ORPO: Monolithic Odds Ratio Preference Optimization without Reference Model

Anonymous ACL submission

Abstract

While recently proposed preference alignment algorithms for language models have demonstrated promising results, supervised fine-tuning (SFT) remains imperative for achieving successful convergence in preference alignment. In this paper, we elaborate on the crucial role of SFT within the context of preference alignment, emphasizing that a minor penalty for the disfavored generation style is sufficient for preference-aligned SFT. Building on this foundation, we introduce a straightforward and innovative reference model-free monolithic odds ratio preference optimization algorithm, ORPO, eliminating the necessity for an additional preference alignment phase. Empirically and theoretically, we demonstrate that the odds ratio serves as a sensible choice for contrasting favored and unfavored styles during SFT. Specifically, fine-tuning Phi-2 (2.7B), Llama-2 (7B), and Mistral (7B) with ORPO on UltraFeedback alone surpasses the performance of state-of-the-art language models with more than 7B and 13B parameters, achieving 66.2%, 81.3%, and 87.94% in AlpacaEval.

1 Introduction

Pre-trained language models (PLMs) with a 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 general-domain applications (i.e., instruction-following), typically through *instruction tuning* and *model 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).

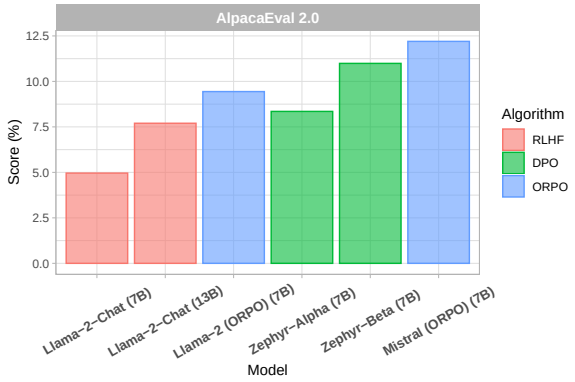

Figure 1: $\text{AlpacaEval}_{2.0}$ result of Llama-2 (7B) and Mistral (7B) fine-tuned with ORPO (blue) in comparison to the state-of-the-art models. Notably, Mistral (ORPO) surpasses Zephyr β and Llama-2-Chat (13B) with a single epoch training on the subset of UltraFeedback.

Preference alignment methods have demonstrated success in several downstream tasks beyond reducing harm. For example, improving factuality (Tian et al., 2023; Cheng et al., 2024; Chen and Li, 2024), code-based question answering (Gorbatovski and Kovalchuk, 2024), and machine translation (Ramos et al., 2023). The versatility of alignment algorithms over a wide range of downstream tasks highlights the necessity of both understanding the alignment procedure and further improving the algorithms in terms of efficiency and performance. However, existing preference alignment methods are normally a multi-stage process, as shown in Figure 2, typically requiring a second reference model and a separate warm-up phase with supervised fine-

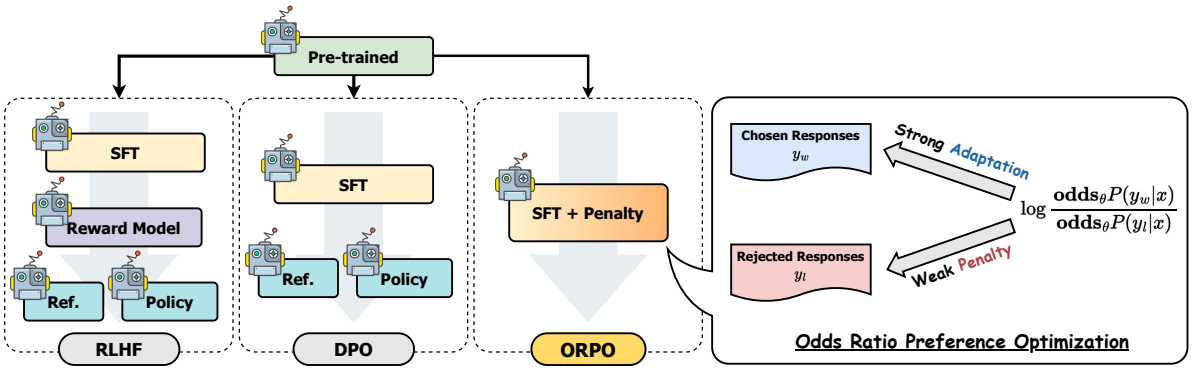

Figure 2: General diagram of preference alignment with ORPO. ORPO aligns the pre-trained language model in a non-segmented manner by giving a weak penalty to the rejected responses and a strong adaptation signal to the chosen responses with a simple log odds ratio term appended to the negative log-likelihood loss.

tuning (SFT) (Ziegler et al., 2020; Rafailov et al., 2023; Wu et al., 2023).

