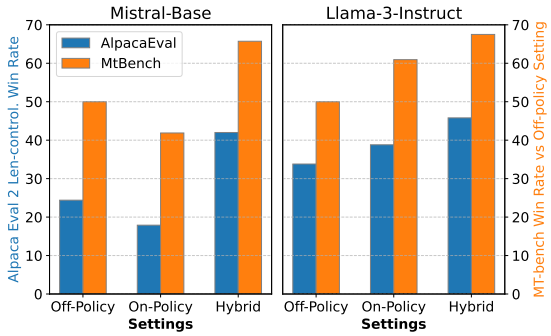

Figure 3: Results of WPO in different RL settings. The hybrid setting consistently yileds better results than other RL settings.

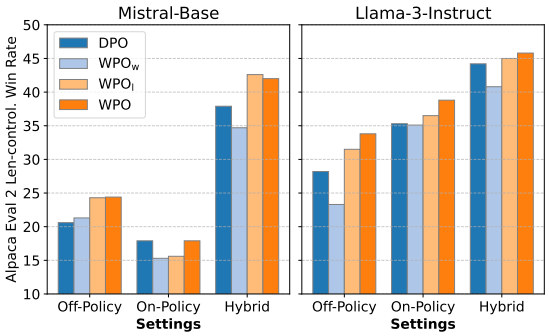

Figure 4: Results of variations of WPO in different RL settings.

ing the effectiveness of this approach.

4.3 Comparison of Different RL Settings

Recent studies on RLHF have employed various RL settings where preference data is generated in an off-policy, on-policy, or hybrid manner. Existing work (Tang et al., 2024a; Xu et al., 2024) has demonstrated that on-policy preference optimization outperforms off-policy methods, while Rosset et al. (2024) show that incorporating high-quality off-policy outputs can yield superior performance, as these outputs can introduce valuable information that the current policy might not encounter on its own. In this study, we compare model performance trained with WPO across these RL settings. The results are presented in Figure 3, showcasing the length-controlled win rate on Alpaca Eval 2 and the pairwise win rate compared to the off-policy setting on MT-bench.

Hybrid RL achieves the best results. Figure 3 shows that for both Mistral-base and Llama-3-Instruct, the hybrid setting—utilizing both on-policy data and high-quality off-policy data from `gpt-4-turbo`—consistently delivers superior performance. This suggests that combining high-quality off-policy data and on-policy data can significantly enhance preference optimization, which is consistent to the results in Rosset et al. (2024).

On-policy is not always better than off-policy. Our analysis reveals that the effectiveness of on-policy versus off-policy preference optimization is model-dependent (Munos et al., 2016; Voloshin et al., 2019). For the Mistral-base model, off-policy setting yields slightly better performance, while for Llama-3-Instruct, on-policy setting shows better performance. We attribute this variation to the quality of the SFT model. In the case of Mistral-base, the sampled outputs are of lower quality, causing the preference optimization process to mimic suboptimal outputs and leading to poorer results. This highlights the importance of the initial policy's quality and suggests that models with higher initial performance might benefit more from on-policy optimization, while those with lower initial quality may not gain as much.

The dispreferred data should be on-policy, the preferred data benefits less. While WPO simulates on-policy data by weighing both y_w and y_l in the preference data, these two outputs play different roles during optimization. The gradient of the WPO is given by:

$$\nabla \mathcal{L}_{\text{WPO}} = -\beta w(x, y_w) w(x, y_l) \sigma\left(\hat{r}\left(x, y_l\right) - \hat{r}\left(x, y_w\right)\right)$$
$$\Big[\underbrace{\nabla \log \pi\left(y_w | x\right)}_{\text{increase the probability of } y_w} - \underbrace{\nabla \log \pi\left(y_l | x\right)}_{\text{reduce the probability of } y_l} \Big].$$

That is, WPO will make the policy model mimic y_w while moving away from y_l. Given their different optimization directions, we investigate the importance of on-policy sampling for y_w and y_l in preference optimization. To achieve this, we further study two different variants of WPO, namely WPO$_{\text{W}}$ and WPO$_{\text{L}}$. These losses are formulated as follows:

$$\mathcal{L}_{\text{WPO}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[w(x,y_w)w(x,y_l)\log p(y_w \succ y_l|x)\right],$$
$$\mathcal{L}_{\text{WPO}_{\text{W}}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[w(x,y_w)\log p(y_w \succ y_l|x)\right],$$
$$\mathcal{L}_{\text{WPO}_{\text{L}}} = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[w(x,y_l)\log p(y_w \succ y_l|x)\right],$$

where in WPO$_{\text{W}}$, we only increase the weights of pairs where y_w is more closed to on-policy outputs. For WPO$_{\text{L}}$, we only increase the weights of pairs where y_l is closer to on-policy outputs. Results on Mistral-base and Llama-3-Instruct are in Figure 4. It shows that WPO$_{\text{L}}$ generally achieves similar results to WPO. Conversely, WPO$_{\text{W}}$ consistently underperforms WPO and even underperforms DPO in most settings. Therefore, making y_l on-policy explains most of the improvements of

WPO, while making y_w on-policy is still useful but not as important. This finding suggests that using on-policy, dispreferred data is important for preference optimization, while using on-policy preferred data may be beneficial but not as critical.

