ORPO: Monolithic Preference Optimization without Reference Model

Jiwoo Hong Noah Lee James Thorne

KAIST AI

{jiwoo_hong, noah.lee, thorne}@kaist.ac.kr

Abstract

While recent preference alignment algorithms for language models have demonstrated promising results, supervised fine-tuning (SFT) remains imperative for achieving successful convergence. In this paper, we revisit SFT in the context of preference alignment, emphasizing that a minor penalty for the disfavored style is sufficient for preference alignment. Building on this foundation, we introduce a straightforward reference model-free monolithic odds ratio preference optimization algorithm, ORPO, eliminating the need for an additional preference alignment phase. We demonstrate, both empirically and theoretically, that the odds ratio is a sensible choice for contrasting favored and disfavored styles during SFT across diverse sizes from 125M to 7B. Specifically, fine-tuning Phi-2 (2.7B), Llama-2 (7B), and Mistral (7B) with ORPO on the UltraFeedback alone surpasses the performance of state-of-the-art language models including Llama-2 Chat and Zephyr with more than 7B and 13B parameters: achieving up to 12.20% on AlpacaEval$_{2.0}$ (Figure 1), and 7.32 in MT-Bench (Table 2). We release code[1] and model checkpoints[2] for Mistral-ORPO-α and Mistral-ORPO-β.

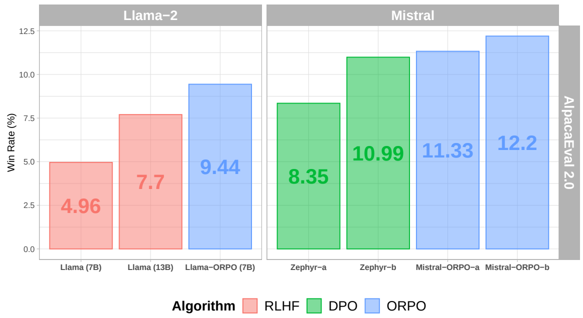


Figure 1: AlpacaEval$_{2.0}$ result of Llama-2 (7B) and Mistral (7B) fine-tuned with ORPO (blue) in comparison to the state-of-the-art models. Notably, Mistral-ORPO-α & β surpasses Zephyr β and Llama-2-Chat (13B) with a single epoch training exclusively on the UltraFeedback.

1 Introduction

Pre-trained language models (PLMs) with vast training corpora such as web texts (Gokaslan and Cohen, 2019; Penedo et al., 2023) or textbooks (Li et al., 2023c) have shown remarkable abilities in diverse natural language processing (NLP) tasks (Brown et al., 2020; Zhang et al., 2022; Touvron et al., 2023; Jiang et al., 2023; Almazrouei et al., 2023). However, the models must undergo further tuning to be usable in downstream applications, typically through processes such as *instruction tuning* and *preference alignment*.

Instruction-tuning (Wei et al., 2022; Taori et al., 2023; Wang et al., 2023; Zhou et al., 2023a) trains models to follow task descriptions given in natural language, which enables models to generalize well to previously unseen tasks. However, despite the ability to follow instructions, models may generate harmful or unethical outputs (Carlini et al., 2021; Gehman et al., 2020; Pryzant et al., 2023). To further align these models with human values, additional training is required with pairwise preference data using techniques such as reinforcement learning with human feedback (Ziegler et al., 2020; Stiennon et al., 2022, RLHF) and direct preference optimization (Rafailov et al., 2023, DPO). Existing preference alignment methods typically consist of a multi-stage process, as shown in Figure 2, typically requiring a second reference model and a separate warm-up phase with supervised fine-tuning (SFT) (Ziegler et al., 2020; Rafailov et al., 2023; Wu et al., 2023), which adds additional resource overheads.

We propose a novel alignment method, Odds Ratio Preference Optimization (ORPO), which adds a penalization term that steers the model away from learning undesired generation styles during SFT. In evaluation, fine-tuning Phi-2 (2.7B), Llama-2 (7B), and Mistral (7B) with ORPO results in higher win rates in AlpacaEval$_{2.0}$, when compared

[1] GitHub: https://github.com/xfactlab/orpo
[2] Models: ORPO collection

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 11170–11189
November 12-16, 2024 ©2024 Association for Computational Linguistics

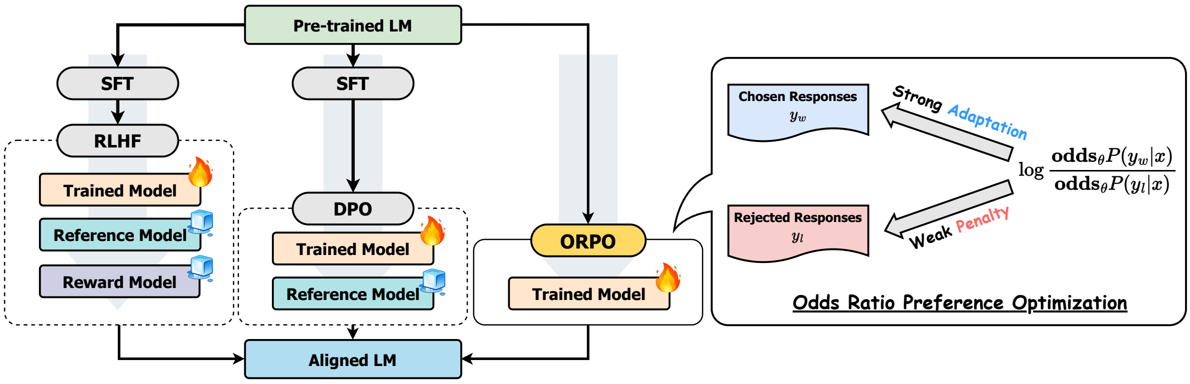


Figure 2: Comparison of model alignment techniques. ORPO aligns the language model *without a reference model* in a single-step manner by assigning a weak penalty to the rejected responses and a strong adaptation signal to the chosen responses with a simple log odds ratio term appended to the negative log-likelihood loss.

to DPO with Phi-2 (2.7B) ($+5.57\%$), and official checkpoints for Llama-2 (7B) Chat ($+4.48\%$) and Llama-2 (13B) Chat ($+1.74\%$), and Zephyr-α (7B) ($+2.98\%$), respectively. In our analyses, we demonstrate empirical superiority by comparing ORPO against established methods for model alignment, RLHF, and DPO for different datasets and model sizes. Further analysis from theoretical, empirical, and efficiency perspectives offers the following enhancements over previous methods: (1) requiring neither a reference model nor SFT stage and (2) less than half of the computational load in actual training. We release the training code and the checkpoints for Mistral-ORPO-α (7B) and Mistral-ORPO-β (7B), which achieve 7.24 and 7.32 in MT-Bench and 11.33% and 12.20% on AlpacaEval$_{2.0}$, as shown in Figure 1 and Table 2.

2 Related Works

Alignment with reinforcement learning Reinforcement learning with human feedback (RLHF) commonly applies the Bradley-Terry model (Bradley and Terry, 1952) to estimate the probability of a pairwise competition between two independently evaluated instances. An additional reward model is trained to score instances. Reinforcement learning algorithms such as proximal policy optimization (PPO) (Schulman et al., 2017) are employed to train the model to maximize the score of the reward model for the chosen response, resulting in language models that are trained with human preferences (Ziegler et al., 2020; Stiennon et al., 2022; Ouyang et al., 2022). However, RLHF faces challenges of extensive hyperparameter searching due to the instability of PPO (Rafailov et al., 2023; Wu et al., 2023) and the sensitivity of the reward models (Gao et al., 2022; Wang et al., 2024). Therefore, there is a demanding need for stabler preference alignment algorithms.

Alignment without reward model Recently proposed techniques for preference alignment mitigate the need for reinforcement learning (Rafailov et al., 2023; Song et al., 2023; Azar et al., 2023; Ethayarajh et al., 2024; Xu et al., 2024; Rosset et al., 2024). Rafailov et al. (2023) introduce direct preference optimization (DPO), which removes the reward modeling stage. Azar et al. (2023) prevented potential overfitting problems in DPO through identity preference optimization (IPO). Ethayarajh et al. (2024) and Cai et al. (2023) proposed Kahneman-Tversky Optimization (KTO) and Unified Language Model Alignment (ULMA) that does not require the pair-wise preference dataset, unlike RLHF and DPO. Song et al. (2023) and Xu et al. (2024) further suggest incorporation of the softmax value of the reference response set in the negative log-likelihood loss to merge the supervised fine-tuning and preference alignment.