In this paper, we study the impact of SFT in pairwise preference datasets and propose a simple and novel monolithic alignment method, odds ratio preference optimization (ORPO), which efficiently penalizes the model from learning undesired generation styles during SFT. Unlike previous works, our approach requires neither an SFT warm-up stage nor a reference model, enabling resource-efficient development of preference-based aligned models. We study the theoretical justification of using the odds ratio in Sections 4.3 and 4.4, and empirically show that our method benefits from the scale through fine-tuning 125M to 7B models in Section 6, including Llama-2 (7B) and Mistral (7B). As our best model, we present Mistral ORPO (7B), which is Mistral (7B) trained on the 32K subset of UltraFeedback (Tunstall et al., 2023) alone with ORPO for a single epoch and surpasses Llama-2-Chat (7B) and (13B) (Touvron et al., 2023) and Zephyr (β) (Tunstall et al., 2023) by achieving 87.94% and 12.20% in $\text{AlpacaEval}_{1.0}$ and $\text{AlpacaEval}_{2.0}$ (Li et al., 2023b), as shown Figure 1.

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). Notably, Ouyang et al. (2022) demonstrated the scalability and versatility of RLHF for instruction-following language models. Extensions such as language model feedback (RLAIF) could serve as a viable alternative to human feedback (Bai et al., 2022b; Lee et al., 2023; Pang et al., 2023). 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 crucial need for stable preference alignment algorithms.

Alignment without Reward Model Several 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., 2023). Rafailov et al. (2023) introduce direct policy optimization (DPO), which merged the reward modeling stage into the preference learning stage. Azar et al. (2023) prevented potential overfitting problems in DPO through identity preference optimization (IPO). Ethayarajh et al. (2023) and Cai et al. (2023) proposed Kahneman-Tversky Optimisation (KTO) and Unified Language Model Alignment (ULMA) that does not require the pairwise preference dataset, unlike RLHF and DPO. Song et al. (2023) further suggests 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 In common, both preference alignment methods with and without reinforcement learning mostly re-

quire supervised fine-tuning (SFT) (i.e., reference model). In contrast, there have been approaches to build human-aligned language models with SFT exclusively (Zhou et al., 2023a; Li et al., 2023a; Haggerty and Chandra, 2024; Zhou et al., 2023b). Zhou et al. (2023a) demonstrated that SFT with a small amount of data with fine-grained filtering and 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, while Zhou et al. (2023b) suggested that the selected subset of preference dataset is sufficient for alignment. While these works highlight the impact and significance of SFT in the context of preference alignment, the actual role of SFT and the theoretical background for incorporating preference alignment in SFT is still understudied.

3 The Role of Supervised Fine-tuning

We study the role of supervised fine-tuning (SFT) as an initial stage of preference alignment methods (Ziegler et al., 2020; Rafailov et al., 2023) through analysis of the loss function in SFT and empirical demonstration of the preference comprehension ability of the trained SFT model. SFT plays a significant role in tailoring the pre-trained language models to the desired domain (Zhou et al., 2023a; Dong et al., 2024) by increasing the log probabilities of pertinent tokens. Nevertheless, this inadvertently increases the likelihood of generating tokens in undesirable styles, as illustrated in Figure 3. Therefore, it is necessary to explore the method that is capable of preserving the domain adaptation role of SFT while concurrently discerning and mitigating unwanted generation styles.

Absence of Penalty in Cross-Entropy Loss The goal of cross-entropy loss model fine-tuning is to penalize the model if the predicted logits for the reference answers are low, as shown in Equation 2.

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

$$= -\frac{1}{m}\sum_{k=1}^{m}\sum_{i=1}^{|V|} y_i^{(k)} \cdot \log(p_i^{(k)}) \quad (2)$$

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. Using cross-entropy alone gives no 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 the rejected responses and compensate for the chosen responses. Therefore, we can infer that the increase in the relevant logits will happen invariant to the preference.

Generalization over Both Response Styles To empirically demonstrate the miscalibration of chosen and rejected responses with supervised fine-tuning, we conduct a pilot study. We fine-tune OPT-350M (Zhang et al., 2022) on *the chosen responses only* with HH-RLHF (Bai et al., 2022b). Throughout the training, we monitor the log probability of rejected responses for each batch and plot this in Figure 3.

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.

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., multi-turn conversation). However, the absence of a penalty for unwanted generations results in rejected responses sometimes having even higher log probabilities than the chosen ones.

4 Methodology

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 loss for differentiating the generation styles between favored and disfavored responses.

