2D-DPO: Scaling Direct Preference Optimization with 2-Dimensional Supervision

Shilong Li*†, Yancheng He*, Hui Huang*, Xingyuan Bu*‡, Jiaheng Liu,
Hangyu Guo, Weixun Wang, Jihao Gu, Wenbo Su, Bo Zheng
Taobao & Tmall Group of Alibaba
`lishilong.aaron@gmail.com, xingyuanbu@gmail.com`

Abstract

Recent advancements in Direct Preference Optimization (DPO) have significantly enhanced the alignment of Large Language Models (LLMs) with human preferences, owing to its simplicity and effectiveness. However, existing methods typically optimize a scalar score or ranking reward, thereby overlooking the multi-dimensional nature of human preferences. In this work, we propose to extend the preference of DPO to two dimensions: **segments** and **aspects**. We first introduce a 2D supervision dataset called **HelpSteer-2D**. For the segment dimension, we divide the response into sentences and assign scores to each segment. For the aspect dimension, we meticulously design several criteria covering the response quality rubrics. With the 2-dimensional signals as feedback, we develop a **2D-DPO** framework, decomposing the overall objective into multi-segment and multi-aspect objectives. Extensive experiments on popular benchmarks demonstrate that 2D-DPO performs better than methods that optimize for scalar or 1-dimensional preferences[1].

1 Introduction

Recent advancements in Large Language Models (LLMs) have shown impressive performance across a wide range of tasks (Zhao et al., 2023; Bai et al., 2024; Wu et al., 2024a; Li et al., 2024a). A pivotal component in LLM training is Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Bai et al., 2022), which aligns LLMs with human preferences. However, due to its complexity, traditional RLHF often leads to challenges such as training instability and reward collapse (Wolf et al., 2023; Song et al., 2023).

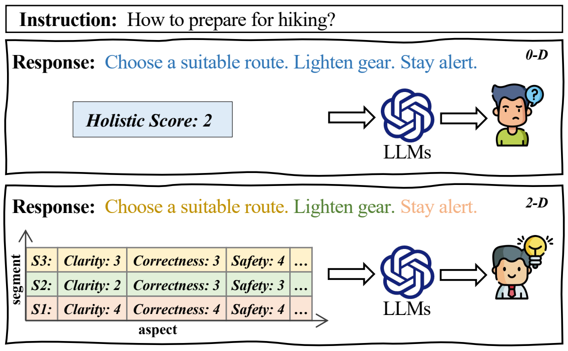

Instruction: How to prepare for hiking?
Response: Choose a suitable route. Lighten gear. Stay alert.
0-D
Holistic Score: 2
LLMs
Response: Choose a suitable route. Lighten gear. Stay alert.
2-D
segment
S3: Clarity: 3 Correctness: 3 Safety: 4 ...
S2: Clarity: 2 Correctness: 3 Safety: 3 ...
S1: Clarity: 4 Correctness: 4 Safety: 4 ...
aspect
LLMs

Figure 1: An illustrative comparison between vanilla DPO and 2D-DPO.

Direct Preference Optimization (DPO) (Rafailov et al., 2023), as a simpler and more effective alternative, has gained considerable attention due to its ability to bypass the need for explicitly fitting a reward model (Meng et al., 2024; Ethayarajh et al., 2024). However, most existing DPO-style approaches rely on scalar scores or rankings and ignore the multi-dimensional nature of human preferences, resulting in inefficient and imprecise optimization. For instance, a response may be deemed satisfactory under one aspect such as `correctness`, but falls short in another such as `clarity`. Moreover, not all segments of a response should be treated uniformly; even in a preferred response, there may be segments of inferior quality. This underscores the need for a more nuanced approach that recognizes the multi-dimensionality of feedback and its critical impact on model training.

In response, some recent works have attempted to leverage signals that are believed to reflect the importance of individual segments as reward scores (Zeng et al., 2024; Chan et al., 2024; Jiang et al., 2024; Chen et al., 2024). However, these signals are often derived from statistical features such as edit distance or confidence estimation, which can introduce noise and lack interpretability. Other approaches incorporate multi-objective optimiza-

---

* Equal contribution. ‡ Corresponding Author.

† Work done during an internship at Taobao & Tmall Group of Alibaba.

[1] Code and datasets are available at `https://github.com/longls777/2D-DPO`.