Alignment with supervised fine-tuning There have been approaches to build human-aligned language models by conducting supervised fine-tuning (SFT) only with filtered datasets (Zhou et al., 2023a; Li et al., 2023a; Haggerty and Chandra, 2024; Zhou et al., 2023b; Dong et al., 2023; Yuan et al., 2023; Gulcehre et al., 2023). Zhou et al. (2023a) demonstrated that SFT with a small amount of data with fine-grained curation could be sufficient for building helpful language model assistants. Furthermore, Li et al. (2023a) and Haggerty and Chandra (2024) proposed an iterative process of fine-tuning the supervised fine-tuned language models with their own generations after fine-grained selection of aligned generations and Zhou et al. (2023b) suggested that a curated subset of preference dataset is sufficient for alignment.

3 Odds Ratio Preference Optimization

We introduce a novel preference alignment algorithm, Odds Ratio Preference Optimization (ORPO), which incorporates an odds ratio-based penalty to the conventional supervised fine-tuning (SFT) loss (i.e., negative log-likelihood (NLL)) for differentiating the generation styles between favored and disfavored responses. We discuss the effects of SFT on preference alignment in Section 3.2 and explain the mechanism of ORPO in Section 3.3.

3.1 Preliminaries

Given a language model π_θ and an input sequence x, the average log-likelihood of π_θ generating the output sequence y is computed as:

$$\log P_\theta(y|x) = \frac{1}{|y|}\sum_{t=1}^{m}\log \pi_\theta(y_t|x, y_{<t}).$$

And the *odds* of generating the output sequence y given an input sequence x is defined as:

$$\textbf{odds}_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)},$$

where we use exponentiated average log-likelihood $P_\theta(y|x)$ to represent the likelihood of response y given prompt x in the form of probability.

We adopt the odds to model the preference given the likelihood of binary outcomes, preferred and dispreferred responses (Rafailov et al., 2023). While this is due to model a binary aspect of preferences, setting in which the odds ratio is applied, the use of average log-likelihood in alignment methods is widely studied for length regularization (Yuan et al., 2023; Park et al., 2024; Grinsztajn et al., 2024; Meng et al., 2024).

Intuitively, $\textbf{odds}_\theta(y|x) = k$ implies that it is k times more likely for the model π_θ to generate the output sequence y than not generating it. Thus, the odds ratio of the chosen response y_w over the rejected response y_l, $\textbf{OR}_\theta(x, y_w, y_l)$,

$$\textbf{OR}_\theta(x, y_w, y_l) = \frac{\textbf{odds}_\theta(y_w|x)}{\textbf{odds}_\theta(y_l|x)}, \quad (1)$$

indicates how much more likely it is for the model θ to generate y_w than y_l given input x.

Figure 3: Log probabilities for chosen and rejected responses during OPT-350M model fine-tuning on HH-RLHF dataset. Despite only chosen responses being used for supervision, rejected responses show a comparable likelihood of generation.

3.2 Revisiting Supervised Fine-tuning

Methods in RLHF often leverage SFT to ensure the stable update of the active policy (Schulman et al., 2017), using the SFT model as a reference policy. Even in non-RL alignment methods, empirical findings indicate that the SFT is crucial for achieving convergence to desired results (Rafailov et al., 2023; Tunstall et al., 2023). In detail, SFT uses cross-entropy loss to penalize the model if the predicted logits for the reference answers are low:

$$\mathcal{L}_{\text{SFT}}(\mathbf{x}, \mathbf{y}) = -\frac{1}{m}\sum_{k=1}^{m}\log P(\mathbf{x}^{(k)}, \mathbf{y}^{(k)}) \quad (2)$$

$$= -\frac{1}{m}\sum_{k=1}^{m}\sum_{i=1}^{|V|} y_i^{(k)} \cdot \log(p_i^{(k)}), \quad (3)$$

where y_i is a boolean value that indicates if ith token in the vocabulary set V is a label token, p_i refers to the probability of ith token, and m is the length of sequence. Cross-entropy alone gives no direct penalty or compensation for the logits of non-answer tokens (Lin et al., 2017) as y_i will be set to 0. While cross-entropy is generally effective for domain adaptation (Mao et al., 2023), there are no mechanisms to penalize rejected responses when compensating for the chosen responses. Therefore, the log probabilities of the tokens in the rejected responses increase along with the chosen responses, which is not desired from the viewpoint of preference alignment.

Generalization over both response styles We conduct a pilot study to empirically demonstrate the miscalibration of chosen and rejected responses with supervised fine-tuning alone. We fine-tune OPT-350M (Zhang et al., 2022) on *the chosen responses only* from the HH-RLHF dataset (Bai et al., 2022b). Throughout the training, we monitor the log probability of rejected responses for each batch and report this in Figure 3. Both the log probability of chosen and rejected responses exhibited a simultaneous increase. This can be interpreted from two different perspectives. First, the cross-entropy loss effectively guides the model toward the intended domain (e.g., dialogue). However, the absence of a penalty for unwanted generations results in rejected responses sometimes having even higher log probabilities than the chosen ones.

Penalizing undesired generations Appending an unlikelihood penalty to the loss has successfully reduced unwanted degenerative traits in models (Welleck et al., 2019; Li et al., 2020). For example, to prevent *repetitions*, an unwanted token set of previous contexts, $k \in \mathcal{C}_{recent}$, is disfavored by adding the following term to $(1 - p_i^{(k)})$ to the loss which penalizes the model for assigning high probabilities to recent tokens. Motivated by SFT ascribing high probabilities to rejected tokens (Figure 3) and the effectiveness of appending penalizing unwanted traits, we design a monolithic preference alignment method that dynamically penalizes the disfavored response for each query without the need for crafting sets of rejected tokens.

3.3 Objective Function of ORPO

The objective function of ORPO in Equation 4 consists of two components: 1) supervised fine-tuning (SFT) loss ($\mathcal{L}_{SFT}$); 2) odd ratio loss ($\mathcal{L}_{OR}$) for the tuple $d = (x, y_l, y_w) \sim D$:

$$\mathcal{L}(d;\theta) = \mathcal{L}_{SFT}\left(x, y_w;\theta\right) + \lambda \mathcal{L}_{OR}(d;\theta). \quad (4)$$

$\mathcal{L}_{SFT}$ follows the conventional NLL loss formulation in Equation (3) to maximize the likelihood of generating the reference tokens as previously discussed in Section 3.2. $\mathcal{L}_{OR}$ maximizes the odds ratio between the likelihood of generating the favored response y_w and the disfavored response y_l:

$$\mathcal{L}_{OR}\left(d;\theta\right) = -\log\sigma\left(\log\frac{\mathbf{odds}_\theta(y_w|x)}{\mathbf{odds}_\theta(y_l|x)}\right). \quad (5)$$

We wrap the log odds ratio with the log sigmoid function so that $\mathcal{L}_{OR}$ could be minimized by increasing the log odds ratio between y_w and y_l.

Weighting the $\mathcal{L}_{OR}$ term with λ tailors the pre-trained language model to adapt to the specific subset of the desired domain and disfavor generations in the rejected response sets. We further discuss the role of the odds ratio in learning the preference in Appendix D, showing that a simple probability ratio could lead to an excessive likelihood margin between the chosen and rejected responses.

3.4 Gradient Analysis

We provide the derivation of $\nabla_\theta \mathcal{L}_{OR}$ in Appendix A. Analysis of the gradient of $\mathcal{L}_{OR}$ justifies using the odds ratio loss. Where the two terms,

$$\nabla_\theta \mathcal{L}_{OR} = \delta(d) \cdot h(d), \quad (6)$$

have complementary roles: $\delta(d)$ penalizes the wrong predictions of the model and $h(d)$ contrasts between chosen and rejected responses:

$$\delta(d) = \left[1 + \frac{\mathbf{odds}_\theta P(y_w|x)}{\mathbf{odds}_\theta P(y_l|x)}\right]^{-1} \quad (7)$$

$$h(d) = \frac{\nabla_\theta \log P_\theta(y_w|x)}{1 - P_\theta(y_w|x)} - \frac{\nabla_\theta \log P_\theta(y_l|x)}{1 - P_\theta(y_l|x)}. \quad (8)$$

When the odds of the favored responses are relatively higher than the disfavored responses, $\delta(d)$ in Equation 7 will converge to 0. This indicates that the $\delta(d)$ will play the role of a penalty term, accelerating the parameter updates if the model is more likely to generate the rejected responses.