4.1 Preliminaries

Given an input sequence x, the average log-likelihood of generating the output sequence y, of length m tokens, is computed as Equation 3. The odds of generating the output sequence y is defined in Equation 4:

$$\log P_\theta(y|x) = \frac{1}{m}\sum_{t=1}^{m} \log P_\theta(y_t|x, y_{<t}) \quad (3)$$

$$\textbf{odds}_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)} \quad (4)$$

Intuitively, $\textbf{odds}_\theta(y|x) = k$ implies that it is k times more likely for the model 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, $\textbf{OR}_\theta(y_w, y_l)$, indicates how much more likely it is for the model θ to generate y_w than y_l given input x, defined in Equation 5.

$$\textbf{OR}_\theta(y_w, y_l) = \frac{\textbf{odds}_\theta(y_w|x)}{\textbf{odds}_\theta(y_l|x)} \quad (5)$$

4.2 Odds Ratio Preference Optimization

The objective function of ORPO in Equation 6 consists of two components: 1) supervised fine-tuning (SFT) loss ($\mathcal{L}_{SFT}$); 2) relative ratio loss ($\mathcal{L}_{Ratio}$).

$$\mathcal{L}_{ORPO} = \mathbb{E}_{(x,y_w,y_l)}\left[\mathcal{L}_{SFT} + \mathcal{L}_{Ratio}\right] \quad (6)$$

$\mathcal{L}_{SFT}$ follows the conventional causal language modeling negative log-likelihood (NLL) loss function to maximize the likelihood of generating the reference tokens as previously discussed in Section 3. $\mathcal{L}_{Ratio}$ in Equation 7 maximizes the odds ratio between the likelihood of generating the disfavored response y_w and the disfavored response y_l. We wrap the log odds ratio with the log sigmoid function so that $\mathcal{L}_{Ratio}$ could be minimized by increasing the log odds ratio between y_w and y_l.

$$\mathcal{L}_{Ratio} = -\log\sigma\left(\log\frac{\textbf{odds}_\theta(y_w|x)}{\textbf{odds}_\theta(y_l|x)}\right) \quad (7)$$

Together, $\mathcal{L}_{SFT}$ and $\mathcal{L}_{Ratio}$ tailor the pre-trained language model to adapt to the specific subset of the desired domain, which excludes the type of generations in the rejected response sets.

4.3 Why Odds Ratio?

The rationale for selecting the odds ratio instead of the probability ratio as a penalty term lies in its stability. The probability ratio for generating the disfavored response y_w over the disfavored response y_l given the input sequence x can be defined as Equation 8.

$$\textbf{PR}_\theta(y_w, y_l) = \frac{P_\theta(y_w|x)}{P_\theta(y_l|x)} \quad (8)$$

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 $\log \textbf{PR}(X_2|X_1)$ and log odds ratio $\log \textbf{OR}(X_2|X_1)$. We sample 50,000 samples each with Equation 9 and plot the log probability ratio and log odds ratio in Figure 4. 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 $\beta = 1.0$.

$$X_1, X_2 \sim \text{Unif}(0, 1) \quad (9)$$

$$Y \sim \beta\left(\log X_1 - \log X_2\right) \quad (10)$$

$$Y \sim \log\frac{X_1}{1 - X_1} - \log\frac{X_2}{1 - X_2} \quad (11)$$

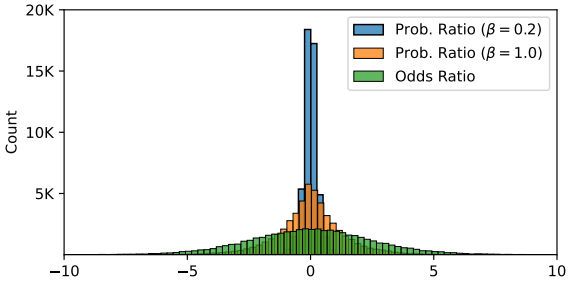

Figure 4: Sampled distribution of $\log \textbf{PR}(X_2|X_1)$ and $\log \textbf{OR}(X_2|X_1)$. $\log \textbf{OR}(X_2|X_1)$ has a wider range given the same input probability pairs (X_1, X_2).

Recalling that the log sigmoid function is applied to the log probability ratio and log odds ratio, the scale of each ratio determines the amount

of discrepancy between the likelihood of the favored style and the disfavored style 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 4. 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 show this through the ablation study in Appendix B. Therefore, the odds ratio is a better choice when the preference alignment is done with SFT simultaneously due to the mild discrimination of disfavored responses and the prioritizing of the favored responses to be generated.

When comparing the log sigmoid loss with $\mathbf{PR}(X_2|X_1)$ to $\mathbf{OR}(X_2|X_1)$, In this context, it is essential to avoid an overly extreme contrast w. This caution is especially important given the sharp distribution of $\log \mathbf{PR}(X_2|X_1)$ depicted in Figure 4. The excessive discrepancy could lead to the unwarranted suppression of logits for tokens in disfavored responses within the incorporated setting, potentially resulting in issues of degeneration.