5 Conclusion

In this study, we tackled the distributional gap problem inherent in off-policy preference optimization. By introducing Weighted Preference Optimization (WPO), we successfully simulated on-policy preference optimization using off-policy preference data, merging the benefits of both approaches. Our method not only addressed the distributional gap without incurring additional costs but also enhanced the effectiveness of preference optimization. Extensive experiments demonstrate that WPO can produce better LLMs that are more closely aligned with human preferences.

Limitations

The performance gap between off and on-policy preference optimization remains. Although WPO simulates on-policy RL with off-policy data, it does not fully bridge the performance gap between off-policy and on-policy RL. As shown in the results, even with WPO, off-policy methods may still underperform compared to on-policy and hybrid methods. Therefore, while we propose WPO as a solution, it does not entirely eliminate the performance disparity, and on-policy preference data remains important. Future work will be on how to further reduce this performance gap without incurring additional training costs.

Comprehensiveness of preference dataset. The goal of our experiments is to compare WPO with other preference optimization algorithms, not to provide a comprehensively aligned LLM. In our experiments, we use Ultrafeedback as the preference data, which primarily focuses on helpfulness, truthfulness, and instruction following, and does not include safety aspects. Additionally, it does not consider preference optimization for multi-turn conversations. Future work should involve collecting more comprehensive preference datasets and integrating multiple aspects of preference optimization to train better-aligned LLMs.

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*.

Leonard Adolphs, Tianyu Gao, Jing Xu, Kurt Shuster, Sainbayar Sukhbaatar, and Jason Weston. 2022. The cringe loss: Learning what language not to model. *arXiv preprint arXiv:2211.05826*.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. 2021. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*.

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.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.

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.

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

Pengyu Cheng, Yifan Yang, Jian Li, Yong Dai, and Nan Du. 2023. Adversarial preference optimization. *arXiv preprint arXiv:2311.08045*.

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

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.

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. *arXiv preprint arXiv:2310.01377*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

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

Jan-Philipp Fränken, Eric Zelikman, Rafael Rafailov, Kanishk Gandhi, Tobias Gerstenberg, and Noah D Goodman. 2024. Self-supervised alignment with mutual information: Learning to follow principles without preference labels. *arXiv preprint arXiv:2404.14313*.

Scott Fujimoto, David Meger, and Doina Precup. 2019. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Amelia Glaese, Nat McAleese, Maja Trębacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, et al. 2022. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*, 2(4):5.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. 2019. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*.

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

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

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. 2016. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29.

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*.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, Bradley Knox, Chelsea Finn, and Scott Niekum. 2024. Scaling laws for reward model overoptimization in direct alignment algorithms. *arXiv preprint arXiv:2406.02900*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. 2024. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*.

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

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. 2023. Prompting gpt-3 to be reliable. In *The Eleventh International Conference on Learning Representations*.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. 2024. Preference fine-tuning of llms should leverage suboptimal, on-policy data. *arXiv preprint arXiv:2404.14367*.

Yunhao Tang, Daniel Zhaohan Guo, Zeyu Zheng, Daniele Calandriello, Yuan Cao, Eugene Tarassov, Rémi Munos, Bernardo Ávila Pires, Michal Valko, Yong Cheng, et al. 2024a. Understanding the performance gap between online and offline alignment algorithms. *arXiv preprint arXiv:2405.08448*.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. 2024b. Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

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*.

Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. 2019. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. 2024a. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. In *ACL*.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024b. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*.

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*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. In *The Twelfth International Conference on Learning Representations*.

Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. 2023. Some things are more cringe than others: Preference optimization with the pairwise cringe loss. *arXiv preprint arXiv:2312.16682*.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. 2024. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*.

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

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*.

Chujie Zheng, Ziqi Wang, Heng Ji, Minlie Huang, and Nanyun Peng. 2024. Weak-to-strong extrapolation expedites alignment. *arXiv preprint arXiv:2404.16792*.

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

Method	ARC	TruthfulQA	WinoGrande	GSM8k	HellaSwag	MMLU	Average
Mistral-Base (7B)							
SFT	58.19	43.03	77.51	38.89	82.30	59.78	59.95
Off-policy DPO	64.42	52.44	79.48	30.17	85.36	59.78	61.94
Off-policy WPO	64.08	51.07	78.14	32.60	85.17	59.51	61.76
Hybrid DPO	64.76	60.46	78.22	32.15	85.30	58.75	63.27
Hybrid WPO	65.70	57.62	79.08	30.71	85.15	59.82	63.01
Llama-3-Instruct (8B)							
SFT	61.60	51.65	76.72	75.82	78.68	65.65	68.35
Off-policy DPO	68.00	61.07	77.43	74.68	82.26	66.31	71.63
Off-policy WPO	66.98	58.91	75.45	71.95	81.87	65.97	70.19
Hybrid DPO	65.53	56.10	78.93	75.13	81.12	65.72	70.42
Hybrid WPO	65.27	55.47	79.72	66.72	81.02	65.97	69.03

Table 4: Results on the OpenLLM leaderboard.