In equation 8, $h(d)$ implies a weighted contrast between the gradients from the chosen and rejected responses. Specifically, the term $1 - P(y|x)$ in the denominators amplifies the gradients when the corresponding side of the likelihood, $P(y|x)$, is high. For chosen responses, this will accelerate the model's adaptation toward the distribution of chosen responses as the likelihood increases.

4 Experimental Settings

4.1 Training Configurations

Models and datasets We finetune state-of-the-art pre-trained language models with ORPO by different scales, Phi-2 (2.7B) (Javaheripi and Bubeck, 2023), Llama-2 (7B) (Touvron et al., 2023) and Mistral (7B) (Jiang et al., 2023) on Binarized UltraFeedback (Tunstall et al., 2023). Furthermore, to assess the controlled scalability of ORPO, we finetune a series of OPT models (Zhang et al., 2022)

Model Name	Size	$\textbf{AlpacaEval}_{\textbf{1.0}}$	$\textbf{AlpacaEval}_{\textbf{2.0}}$
Phi-2 + SFT	2.7B	48.37% (1.77)	0.11% (0.06)
Phi-2 + SFT + DPO	2.7B	50.63% (1.77)	0.78% (0.22)
Phi-2 + ORPO *(Ours)*	2.7B	**71.80% (1.59)**	**6.35% (0.74)**
Llama-2 Chat *	7B	71.34% (1.59)	4.96% (0.67)
Llama-2 Chat *	13B	81.09% (1.38)	7.70% (0.83)
Llama-2 + ORPO *(Ours)*	7B	**81.26% (1.37)**	**9.44% (0.85)**
Zephyr (α) *	7B	85.76% (1.23)	8.35% (0.87)
Zephyr (β) *	7B	90.60% (1.03)	10.99% (0.96)
Mistral-ORPO-α *(Ours)*	7B	87.92% (1.14)	11.33% (0.97)
Mistral-ORPO-β *(Ours)*	7B	**91.41% (1.15)**	**12.20% (0.98)**

Table 1: Table of instruction-following abilities of each checkpoint measured through AlpacaEval. While clearly showing the improvements in instruction-following abilities after training with ORPO, it is notable that ORPO models exceed RLHF or DPO models of Llama-2 and Mistral (* indicates the results from the official leaderboard.)

scaling from 125M to 1.3B parameters on Anthropic's HH-RLHF (Bai et al., 2022a) and UltraFeedback, comparing supervised fine-tuning (SFT), proximal policy optimization (PPO), direct preference optimization (DPO), and compare these to ORPO. PPO and DPO models were fine-tuned with TRL (von Werra et al., 2020) on SFT models trained for a single epoch on the chosen responses following Rafailov et al. (2023) and Tunstall et al. (2023). We notate this by prepending "+" to each algorithm (e.g., +DPO). Further training details are in Appendix C. We filtered out instances where $y_w = y_l$ or where $y_w = \emptyset$, or $y_l = \emptyset$.

Reward models We train OPT-350M and OPT-1.3B on each dataset for a single epoch for reward modeling with the objective function in Equation 9 (Ziegler et al., 2020). The OPT-350M reward model was used for PPO, and OPT-1.3B reward model was used to assess the generations of fine-tuned models. We refer to these reward models as RM-350M and RM-1.3B in Section 5.

$$-\mathbb{E}_{(x,y_l,y_w)}\left[\log\sigma\left(r(x,y_w)-r(x,y_l)\right)\right] \qquad (9)$$

4.2 Leaderboard Evaluation

In Section 5.1, we evaluate the models using the $\text{AlpacaEval}_{1.0}$ and $\text{AlpacaEval}_{2.0}$ (Li et al., 2023b) benchmarks, comparing ORPO to other instruction-tuned models reported in the official leaderboard,[3] including Llama-2 Chat (7B) and (13B) (Touvron et al., 2023), and Zephyr α and β (Tunstall et al., 2023). Similarly, in Section 5.2, we evaluate the models with MT-Bench (Zheng et al., 2023) and report the results and the scores of the same models reported in the official leaderboard.[4] Using GPT-4 (Achiam et al., 2023) as an evaluator in $\text{AlpacaEval}_{1.0}$, we assess if the trained model can be preferred over the responses generated from `text-davinci-003`. For $\text{AlpacaEval}_{2.0}$, we used GPT-4-turbo[5] as an evaluator following the default setting. We assess if the generated responses are favored over those generated from GPT-4. Finally, using GPT-4 as an evaluator in MT-Bench, we check if the models can follow the instructions with hard answers in a multi-turn conversation.

5 Experimental Results

First, we assess the general instruction-following abilities of the models by comparing the preference alignment algorithms in single-turn (Section 5.1) and multi-turn (Section 5.2) instruction following benchmarks. Then, we compare ORPO against other alignment methods in the controlled setting, using OPT with various model sizes (Section 5.3).

5.1 Single-turn Instruction Following

Phi-2 (2.7B) ORPO improved pre-trained Phi-2 to exceed the performance of the Llama-2 Chat instruction-following language model by *only using UltraFeedback* as the instruction-tuning dataset, as shown in Table 1. λ of 0.25 was applied for Phi-2, resulting in 71.80% and 6.35% in AlpacaEval.

Llama-2 (7B) Notably, UltraFeedback and ORPO with λ of 0.2 on Llama-2 (7B) resulted in higher

[3] https://tatsu-lab.github.io/alpaca_eval/

[4] https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard

[5] https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo

AlpacaEval scores than the chat versions of both 7B and 13B scale trained with RLHF, eventually showing 81.26% and 9.44% in both AlpacaEvals.

In contrast, in our controlled experimental setting of conducting one epoch of SFT and three epochs of DPO following Tunstall et al. (2023) and Rafailov et al. (2023), Llama-2 + SFT and Llama-2 + SFT + DPO yielded models with outputs that could not be evaluated. This supports the efficacy of ORPO, in which the model can rapidly learn the desired domain and the preference with limited data. This aligns with the $h(d)$ examination in the gradient of our method studied in Section 3.4.

Mistral-ORPO-α (7B) Furthermore, fine-tuning Mistral (7B) with single-turn conversation dataset, UltraFeedback, and ORPO with λ of 0.1 outperforms Zephyr series, which are the Mistral (7B) models fine-tuned with SFT on 200K UltraChat (Ding et al., 2023) and DPO on the full UltraFeedback. As shown in Table 1, Mistral-ORPO-α (7B) achieves 87.92% and 11.33%, which exceeds Zephyr α by 1.98% and Zephyr β by 0.34% in AlpacaEval$_{2.0}$. The sample responses and corresponding references from GPT-4 can be found in Appendix I.

Mistral-ORPO-β (7B) Using the same configuration of Mistral-ORPO-α (7B), we additionally compare fine-tuning Mistral on the cleaned version of the UltraFeedback (Bartolome et al., 2023) to demonstrate the effect of the data quality. While the sizes of datasets are similar, ORPO gains further advantages from the dataset quality by scoring over 91% and 12% on AlpacaEval, as shown in Table 1. Further evaluations on instruction-following with IFEval (Zhou et al., 2023c) and four different benchmarks are in the Appendices E and F.

5.2 Multi-turn Instruction Following

With our best model, Mistral-ORPO-α (7B) and Mistral-ORPO-β (7B), we also assess the multi-turn instruction-following skills with deterministic answers (e.g., math) through MT-Bench.

As shown in Table 2, Mistral-ORPO series exceeds larger instruction-following chat models, especially Llama-2-Chat (70B). Eventually, Mistral-ORPO-α (7B) and Mistral-ORPO-β (7B) scored 7.23 and 7.32 in MT-Bench without being exposed to the multi-turn conversation dataset during training, while Zephyr-β was trained on 200k multi-turn conversations during SFT.

MT-Bench	1st Turn	2nd Turn	Average
Llama-2-7B Chat	6.41	6.13	6.27
Llama-2-13B Chat	7.06	6.24	6.65
Llama-2-70B Chat	6.99	6.73	6.86
Mistral-ORPO-α	7.49	6.96	7.23
Zephyr-β	**7.68**	6.98	**7.33**
Mistral-ORPO-β	7.64	**7.00**	7.32

Table 2: MT-Bench results of instruction-following language models. Mistral-ORPO-β surpasses or is on par with the models trained on more data.

5.3 Case Study with Smaller Models

We assess the win rate of ORPO over other preference alignment methods using different scales of OPT models, including supervised fine-tuning (SFT), PPO, and DPO, using RM-1.3B to understand the effectiveness of ORPO in Tables 3 and 4. Additionally, we visually verify that ORPO can effectively enhance the expected reward compared to SFT in Figure 4.