4.4 Gradients of ORPO

The gradient of $\mathcal{L}_{Ratio}$ further justifies the use of the odds ratio loss. It comprises two terms: one that penalizes the wrong predictions and one that contrasts between chosen and rejected responses, denoted in Equation 12[1] for $d = (x, y_l, y_w) \sim D$.

$$\nabla_\theta \mathcal{L}_{Ratio} = \delta(d) \cdot h(d) \tag{12}$$

$$\delta(d) = \left[1 + \frac{\mathbf{odds}_\theta P(y_w|x)}{\mathbf{odds}_\theta P(y_l|x)}\right]^{-1} \tag{13}$$

$$h(d) = \frac{\nabla_\theta \log P_\theta(y_w|x)}{1 - P_\theta(y_w|x)} - \frac{\nabla_\theta \log P_\theta(y_l|x)}{1 - P_\theta(y_l|x)} \tag{14}$$

When the odds of the favored responses is relatively higher than the disfavored responses, $\delta(d)$ in Equation 13 will converge to 0. This indicates that the $\delta(d)$ will play the role of a penalty term, which accelerates the parameter updates if the model is more likely to generate the rejected responses.

Meanwhile, $h(d)$ in Equation 14 implies a weighted contrast of the two gradients from the chosen and rejected responses. Specifically, $1-P(y|x)$ in denominators amplifies the gradients of the corresponding side of the likelihood $P(y|x)$ is low. For the chosen responses, this accelerates the model's adaptation toward the distribution of chosen responses as the likelihood increases.

[1]The full derivation for $\nabla_\theta \mathcal{L}_{Ratio}$ is in Appendix A.

5 Empirical Study

We study the effectiveness of ORPO through several experimental settings. First, we assess the general instruction-following abilities of the models comparing the preference alignment algorithms in Section 6.3. Second, we measure the win rate of OPT models trained with ORPO against other alignment methods training OPT 1.3B as a reward model in Section 6.2. We then perform further analyses to demonstrate the odds ratio increasing as intended while fine-tuning with ORPO in Section 6.1. And finally, we measure the lexical diversity of the models trained with ORPO and DPO in Section 6.4.

5.1 Training Configurations

Models We train OPT (Zhang et al., 2022) series with from 125M to 1.3B parameters using supervised fine-tuning (SFT), proximal policy optimization (PPO), direct policy optimization (DPO), and compare these to our ORPO. PPO and DPO models were fine-tuned with TRL library (von Werra et al., 2020) on top of SFT models trained for a single epoch on the chosen responses following Rafailov et al. (2023) and Tunstall et al. (2023), which we notate this by prepending "+" to each algorithm (e.g., +DPO). Additionally, we train Phi-2 (2.7B) (Javaheripi and Bubeck, 2023), a pre-trained language model with promising downstream performance (Beeching et al., 2023), as well as Llama-2 (7B) (Touvron et al., 2023) and Mistral (7B) (Jiang et al., 2023). Further training details for each method are in Appendix C.

Datasets We test each training method and model on two datasets: 1) Anthropic's HH-RLHF (Bai et al., 2022a), 2) Binarized UltraFeedback (Tunstall et al., 2023). We filtered out instances where $y_w = y_l$ or where $y_w = \emptyset$ or where $y_l = \emptyset$.

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 15 (Ziegler et al., 2020). OPT-350M reward model was used for PPO, and OPT-1.3B reward model was used to assess the generations of final models.

We refer to these models as RM-350M and RM-1.3B in Section 6.

$$-\mathbb{E}_{(x,y_l,y_w)}\left[\log \sigma\left(r(x,y_w) - r(x,y_l)\right)\right] \quad (15)$$

5.2 Leaderboard Evaluation

In Section 6.3, we perform evaluation using the AlpacaEval (Li et al., 2023b) benchmarks, comparing ORPO to other instruction-tuned models reported in the official leaderboard[2], including Vicuna (7B) (Chiang et al., 2023), Alpaca (7B) (Taori et al., 2023), Llama-2 Chat (7B) (Touvron et al., 2023), and Falcon-Instruct (40B) (Almazrouei et al., 2023). We report model performance using $\text{AlpacaEval}_{1.0}$ and $\text{AlpacaEval}_{2.0}$. 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[3] as an evaluator following the default setting. We assess if the generated responses are favored over the responses generated from GPT-4.

6 Results and Analysis

By starting with monitoring the empirical validity of the odds ratio penalty in Section 6.1, we evaluate if the models have precisely learned the preference through both reward model win rate with RM-1.3B exclusively fine-tuned on each dataset and general instruction-following evaluation with GPT-4in Sections 6.2 and 6.3. Then, we expand our analysis to the lexical diversity of the models in Section 6.4.