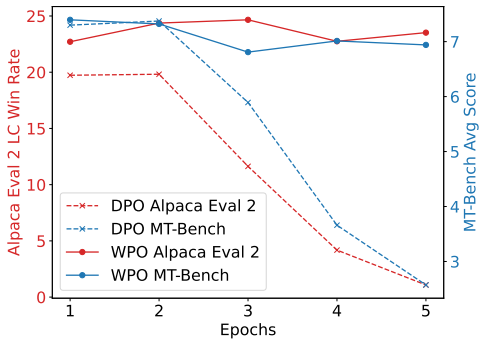

Figure 5: Results of DPO and WPO when trained for more epochs.

A Additional Analysis

Results on downstream tasks. We further evaluate the performance of SFT, DPO, and WPO models on the OpenLLM leaderboard (Beeching et al., 2023) to assess their capabilities on downstream tasks. For this evaluation, we use the lm-evaluation-harness[5], the official code base for the OpenLLM leaderboard. Results are shown in Table 4. Generally, we find that preference optimization with DPO or WPO outperforms the SFT model, while Llama-3-Instruct based models outperform Mistral-base. However, we do not observe a correlation between performance on the OpenLLM leaderboard and performance on instruction-following benchmarks such as Alpaca Eval 2 and MT-bench. For example, although Llama-3-Instruct with DPO or WPO in the hybrid setting shows the best results on instruction-following benchmarks, it underperforms its off-policy counterparts on the OpenLLM leaderboard. Additionally, we find that preference optimization may not improve results on all downstream tasks. On MMLU, the results are similar to SFT, and on GSM8K, the results are even lower than SFT in all settings. Our findings are consistent with the alignment tax phenomenon (Askell et al., 2021), which indicates that better alignment may not improve and can sometimes even hurt performance on downstream tasks.

Comparison between DPO and WPO on training dynamics. We investigate how the performance of DPO and WPO changes with different numbers of training epochs. Both DPO and WPO were trained using the SFT checkpoint of Mistral-base and the Ultrafeedback dataset for five epochs, with evaluation results recorded at the end of each epoch, as shown in Figure 5. In this study, we use the same set of hyperparameters as mentioned in Section 4.1, with DPO and WPO using the same set of hyperparameters. We observed that DPO's performance declines sharply after two epochs, suggesting strong reward model overoptimization (Rafailov et al., 2024). In contrast, WPO maintains consistent performance over more

[5] https://github.com/EleutherAI/lm-evaluation-harness

epochs, indicating better training stability. This suggests that simulating on-policy RL, as done by WPO, may mitigate issues related to reward model overoptimization and increase the stability of preference optimization. Furthermore, a comparison of results between DPO and WPO, particularly on Alpaca Eval 2, shows that the peak performance of DPO across various epochs still falls below that of WPO. This indicates that WPO not only provides more stable training dynamics but also finds a different and better solution than DPO. This enhanced performance and stability highlight the advantages of WPO in effectively leveraging the preference data and maintaining stable and robust preference optimization throughout the training process.

B Link of Open Sourced Models in Experiments

The list of open-sourced LLMs and their Huggingface IDs are listed in Table 5.

Model	Huggingface ID
Mistral-base SFT	`HuggingFaceH4/mistral-7b-sft-beta`
Mistral-base ORPO	`kaist-ai/mistral-orpo-beta`
Mistral-base KTO	`ContextualAI/zephyr_sft_kto`
Mistral-base SimPO	`princeton-nlp/Mistral-7B-Base-SFT-SimPO`
Llama-3-instruct SFT	`meta-llama/Meta-Llama-3-8B-Instruct`

Table 5: List of open-source models in experiments.

C Additional Details

Scientific artifacts. We use various scientific artifacts throughout the paper, including base LLM models, preference datasets, and evaluation tools/benchmarks. References to all used artifacts are provided, and details such as their license, language, coverage, number of parameters, and any safety issues can be found by following the respective references. Note that current LLMs and preference datasets may encompass a wide range of data types and utilizes data from different domains and sources, so we do not list the details in this paper and encourage readers to refer to the original sources for more information. In this paper, we primarily use these artifacts for non-distributive and non-commercial purposes, which is in compliance with their licenses.

Budget. We conduct all experiments using 8 $\times$ H100 GPUs. The experiments take approximately 1.5 hours for Mistral-base and around 4 hours for Llama-3-Instruct.

Use of AI assistants. We used ChatGPT solely for revising the language of the paper. Note that the revision is exclusively for enhancing the clarity and readability of the text, and not for any other purposes.