HH-RLHF In Table 3, ORPO outperforms SFT and PPO across all model scales. The highest win rate against SFT and PPO across the size of the model was 78.0% and 79.4%, respectively. Meanwhile, the win rate over DPO correlated to the model's size, with the largest model having the highest win rate: 70.9%.

ORPO vs	SFT	+DPO	+PPO
OPT-125M	84.0 (0.62)	41.7 (0.77)	66.1 (0.26)
OPT-350M	82.7 (0.56)	49.4 (0.54)	79.4 (0.29)
OPT-1.3B	78.0 (0.16)	70.9 (0.52)	65.9 (0.33)

Table 3: Average win rate (%) and its standard deviation of ORPO and standard deviation over other methods on **HH-RLHF** dataset for three rounds. Sampling decoding with a temperature of 1.0 was used on the test set.

UltraFeedback The win rate in UltraFeedback followed similar trends to what was reported in HH-RLHF, as shown in Table 4. ORPO was preferred over SFT and PPO for maximum 80.5% and 85.8%, respectively. While consistently preferring ORPO over SFT and PPO, the win rate over DPO gradually increases as the model size increases. The scale-wise trend exceeding DPO will be further shown through 2.7B models in Section 5.1.

6 Analysis

In this section, we study the advantages of ORPO from different perspectives. We first compare the

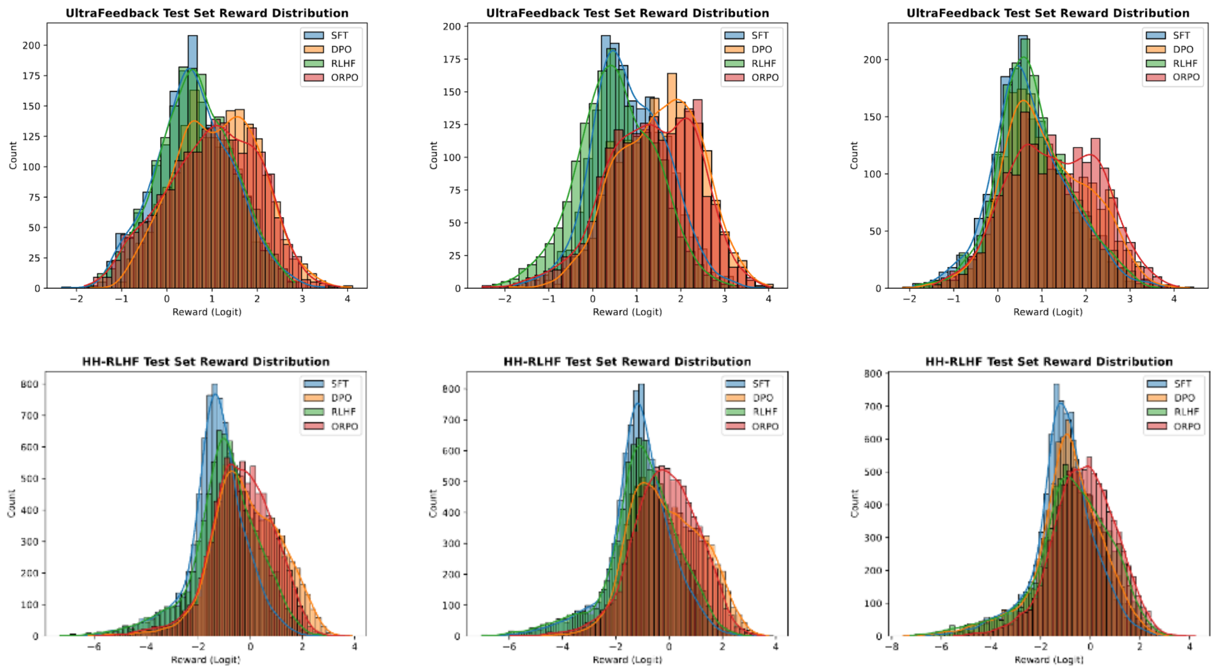


Figure 4: Reward distribution comparison between OPT-125M (left), OPT-350M (middle), and OPT-1.3B (right) trained with SFT (blue), RLHF (green), DPO (orange), and ORPO (red) on the test set of UltraFeedback (top) and HH-RLHF (bottom) using the RM-1.3B. While the rewards of the trained models are roughly normal and preference optimization algorithms (RLHF, DPO, and ORPO) tend to move the reward distribution in the positive direction, ORPO is on par or better than RLHF and DPO in increasing the expected reward.

ORPO vs	SFT	+DPO	+PPO
OPT-125M	73.2 (0.12)	48.8 (0.29)	71.4 (0.28)
OPT-350M	80.5 (0.54)	50.5 (0.17)	85.8 (0.62)
OPT-1.3B	69.4 (0.57)	57.8 (0.73)	65.7 (1.07)

Table 4: Average win rate (%) and its standard deviation of ORPO and standard deviation over other methods on **UltraFeedback** dataset for three rounds. Sampling decoding with a temperature of 1.0 was used.

computational costs of DPO and ORPO in Section 6.1. Then, we analyze ORPO from the reward maximization objective in Section 6.2. Furthermore, we measure the lexical diversity of the models trained with ORPO and DPO in Section 6.3.

6.1 Computational Efficiency

	DPO	ORPO
Training Time (hours) (↓)	12.6	**5.5**
Max Batch (↑)	1	**4**

Table 5: Computational costs of DPO and ORPO for 1 epoch on UltraFeedback using 2 NVIDIA A100 GPUs with AdamW and DeepSpeed ZeRO 3. We *exclude* SFT training time for DPO.

As depicted in Figure 2, ORPO's efficiency derives from two aspects: 1) no reference model and 2) no separate SFT stage. In that sense, ORPO is computationally more efficient than RLHF and DPO in both time and memory.

We demonstrate this through controlled training on DPO and ORPO with 2 NVIDIA A100 GPUs using DeepSpeed ZeRO 3(Rajbhandari et al., 2021) and AdamW (Loshchilov and Hutter, 2019) on Mistral (7B). As shown in Table 5, ORPO reduces 56.3% of training time compared to that of DPO and was able to fit a four times larger batch size per device. Furthermore, as DPO typically requires a pre-supervised fine-tuning (SFT) stage, the actual computational efficiency difference becomes more drastic as the SFT datasets are quite large[6]. The reference model (π_{SFT}) in the context of RLHF and DPO denotes the model trained with SFT, which will be the base model for updating the parameters with PPO or DPO (Ziegler et al., 2020; Rafailov et al., 2023). Thus, two π_{SFT}s, a frozen reference model and the model undergoing tuning, are required during training. Furthermore, in theory, two forward passes should be calculated for each model to acquire the logits for the chosen and rejected responses. In other words, four forward passes

[6]UltraChat dataset consists of 200k instances

	Per Input↓	Across Input↓
Phi-2 + SFT + DPO	**0.8012**	0.6019
Phi-2 + ORPO	0.8909	**0.5173**
Llama-2 + SFT + DPO	**0.8889**	0.5658
Llama-2 + ORPO	0.9008	**0.5091**

Table 6: Lexical diversity of Phi-2 and Llama-2 fine-tuned with DPO and ORPO. Lower cosine similarity is equivalent to higher diversity. The highest value in each column within the same model family is bolded.

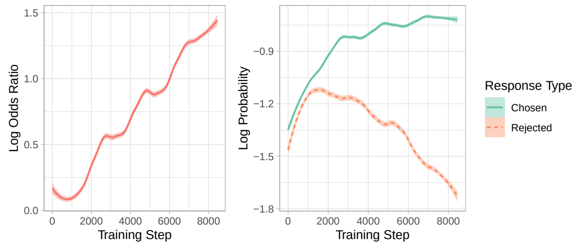


Figure 5: Average log-likelihood for chosen and rejected responses and log odds ratio per batch. The odds consistently increase during training with ORPO.

happen for a single batch. On the other hand, a reference model is not required in ORPO as π_{SFT} is directly updated. This leads to half the number of forward passes required for each batch.

6.2 Overall Reward Distribution

In addition to the reward model win rate discussed in Section 5.3, we compare the reward distribution of the responses generated with respect to the test set of the UltraFeedback and HH-RLHF in Figure 4. Regarding the SFT reward distribution as a default, PPO, DPO, and ORPO shift it in both datasets. However, the magnitude of reward shifts for each algorithm differs.