6.1 Log Odds Ratio during Training

We demonstrate that models trained with ORPO learned the preference throughout the training process. We monitored the log probabilities of the chosen and rejected responses, along with the log odds ratio. 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 role of supervised fine-tuning (SFT) while the penalty term L_{Ratio} induces the model to lower the likelihood of unwanted generations.

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

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

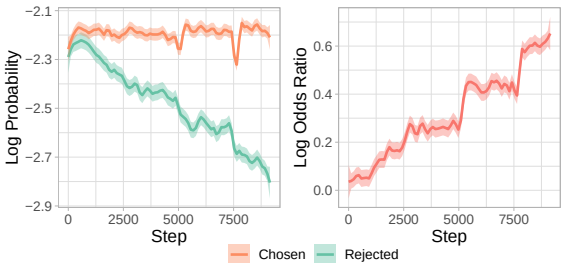

Figure 5: Average log-likelihood for chosen and rejected responses and log odds ratio per batch. The odds of disfavored style generations consistently increase during training with ORPO.

6.2 Reward Model Win Rate

We assess the win rate of ORPO over other preference alignment methods, including supervised fine-tuning (SFT), PPO, and DPO, using RM-1.3B fine-tuned for reward modeling to understand the effectiveness and scalability of ORPO in Tables 1 and 2. Additionally, we visually verify that ORPO can effectively enhance the expected reward in comparison to SFT in Figure 6.

HH-RLHF In Table 1, ORPO outperforms SFT and RLHF across all model scales. The highest win rate against SFT and RLHF across the size of the model was 78.0% and 79.4%, respectively. Meanwhile, the win rate over DPO was correlated to the size of the model with the largest model having the highest win rate: 70.9% in the OPT-1.3B model.

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 1: 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 2. ORPO was preferred over SFT and PPO for maximum 80.5% and 85.8%, respectively. While consistently preferring ORPO over SFT and RLHF, the win rate over DPO gradually increases as the size of the model increases. The scale-wise trend in exceeding DPO will be further shown through 2.7B models in Section 6.3.

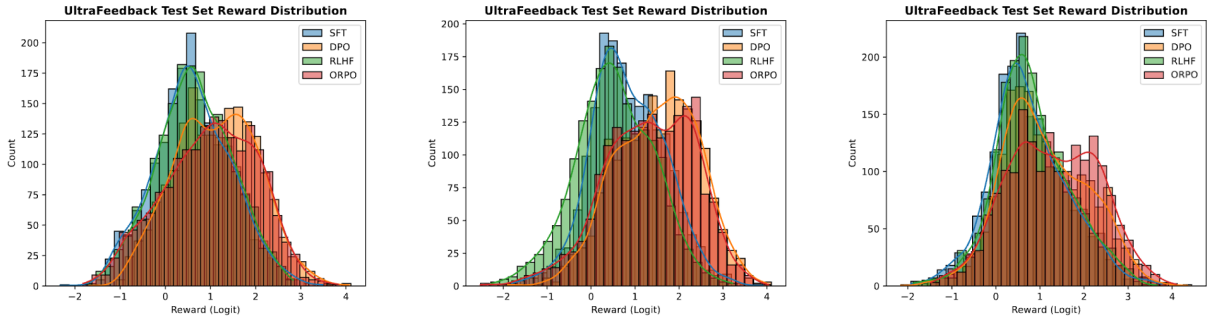

Figure 6: Reward distribution comparison between OPT-125M (left), OPT-350M (middle), and OPT-1.3B (right) trained with SFT (blue), RLHF (green), DPO (orange), and ORPO (red) on the test set of UltraFeedback using the RM-1.3B. While the rewards of the trained models are roughly normal and preference optimization 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. The same plot for HH-RLHF is in Appendix D.

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 2: Average win rate (%) and its standard deviation of ORPO and standard deviation over other methods on **UltraFeedback** dataset for three rounds. Same decoding strategy with Table 1.

	Size	$\text{AlpacaEval}_{1.0}$	$\text{AlpacaEval}_{2.0}$
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	**66.17% (1.67)**	**4.24% (0.61)**
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	71.34% (1.59)	8.35% (0.87)
Zephyr (β) *	7B	81.09% (1.38)	10.99% (0.96)
Mistral + ORPO *(Ours)*	7B	**87.94% (1.15)**	**12.20% (0.98)**

Table 3: 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 Phi-2-ORPO either overwhelms or is on par with the larger state-of-the-art models. (* indicates the results excerpted from the official leaderboard.)

Overall Reward Distribution In addition to the win rate, we compare the reward distribution of the responses generated with respect to the test set of the UltraFeedback dataset in Figure 6 and HH-RLHF dataset in Appendix D. Regarding the SFT reward distribution as a default, RLHF, DPO, and ORPO shift it in both datasets. However, the magnitude of reward shifts for each algorithm differs.