In Figure 4, RLHF (i.e., SFT + PPO) has some abnormal properties of the distribution with a low expected reward. We attribute this to empirical evidence of the instability and reward mismatch problem (Rafailov et al., 2023; Gao et al., 2022; Shen et al., 2023) as the RLHF models were trained with RM-350M and assessed with RM-1.3B. Meanwhile, it is notable that the ORPO distribution (red) is mainly located on the very right side of each subplot, indicating higher expected rewards. Recalling the intent of preference alignment, the distributions in Figure 4 indicate that ORPO tends to fulfill the aim of preference alignment for all model sizes.

6.3 Lexical Diversity

The lexical diversity of the preference-aligned language models was studied in previous works (Kirk et al., 2024). We expand the concept of per-input and across-input diversity introduced in Kirk et al. (2024) by using Gemini-Pro (Gemini Team et al., 2023) as an embedding model, which is suitable for assessing the diversity of instruction-following language models by encoding a maximum of 2048 tokens. The diversity metric with the given set of sampled responses is defined as:

$$\mathcal{O}_\theta^i := \{y_j \sim \theta(y|x_i) | j = 1, 2, ..., K\} \tag{10}$$

$$D(\mathcal{O}_\theta^i) = \frac{1}{2} \cdot \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \cos(h_i, h_j)}{N \cdot (N-1)} \tag{11}$$

where $\cos(h_i, h_j)$ refers to the cosine similarity between the embedding h_i and h_j. 5 different responses are sampled with a temperature of 1.0 to 160 queries in AlpacaEval (i.e., $K = 5, N = 160$) using Phi-2 and Llama-2 trained with ORPO and DPO. We report the results in Table 6.

Per Input Diversity (PID) We average the input-wise average cosine similarity between the generated samples with Equation 12 to assess the per-input diversity. In Table 6, ORPO models have the highest average cosine similarity in the first column for both models, which implies the lowest diversity per input. This indicates that ORPO generally assigns high probabilities to the desired tokens, while DPO has a relatively smoother logit distribution.

$$\text{PID}_D(\theta) = \frac{1}{N} \sum_{i=1}^{N} D(\mathcal{O}_\theta^i) \tag{12}$$

Across Input Diversity (AID) Using 8 samples generated per input, we sample the first item for each input and examine their inter cosine similarity with Equation 13 for across-input diversity. Unlike per-input diversity, it is noteworthy that Phi-2 (ORPO) has lower average cosine similarity in the second row of Table 6. We can infer that ORPO triggers the model to generate more instruction-specific responses than DPO.

$$\text{AID}_D(\theta) = D\left(\bigcup_{i=1}^{N} \mathcal{O}^i{}_{,\theta,j=1}\right) \tag{13}$$

6.4 Minimizing $\mathcal{L}_{OR}$

We demonstrate that models trained with ORPO learned to reflect the preference throughout the training process. We monitored the log probabilities of the chosen and rejected responses and the log odds ratio with $\lambda = 1.0$. With the same dataset and model as Figure 3, Figure 5 shows that the log probability of rejected responses is diminishing while that of chosen responses is on par with Figure 3 as the log odds ratio increases. This indicates that ORPO is successfully preserving the domain adaptation of SFT while the penalty term L_{OR} induces the model to lower the likelihood of unwanted generations. We discuss the effect of λ in Equation 4 in Appendix G, studying the proclivity of the log probability margin between the favored and disfavored responses with respect to λ.

7 Conclusion

In this paper, we introduced a reference-free monolithic preference alignment method, odds ratio preference optimization (ORPO), by revisiting and understanding the value of the supervised fine-tuning (SFT) phase in the context of preference alignment. ORPO was consistently preferred by the fine-tuned reward model against SFT and RLHF across the scale, and the win rate against DPO increased as the model size increased. Furthermore, we validate the scalability of ORPO with 2.7B and 7B pre-trained language models by exceeding the larger state-of-the-art instruction-following language models in AlpacaEval. Specifically, Mistral-ORPO-α and Mistral-ORPO-β achieved 11.33% and 12.20% in AlpacaEval$_{2.0}$, 7.23 and 7.32 in MT-Bench, thereby underscoring the efficiency and effectiveness of ORPO.

Limitations

While conducting a comprehensive analysis of the diverse preference alignment methods, including DPO and RLHF, we did not incorporate a more comprehensive range of preference alignment algorithms. We leave the broader range of comparison against other methods as future work, along with scaling our method to over 7B models. In addition, we will expand the fine-tuning datasets into diverse domains and qualities, thereby verifying the generalizability of our method in various NLP downstream tasks. Finally, we would like to study the internal impact of our method on the pre-trained language model, expanding the understanding of preference alignment procedure to not only the supervised fine-tuning stage but also consecutive preference alignment algorithms.

Acknowledgements

This work was supported by Institute for Information & communications Technology Promotion(IITP) grants funded by the Korea government(MSIT) (RS-2024-00398115, Research on the reliability and coherence of outcomes produced by Generative AI), and (RS-2019-II190075, Artificial Intelligence Graduate School Program(KAIST)).

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*.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models.

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022b. Constitutional ai: Harmlessness from ai feedback.

Alvaro Bartolome, Gabriel Martin, and Daniel Vila. 2023. Notus. https://github.com/argilla-io/notus.

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, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tianchi Cai, Xierui Song, Jiyan Jiang, Fei Teng, Jinjie Gu, and Guannan Zhang. 2023. Ulma: Unified language model alignment with demonstration and pointwise human preference. *ArXiv*, abs/2312.02554.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

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.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment.

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

Leo Gao, John Schulman, and Jacob Hilton. 2022. Scaling laws for reward model overoptimization.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus.

Nathan Grinsztajn, Yannis Flet-Berliac, Mohammad Gheshlaghi Azar, Florian Strub, Bill Wu, Eugene Choi, Chris Cremer, Arash Ahmadian, Yash Chandak, Olivier Pietquin, and Matthieu Geist. 2024. Averaging log-likelihoods in direct alignment.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced self-training (rest) for language modeling.

Hamish Haggerty and Rohitash Chandra. 2024. Self-supervised learning for skin cancer diagnosis with limited training data.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Mojan Javaheripi and Sébastien Bubeck. 2023. Phi-2: The surprising power of small language models.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. 2024. Understanding the effects of rlhf on llm generalisation and diversity.

Margaret Li, Stephen Roller, Ilia Kulikov, Sean Welleck, Y-Lan Boureau, Kyunghyun Cho, and Jason Weston. 2020. Don't say that! making inconsistent dialogue unlikely with unlikelihood training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4715–4728, Online. Association for Computational Linguistics.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023a. Self-alignment with instruction back-translation.

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

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023c. Textbooks are all you need ii: phi-1.5 technical report.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. 2023. Cross-entropy loss functions: Theoretical analysis and applications.

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

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. 2024. Disentangling length from quality in direct preference optimization. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 4998–5017, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model.

Samyam Rajbhandari, Olatunji Ruwase, Jeff Rasley, Shaden Smith, and Yuxiong He. 2021. Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

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.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Wei Shen, Rui Zheng, Wenyu Zhan, Jun Zhao, Shihan Dou, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. Loose lips sink ships: Mitigating length bias in reinforcement learning from human feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2859–2873, Singapore. Association for Computational Linguistics.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2023. Preference ranking optimization for human alignment.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2022. Learning to summarize from human feedback.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Binghai Wang, Rui Zheng, Lu Chen, Yan Liu, Shihan Dou, Caishuang Huang, Wei Shen, Senjie Jin, Enyu Zhou, Chenyu Shi, Songyang Gao, Nuo Xu, Yuhao Zhou, Xiaoran Fan, Zhiheng Xi, Jun Zhao, Xiao Wang, Tao Ji, Hang Yan, Lixing Shen, Zhan Chen, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. 2024. Secrets of rlhf in large language models part ii: Reward modeling.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. How far can camels go? exploring the state of instruction tuning on open resources.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Tianhao Wu, Banghua Zhu, Ruoyu Zhang, Zhaojin Wen, Kannan Ramchandran, and Jiantao Jiao. 2023. Pairwise proximal policy optimization: Harnessing relative feedback for llm alignment.

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*.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2019. Quick and (not so) dirty: Unsupervised selection of justification sentences for multi-hop question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2578–2589, Hong Kong, China. Association for Computational Linguistics.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. Rrhf: Rank responses to align language models with human feedback without tears.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. 2023. Pytorch fsdp: Experiences on scaling fully sharded data parallel.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. ArXiv:2306.05685 [cs].

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. Lima: Less is more for alignment.

Haotian Zhou, Tingkai Liu, Qianli Ma, Jianbo Yuan, Pengfei Liu, Yang You, and Hongxia Yang. 2023b. Lobass: Gauging learnability in supervised fine-tuning data. *ArXiv*, abs/2310.13008.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023c. Instruction-following evaluation for large language models.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences.