In Figure 6, RLHF 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 of RLHF (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 mostly located on the very right side of each subplot indicating higher expected rewards. Recalling the intent of preference alignment methods, the distributions in Figure 6 indicate that ORPO tends to fulfill the aim of preference alignment for all model sizes.

6.3 Instruction Following

Phi-2 (ORPO) In general, ORPO improved pre-trained Phi-2 to be a comparable instruction-following language model by *only using UltraFeedback* as the instruction-tuning dataset, as shown in Table 3. Within the same model family, ORPO is preferred over other training methods for OPT and Phi-2. It is notable that Phi-2 (ORPO) exceeds Alpaca and Vicuna with the win rate of 66.17%, which are 7B instruction-following models with the win rates of 26.46% and 64.41%.

Meanwhile, in $\text{AlpacaEval}_{2.0}$, Phi-2 (ORPO) was preferred for 4.24% with 2.7B parameters. It was on par with the Llama-2 Chat (7B) model, which is one of the state-of-the-art chat models trained with RLHF in the 7B scale.

Llama-2 (ORPO) Notably, instruction tuning with only UltraFeedback and ORPO on Llama-2-7B resulted in higher AlpacaEval scores than the -chat and version for both 7B and 13B scale, eventually showing 81.26% and 9.44% in two AlpacaEval.

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 implies the efficiency of our method, in which the model can rapidly learn both the desired domain and the preference with a limited amount of data. This aligns with the examination of $h(d)$ in the gradient of our method studied in Section 4.4.

Mistral (ORPO) Finally, fine-tuning Mistral (7B) with only 32,000 instances UltraFeedback and ORPO for 5 hours with the setting in Appendix C outperforms Zephyr series, which are the Mistral (7B) models fine-tuned with SFT on 20K UltraChat (Ding et al., 2023) and DPO on the full UltraFeedback. As shown in Table 3, Mistral (ORPO) achieves 87.94% and 12.20%, which exceeds Zephyr β by 6.85% and 1.21%. It is noteworthy that Zephyr was fine-tuned to UltraChat in the first place before DPO, and Misral (ORPO) was trained directly with Ultrafeedback only.

6.4 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 the Gemini-Pro (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 Equation 17.

$$\mathcal{O}^i_\theta := \{y_j \sim \theta(y|x_i) | j = 1, 2, ..., K\} \tag{16}$$

$$D(\mathcal{O}^i_\theta) = \frac{1}{2} \cdot \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \cos(h_i, h_j)}{N \cdot (N-1)} \tag{17}$$

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 4.

Per Input Diversity (PID) We average the input-wise average cosine similarity between the generated samples with Equation 18 to assess the per-input diversity. In Table 4, ORPO checkpoints 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.

$$\text{PID}_D(\theta) = \frac{1}{N} \sum_{i=1}^{N} D(\mathcal{O}^i_\theta) \tag{18}$$

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 19 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 4. We can infer that ORPO triggers the model to generate more instruction-specific responses in comparison to DPO.

$$\text{AID}_D(\theta) = D\left(\bigcup_{i=1}^{N} \mathcal{O}^{i}{}_{,\theta,j=1}\right) \tag{19}$$

	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 4: 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 is bolded.

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 size of the model 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, Llama-2 (ORPO) and Mistral (ORPO) achieved 81.26% and 87.94% in $\text{AlpacaEval}_{1.0}$, 9.44% and 12.20% in $\text{AlpacaEval}_{2.0}$, thereby underscoring the efficiency and effectiveness of ORPO. We release code to aid reproducability (see supplementary material).

Limitations

While conducting a comprehensive analysis of the diverse preference alignment methods, including DPO and RLHF, we did not incorporate a wider range of preference alignment algorithms. We leave the wider 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.

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.

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.

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.

Weixin Chen and Bo Li. 2024. Grath: Gradual self-truthifying for large language models.

Qinyuan Cheng, Tianxiang Sun, Xiangyang Liu, Wenwei Zhang, Zhangyue Yin, Shimin Li, Linyang Li, Kai Chen, and Xipeng Qiu. 2024. Can ai assistants know what they don't know?

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

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

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. *9th International Conference on Learning Representations, ICLR*.

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.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition.

Kawin Ethayarajh, Winnie Xu, Dan Jurafsky, and Douwe Kiela. 2023. Human-centered loss functions (halos). Technical report, Contextual AI.

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

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.

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

Alexey Gorbatovski and Sergey Kovalchuk. 2024. Reinforcement learning for question answering in programming domain using public community scoring as a human feedback.

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

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.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback.

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

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.

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

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.