A Derivation of $\nabla_\theta \mathcal{L}_{OR}$ with Odds Ratio

Suppose that $g(x, y_l, y_w) = \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)}$

$$\nabla_\theta \mathcal{L}_{OR} = \nabla_\theta \log \sigma \left(\log \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \right) \tag{14}$$

$$= \frac{\sigma' \left(\log g(x, y_l, y_w) \right)}{\sigma \left(\log g(x, y_l, y_w) \right)} \tag{15}$$

$$= \frac{\sigma \left(- \log g(x, y_l, y_w) \right)}{g(x, y_l, y_w)} \cdot \nabla_\theta g(x, y_l, y_w) \tag{16}$$

$$= \sigma \left(- \log g(x, y_l, y_w) \right) \cdot \nabla_\theta \log g(x, y_l, y_w) \tag{17}$$

$$= \left(1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \right)^{-1} \cdot \nabla_\theta \log \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \tag{18}$$

In Equation 18, the remaining derivative can be further simplified by replacing $1 - P_\theta(y|x)$ terms where $P(y|x) = \sqrt[N]{\prod_t^N P_\theta(y_t|x, y_{<t})}$ in $\textbf{odds}_\theta(y|x)$ as follows.

$$\nabla_\theta \log \left(1 - P_\theta(y|x) \right) = \frac{\nabla_\theta \left(1 - P_\theta(y|x) \right)}{1 - P_\theta(y|x)} \tag{19}$$

$$= \frac{- \nabla_\theta P_\theta(y|x)}{1 - P_\theta(y|x)} \tag{20}$$

$$= - \frac{P_\theta(y|x)}{1 - P_\theta(y|x)} \cdot \nabla_\theta \log P_\theta(y|x) \tag{21}$$

$$= - \textbf{odds}_\theta(y|x) \cdot \nabla_\theta \log P_\theta(y|x) \tag{22}$$

$$\nabla_\theta \log \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} = \nabla_\theta \log \frac{P_\theta(y_w|x)}{P_\theta(y_l|x)} - \Big(\nabla_\theta \log(1 - P_\theta(y_w|x)) - \nabla_\theta \log(1 - P_\theta(y_l|x)) \Big) \tag{23}$$

$$= \left(1 + \textbf{odds}_\theta P(y_w|x) \right) \nabla_\theta \log P_\theta(y_w|x) - \left(1 + \textbf{odds}_\theta P(y_l|x) \right) \nabla_\theta \log P_\theta(y_l|x) \tag{24}$$

Therefore, the final form of $\nabla_\theta \mathcal{L}_{OR}$ would be

$$\nabla_\theta \mathcal{L}_{OR} = \frac{1 + \textbf{odds}_\theta P(y_w|x)}{1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)}} \cdot \nabla_\theta \log P_\theta(y_w|x) - \frac{1 + \textbf{odds}_\theta P(y_l|x)}{1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)}} \cdot \nabla_\theta \log P_\theta(y_l|x) \tag{25}$$

$$= \left(1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \right)^{-1} \cdot \left(\frac{\nabla_\theta \log P_\theta(y_w|x)}{1 - P(y_w|x)} - \frac{\nabla_\theta \log P_\theta(y_l|x)}{1 - P(y_l|x)} \right) \tag{26}$$

B Ablation on Probability Ratio and Odds Ratio

In this section, we continue the discussion in Appendix D through empirical results comparing the log probabilities of chosen and rejected responses in UltraFeedback when trained with probability and odds ratios. Recalling the sensitivity of each ratio discussed in Appendix D, it is expected for the probability ratio to lower the log probabilities of the rejected responses with a larger scale than the odds ratio. This is well-shown in Figure 6, which is the log probabilities of each batch while fine-tuning with probability ratio (left) rapidly reaches under -4, while the same phenomenon happens after the over-fitting occurs in the case of odds ratio (right).

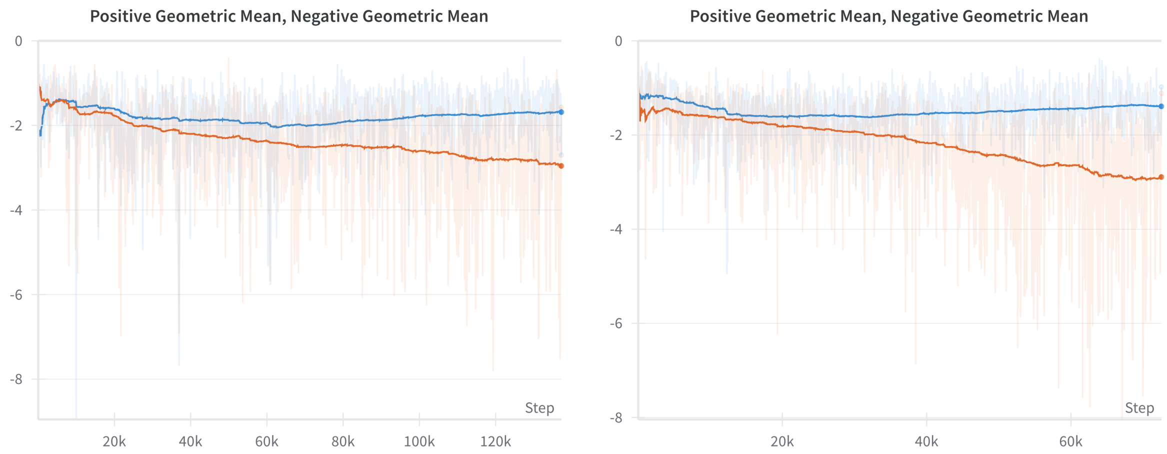


Figure 6: The log probability trace when the model is trained with the probability ratio (left) and the odds ratio (right) given the same hyperparameters. The probability ratio leads the rejected responses to have relatively lower log probabilities.

C Experimental Details

Flash-Attention 2 (Dao, 2023) is applied for all the pre-trained models for computational efficiency. In particular, the OPT series and Phi-2 (2.7B) were trained with DeepSpeed ZeRO 2 (Rasley et al., 2020), Llama-2 (7B) and Mistral (7B) were trained with Fully Sharded Data Parallel(FSDP) (Zhao et al., 2023). 7B and 2.7B models were trained with four and two NVIDIA A100, and the rest were trained on four NVIDIA A6000. For optimizer, AdamW optimizer (Loshchilov and Hutter, 2019) and paged AdamW (Dettmers et al., 2023) were used, and the linear warmup with cosine decay was applied for the learning rate. For input length, every instance was truncated and padded to 1,024 tokens and 2,048 tokens for HH-RLHF and UltraFeedback, respectively. To guarantee that the models can sufficiently learn to generate the proper response to the conversation history or the complex instruction, we filtered instances with prompts with more than 1,024 tokens.

Supervised Fine-tuning (SFT) For SFT, the maximum learning rate was set to 1e-5. Following Ziegler et al. (2020) and Rafailov et al. (2023), the training epoch is set to 1.

Reinforcement Learning with Human Feedback (RLHF) For RLHF, the hyperparameters were set as Table 7 for UltraFeedback. For the HH-RLHF dataset, the `output_min_length` and `output_max_length` were set to 64 and 256.

Direct Preference Optimization (DPO) For DPO, β was set to 0.1 for every case. The learning rate was set to 5e-6, and the model was trained for three epochs to select the best model by evaluation loss in each epoch. However, in most cases, the first or the second checkpoint was selected as the best model as the evaluation loss increased from the third epoch.

Hyperparameter	Setting
ppo_epoch	4
init_kl_coef	0.1
horizon	2,000
batch_size	64
mini_batch_size	8
gradient_accumulation_steps	1
output_min_length	128
output_max_length	512
optimizer	AdamW
learning_rate	1e-05
gamma	0.99

Table 7: Hyperparameter settings for RLHF.

Odds Ratio Preference Optimization (`ORPO`) As `ORPO` does not require any special hyperparameter, only the learning rate and epoch were the only hyperparameter to set. For `ORPO`, the maximum learning rate was set to 8e-6 and trained for 10 epochs. The best model is selected based on the lowest evaluation loss for the OPT series, Phi-2 (2.7B) and Llama-2 (7B).