Jing-Cheng Pang, Pengyuan Wang, Kaiyuan Li, Xiong-Hui Chen, Jiacheng Xu, Zongzhang Zhang, and Yang Yu. 2023. Language model self-improvement by reinforcement learning contemplation.

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.

Miguel Moura Ramos, Patrick Fernandes, António Farinhas, and André F. T. Martins. 2023. Aligning neural machine translation models: Human feedback in training and inference.

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.

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`.

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*.

Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D. Manning, and Chelsea Finn. 2023. Fine-tuning language models for factuality.

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.

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

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.

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.

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 $\nabla_\theta \mathcal{L}_{Ratio}$ with Odds Ratio

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

$$\nabla_\theta \mathcal{L}_{Ratio} = \nabla_\theta \log \sigma \left(\log \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \right) \tag{20}$$

$$= \frac{\sigma' \left(\log g(x, y_l, y_w) \right)}{\sigma \left(\log g(x, y_l, y_w) \right)} \tag{21}$$

$$= \sigma \left(-\log g(x, y_l, y_w) \right) \cdot \nabla_\theta \log g(x, y_l, y_w) \tag{22}$$

$$= \frac{\sigma \left(-\log g(x, y_l, y_w) \right)}{g(x, y_l, y_w)} \cdot \nabla_\theta g(x, y_l, y_w) \tag{23}$$

$$= \sigma \left(-\log g(x, y_l, y_w) \right) \cdot \nabla_\theta \log g(x, y_l, y_w) \tag{24}$$

$$= \left(1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \right)^{-1} \cdot \nabla_\theta \log \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \tag{25}$$

In Equation 25, 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.

$$\nabla_\theta \log \left(1 - P_\theta(y|x) \right) = \frac{\nabla_\theta \left(1 - P_\theta(y|x) \right)}{1 - P_\theta(y|x)} \tag{26}$$

$$= \frac{-\nabla_\theta P_\theta(y|x)}{1 - P_\theta(y|x)} \tag{27}$$

$$= -\frac{P_\theta(y|x)}{1 - P_\theta(y|x)} \cdot \nabla_\theta \log P_\theta(y|x) \tag{28}$$

$$= \textbf{odds}_\theta(y|x) \cdot \nabla_\theta \log P_\theta(y|x) \tag{29}$$

$$\nabla_\theta \log \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} = \nabla_\theta \log \frac{P_\theta(y_w|x)}{P_\theta(y_l|x)} - \Big(\nabla_\theta \log(1 - P_\theta(y_w|x)) - \nabla_\theta \log(1 - P_\theta(y_l|x)) \Big) \tag{30}$$

$$= \left(1 + \textbf{odds}_\theta P(y_w|x) \right) \nabla_\theta \log P_\theta(y_w|x) - \left(1 + \textbf{odds}_\theta P(y_l|x) \right) \nabla_\theta \log P_\theta(y_l|x) \tag{31}$$

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

$$\nabla_\theta \mathcal{L}_{Ratio} = \frac{1 + \textbf{odds}_\theta P(y_w|x)}{1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)}} \cdot \nabla_\theta \log P_\theta(y_w|x) - \frac{1 + \textbf{odds}_\theta P(y_l|x)}{1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)}} \cdot \nabla_\theta \log P_\theta(y_l|x) \tag{32}$$

$$= \left(1 + \frac{\textbf{odds}_\theta P(y_w|x)}{\textbf{odds}_\theta P(y_l|x)} \right)^{-1} \cdot \left(\frac{\nabla_\theta \log P_\theta(y_w|x)}{1 - P(y_w|x)} - \frac{\nabla_\theta \log P_\theta(y_l|x)}{1 - P(y_l|x)} \right) \tag{33}$$

B Ablation on Probability Ratio and Odds Ratio

In this section, we continue the discussion in Section 4.3 through empirical results comparing the log probabilities of chosen and rejected responses in UltraFeedback when trained with probability ratio and odds ratio. Recalling the sensitivity of each ratio discussed in Section 4.3, 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 7, 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).

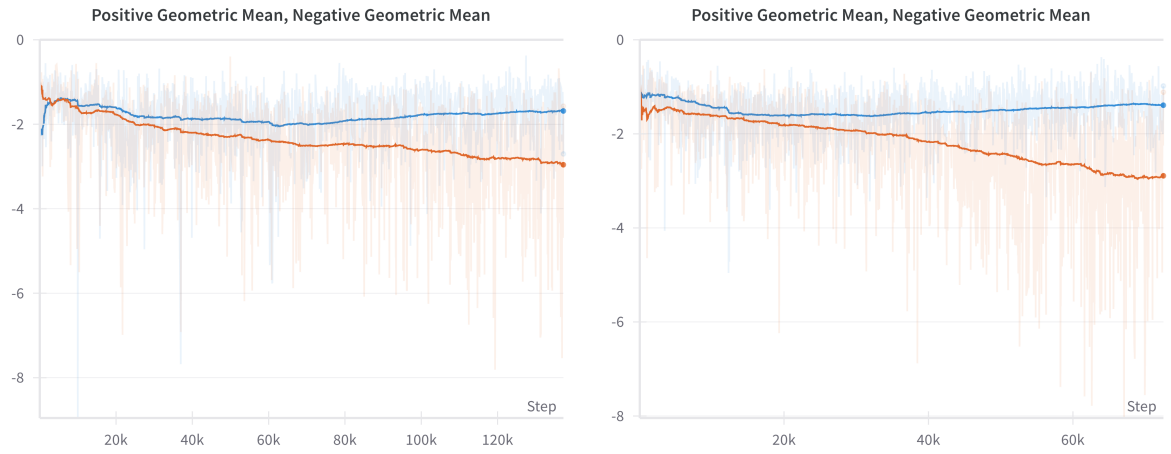

Figure 7: The log probability trace when the model is trained with the probability ratio (left) and the odds ratio (right) given the same hyperparameters. The probability ratio leads the rejected responses to have relatively lower log probabilities in a manner.

C Experimental Details

The OPT series and Phi-2 were trained by applying Flash-Attention 2 (Dao, 2023) and DeepSpeed ZeRO 2 (Rasley et al., 2020) for computational efficiency, and Llama-2 models 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, 8-bit AdamW optimizer (Dettmers et al., 2022) was 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 either to the conversation history or the complex instruction, we filtered instances that have 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 5 for UltraFeedback. For HH-RLHF dataset, the `output_min_length` and `output_max_length` was 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 3 epochs to select the best model by evaluation loss in each epoch. But in most cases, the first or the second checkpoint was selected as the best model as the evaluation loss got higher from the third epoch.

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 by the lowest evaluation loss.

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 5: Hyperparameter settings for RLHF.

D Test Set Reward Distribution on HH-RLHF

Along with Figure 8, which depicts the reward distribution of OPT2-125M, OPT2-350M, and OPT2-1.3B on the UltraFeedback dataset, we report the reward distribution of each pre-trained checkpoint trained on the HH-RLHF dataset. As discussed in Section 6.2, ORPO is consistently pushing the reward distribution of SFT to the right side.

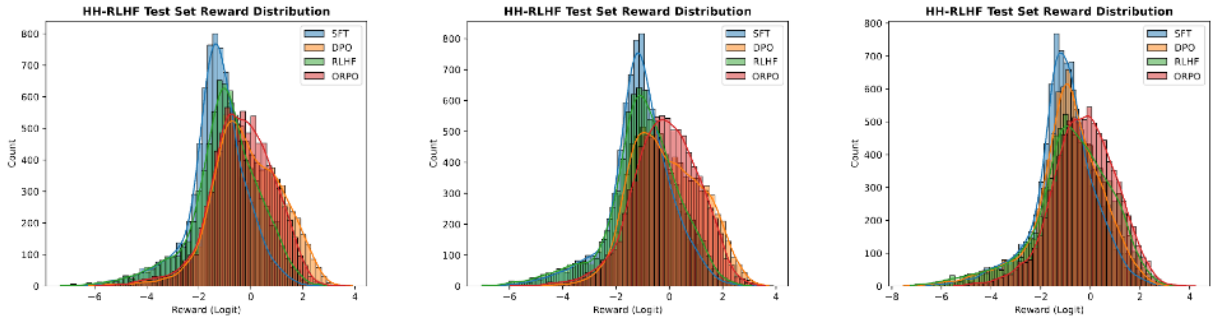

Figure 8: Reward distribution comparison between OPT-125M (left), OPT-350M (middle), and OPT-1.3B (right) trained with SFT (blue), RLHF (green), DPO (orange), and ORPO (red) on the test set of HH-RLHF using the 1.3B reward model. General tendency follows that of Figure 6.

E Special Instructions for Verbosity Assessment

#	Succinctness	Verboseness
1	Please generate a short and concise response.	Please generate an elaborative and chatty response.
2	Provide a brief and concise answer.	Provide a detailed answer.
3	Keep your reply short and to the point.	Keep your reply elaborative and intricate.
4	Keep your answer brief for clarity.	Keep your answer detailed.
5	Generate a brief and to-the-point answer.	Generate a chatty and step-wise answer.

Table 6: Instructions prepended to the queries from AlpacaEval. Each instruction set asks the model to generate either shorter or longer responses given the query, respectively.

For the succinctness and verboseness instructions, we generated 5 different instructions each with ChatGPT [4]. From the instructions in Table 6, we randomly sampled one prompt each for every batch to prevent potential word bias.

[4] https://chat.openai.com/