D Comparison to Probability Ratio

The rationale for selecting the odds ratio instead of the probability ratio lies in its stability. The probability ratio for generating the favored response y_w over the disfavored response y_l given an input sequence x can be defined as:

$$\mathbf{PR}_\theta(y_w, y_l) = \frac{P_\theta(y_w|x)}{P_\theta(y_l|x)}. \tag{27}$$

While this formulation has been used in previous preference alignment methods that precede SFT (Rafailov et al., 2023; Azar et al., 2023), the odds ratio is a better choice in the setting where the preference alignment is incorporated in SFT as the odds ratio is more sensitive to the model's preference understanding. In other words, the probability ratio leads to more extreme discrimination of the disfavored responses than the odds ratio.

We visualize this through the sample distributions of the log probability ratio $\log \mathbf{PR}(X_2|X_1)$ and log odds ratio $\log \mathbf{OR}(X_2|X_1)$. We sample 50,000 samples each with Equation 28 and plot the log probability ratio and log odds ratio in Figure 7. We multiply β for the probability ratio as it is practiced in the probability ratio-based methods and report the cases where $\beta = 0.2$ and $\beta = 1.0$.

$$X_1, X_2 \sim \text{Unif}(0, 1) \tag{28}$$

$$Y \sim \beta \left(\log X_1 - \log X_2\right) \tag{29}$$

$$Y \sim \log \frac{X_1}{1 - X_1} - \log \frac{X_2}{1 - X_2} \tag{30}$$

Recalling that the log sigmoid function is applied to the log probability ratio and log odds ratio, each ratio's scale determines the expected margin between the likelihood of the favored and disfavored styles when the loss is minimized. In that sense, the contrast should be relatively extreme to minimize the log sigmoid loss when $\mathbf{PR}(X_2|X_1)$ is inputted instead of $\mathbf{OR}(X_2|X_1)$ to the log sigmoid function, regarding the sharp distribution of $\log \mathbf{PR}(X_2|X_1)$ in Figure 7. This results in overly suppressing the logits for the tokens in the disfavored responses in the setting where SFT and preference alignment are incorporated, as the model is not adapted to the domain. We empirically support this analysis through the ablation study in Appendix B. Therefore, the odds ratio is a better choice when the preference alignment is done with SFT

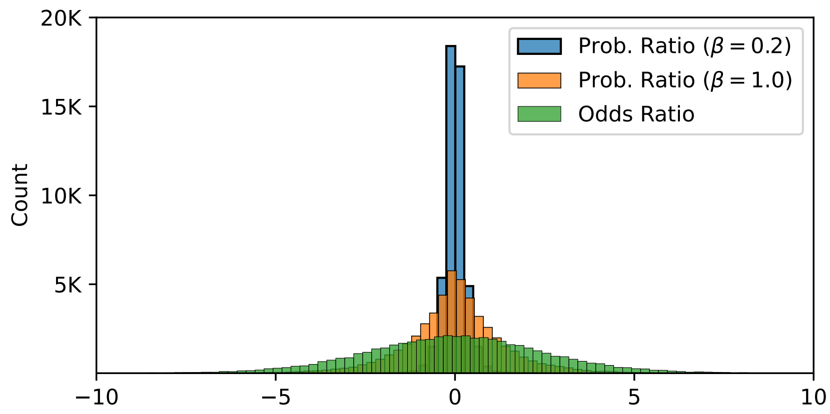


Figure 7: Sampled distribution of $\log \mathbf{PR}(X_2|X_1)$ and $\log \mathbf{OR}(X_2|X_1)$. $\log \mathbf{OR}(X_2|X_1)$ has a wider range given the same input probability pairs (X_1, X_2).

due to the mild discrimination of disfavored responses and the prioritizing of the favored responses to be generated.

Throughout fine-tuning, minimizing the log sigmoid loss leads to either $\mathbf{PR}(X_2|X_1)$ or $\mathbf{OR}(X_2|X_1)$ to be larger. This is equivalent to the rejected responses' token-wise likelihood, which will generally get smaller. In this context, it is essential to avoid an overly extreme contrast. This precaution is especially important given the sharp distribution of $\log \mathbf{PR}(X_2|X_1)$ depicted in Figure 7. The excessive margin could lead to the unwarranted suppression of logits for tokens in disfavored responses within the incorporated setting, potentially resulting in degeneration issues.

E IFEval Result for Mistral-ORPO-α and Mistral-ORPO-β

Along with the AlpacaEval results reported in Section 5.1, we report the results of Mistral-ORPO-α and Mistral-ORPO-β on IFEval (Zhou et al., 2023c), calculated with the codes from Gao et al. (2023).

Model Type	Prompt-Strict	Prompt-Loose	Inst-Strict	Inst-Loose
Mistral-ORPO-α	0.5009	0.5083	0.5995	0.6163
Mistral-ORPO-β	0.5287	0.5564	0.6355	0.6619

Table 8: IFEval scores of Mistral-ORPO-α and Mistral-ORPO-β.

F Benchmark Evaluation Result for Mistral-ORPO-β and Zephyr-β

We report the benchmark evaluation results of Mistral-ORPO-β and Zephyr-β on ARC (Yadav et al., 2019), MMLU (Hendrycks et al., 2021), HellaSwag (Zellers et al., 2019), and Winogrande (Sakaguchi et al., 2021). While Zephyr-β was trained on 200k multi-turn conversations with supervised fine-tuning and 61k single-turn conversations with DPO, Mistral-ORPO-β was trained on 61k single-turn with ORPO, highlighting the data efficiency of ORPO by being on par.

Model Name	ARC (25)	MMLU (5)	HellaSwag (10)	Winogrande (5)
Mistral-ORPO-β	60.7	59.5	**85.0**	**79.6**
Zephyr-β	**62.0**	**61.1**	84.4	77.7

Table 9: Language model benchmark evaluation results of Mistral-ORPO-β and Zehyr-β

G Ablation on the Weighting Value (λ)

For the weighting value λ in Equation 4, we conduct an ablation study with $\{0.1, 0.5, 1.0\}$. Mistral (7B) and UltraFeedback were used for the base model and dataset. In Section G.1, we compare the log probability trends by the value of λ, and we assess the downstream effect of λ in Section G.2.

G.1 Log Probability

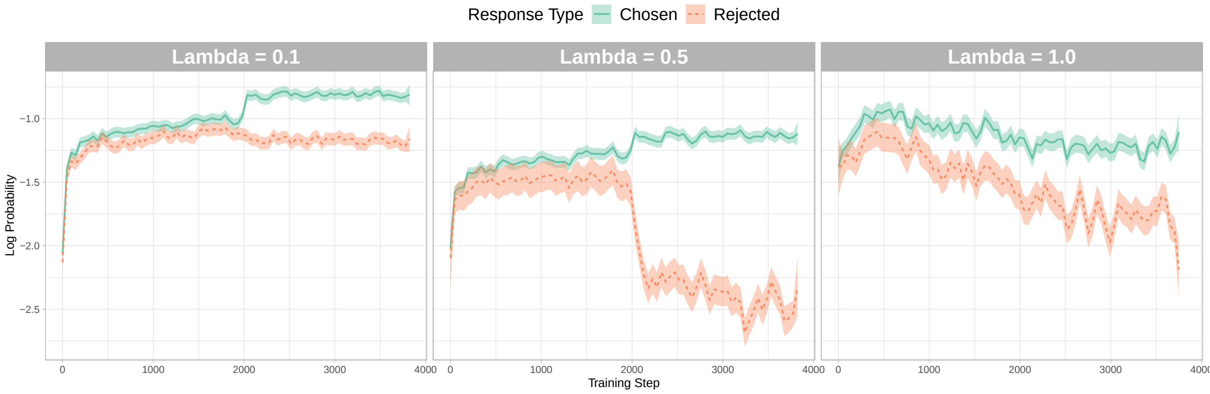


Figure 8: The log probability trend by λ. With larger λ (e.g., $\lambda = 1.0$), $\mathcal{L}_{OR}$ gets more influential in fine-tuning the models with ORPO.

In Figure 8, we find that larger λ leads to stronger discrimination of the rejected responses in general. With $\lambda = 0.1$, the average log probability of the chosen and the rejected responses stay close as the fine-tuning proceeds. Also, unlike other settings, the log probabilities for the rejected responses do not decrease, but rather, the log probabilities of the chosen responses increase to minimize $\mathcal{L}_{OR}$ term.

Moreover, in $\lambda = 0.5$, there exists a similar trend of further increasing the log probabilities of the chosen responses, but the log probabilities of the rejected responses are diminishing simultaneously. Lastly, in $\lambda = 1.0$, the chosen responses diminish along with the rejected responses while enlarging the margin between them. However, this does not mean smaller λ is always the better. It will depend on the specific need and model.

G.2 MT-Bench

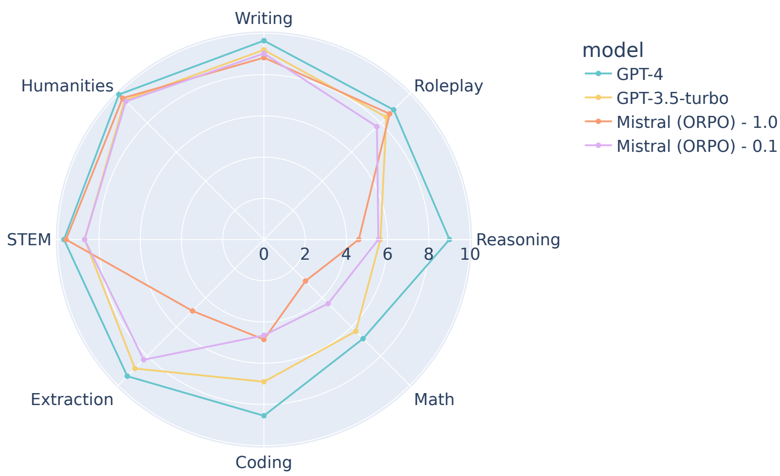
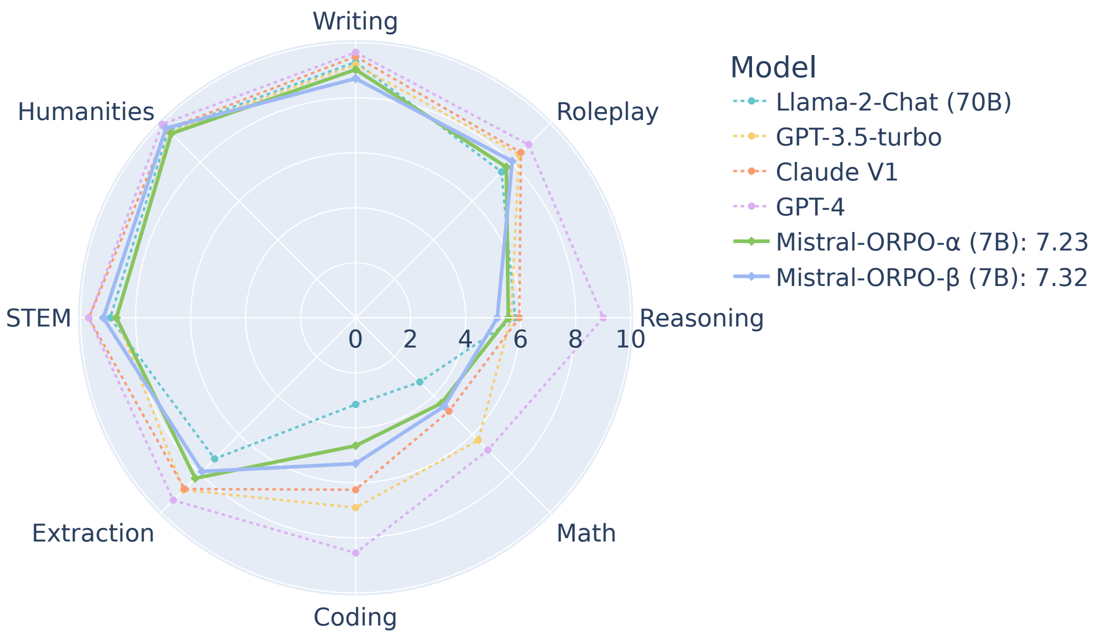


Figure 9: MT-Bench result comparison by differing $\lambda = 0.1$ and $\lambda = 1.0$.

The downstream impact of λ stands out in the MT-Bench result. In comparison to $\lambda = 0.1$, Mistral+ORPO (7B) with $\lambda = 1.0$ performs worse in extraction, math, and reasoning, which are the categories that generally require deterministic answers. On the other hand, it performs better in STEM, humanities, and roleplay, which ask the generations without hard answers. Along with the amount of discrepancy between the trend in the logits of chosen and rejected responses, we can infer that making a more significant margin between the chosen and the rejected responses through higher λ in ORPO leads to overly adapting to the chosen responses set in the training dataset. This proclivity results in open-ended generations generally being preferred by the annotator while showing weaker performance in the hard-answered questions.

H MT-Bench Result of Mistral-ORPO-α (7B) and Mistral-ORPO-β (7B)

For the MT-Bench result in Section 5.2, we report the category-wise scores of **Mistral-ORPO-α (7B)** and **Mistral-ORPO-β (7B)** in Figure 10. While surpassing Llama-2 Chat (13B) and Llama-2 Chat (70B) in most cases, **Mistral-ORPO-β (7B)** is comparable to GPT-3.5-turbo in the categories that require descriptive generations. However, it lacks coding and math skills, which we speculate is due to the lack of training data, as we used 61k instances in UltraFeedback.



Loading [MathJax]/extensions/MathMenu.js

Figure 10: MT-Bench result of **Mistral-ORPO-α (7B)** and **Mistral-ORPO-β (7B)** by the category.

I Generation Samples from Mistral-ORPO-α (7B)

In this section, we two sample generations from **Mistral-ORPO-α (7B)** on AlpacaEval, including the one which is preferred over the response of GPT-4 and the one in which GPT-4 was preferred. For readability, the instances with short responses are selected for both cases. Additionally, the responses for Mistral-ORPO-α (7B) were sampled with a temperature of 0.7.

Query 1 (GPT-4 Win): Give a brief description of the book that you are given. Matilda

---

1. **GPT-4**:
   "Matilda" is a children's novel written by the British author Roald Dahl. It was first published in 1988. The story revolves around a very intelligent and precocious young girl named Matilda Wormwood. Despite being neglected and mistreated by her ignorant and self-absorbed parents, Matilda finds solace in reading books and quickly develops a love for learning.

   At the age of five, she begins attending school and meets Miss Honey, a kindhearted teacher who recognizes Matilda's extraordinary intellectual abilities. However, the school's tyrannical headmistress, Miss Trunchbull, is a cruel and oppressive figure who takes pleasure in punishing the children with her unreasonable and harsh methods.

   Matilda discovers that she possesses a unique telekinetic ability, which she learns to control and use to her advantage. With her wit, intelligence, and special powers, Matilda sets out to defend her friends from Miss Trunchbull's abuses and to help Miss Honey overcome her own difficult past.

   The book is a tale of good triumphing over evil, and it celebrates the power of knowledge, kindness, and standing up for what is right. It has been adapted into a successful film, stage musical, and has remained a beloved classic in children's literature.

2. **Mistral-ORPO-β**:
   "Matilda" is a children's book written by Roald Dahl, published in 1988. The story revolves around the life of Matilda Wormwood, a brilliant and gifted five-year-old girl who has telekinetic powers. The story explores themes of education, family, and the importance of reading and intellectual curiosity. The book has been praised for its wit, humor, and powerful message, making it a classic and beloved children's book.

Query 2 (Mistral-ORPO-β Win): What do you think about ChatGPT?

1. **GPT-4**:
   As an AI developed by OpenAI, I don't have personal opinions. However, I can share that ChatGPT is an advanced language model that has received considerable attention for its ability to generate human-like text. It's designed to assist users in a wide range of tasks, from answering questions to creating content and engaging in conversation.

   ChatGPT has been praised for its fluency and versatility in handling various topics and for its potential to streamline and enhance communication in both personal and professional settings. However, like any technology, it also faces challenges and criticisms, such as ensuring the accuracy of information, preventing misuse, and managing ethical considerations surrounding AI-generated content.

   Overall, ChatGPT represents a significant achievement in natural language processing and has sparked discussions about the future of AI in our daily lives.

2. **Mistral-ORPO-β**:
   As an AI language model, I am an example of the technology used in ChatGPT. I can provide a personal opinion, but please keep in mind that I am designed to be socially unbiased and positive in nature.

   ChatGPT is an impressive advancement in artificial intelligence and natural language processing. It has the ability to understand and generate human-like text, making it possible for users to have engaging and informative conversations with the AI. This technology has the potential to revolutionize various industries, including customer support, education, content creation, and more.

   However, it's important to recognize that ChatGPT, like any AI technology, is not perfect. It may sometimes provide inaccurate or biased information due to the limitations of its training data or the complexity of human language. It's crucial to verify the facts and use critical thinking when interacting with AI systems like ChatGPT.

   Overall, ChatGPT represents a significant step forward in the field of AI, and its potential applications are vast. As a responsible user, it's important to approach this technology with a critical and respectful mindset, recognizing its strengths and limitations.