Findings of the Association for Computational Linguistics:
NAACL 2025, pages 8164–8188
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

tion to balance different aspects of human preferences (Wu et al., 2024b; Guo et al., 2023; Cao et al., 2024). However, these methods mostly rely on Proximal Policy Optimization (PPO) (Schulman et al., 2017), which is prone to instability during training. Furthermore, these efforts only extend preference optimization from 0-dimensional (scalar reward) to 1-dimensional (aspect/segment) supervision, which remains insufficient for capturing the complexity of real-world human preferences.

To better address the intricacy of human preferences, we propose **2D-DPO**, a novel direct alignment strategy that enables 2-dimensional (2D) fine-grained optimization. Our core idea is to scale supervision signals across two dimensions: **segments** and **aspects**. To this end, we first construct a preference dataset called HelpSteer-2D, where each sample is annotated with a 2-dimensional score matrix evaluating each segment across multiple aspects. These signals are derived from a robust model guided by a set of stringent principles, ensuring the generation of highly accurate and interpretable supervision signals. Building on this, we propose a novel approach to achieve 2-dimensional direct preference alignment. Experimental results on three public benchmarks demonstrate that 2D-DPO significantly outperforms previous methods. In summary, our main contributions are threefold:

- We introduce a novel 2-dimensional preference alignment method, 2D-DPO, which scales supervision signals across both segments and aspects to better align with human preferences.
- We develop a high-quality, fine-grained preference dataset, HelpSteer-2D, which will be released to the community for future research.
- Extensive experiments show that 2D-DPO delivers superior performance in aligning with human preferences compared to prior approaches.

2 Related Work

2.1 Preference Optimization

Large language models (LLMs) have advanced rapidly, with reinforcement learning from human feedback (RLHF) commonly used to align LLMs with human preferences (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Bai et al., 2022; Liu et al., 2024b,a; Peng et al., 2023; Feng et al., 2022). However, traditional RLHF methods face challenges like instability and high resource demands (Wolf et al., 2023; Song et al., 2023), prompting the search for simpler alternatives. One such representative approach is Direct Preference Optimization (DPO) (Rafailov et al., 2023), which optimizes alignment without explicit reward modeling, offering simplicity and stability. Building on DPO, IPO (Azar et al., 2024) adds a regularization term to alleviate overfitting. KTO (Ethayarajh et al., 2024) only requires a binary signal of whether an output is desirable or undesirable for an input to align LLMs, simplifying the data acquisition process. ORPO (Hong et al., 2024) simplifies training with odds ratio-based penalties, and SimPO (Meng et al., 2024) improves efficiency by using average log probability as an implicit reward.

2.2 Token-level Preference Optimization

The response-level rewards in naive PPO and DPO often lack token-level details. To address this, researchers have explored fine-grained supervision signals in three ways: (1) Human annotation: Methods like PRM (Lightman et al., 2023) and FGRLHF (Wu et al., 2024b) involve human annotators labeling each segment of the response to generate fine-grained signals. (2) LLM annotation: To reduce the cost of human labeling, stronger LLMs are used to generate preference pairs with minimal edits (Guo et al., 2023; Chen et al., 2024; Yoon et al., 2024; Jiang et al., 2024) or to identify positive and negative response segments (Cao et al., 2024). (3) Internal signal: Some works use the model's internal information as reward signals, such as using attention scores for token rewards in ABC (Chan et al., 2024) or decomposing DPO's response-level rewards into token-level signals in SePO (Yang et al., 2024b; Rafailov et al., 2024). TDPO (Zeng et al., 2024) achieves token-level alignment by controlling the KL divergence for each token.

2.3 Multi-objective Preference Optimization

Human preferences are often complex, diverse, and even contradictory, making single-dimensional training insufficient. To address this, some studies align LLMs with multiple objectives by either training separate reward models and averaging their outputs (Pan et al., 2023; Ji et al., 2024; Rame et al., 2024; de Langis et al., 2024; Wang et al., 2024a). However, this approach demands significant computational resources. In contrast, MODPO (Zhou et al., 2024) offers a simpler, reinforcement learning-free method for optimizing multiple objectives. RiC (Yang et al., 2024c) and

CPO (Guo et al., 2024) focus on integrating multiple reward values for controllable generation.

3 Approach

In this section, we provide a detailed introduction to our 2D-DPO approach for aligning LLMs. We first describe the construction of the 2D preference dataset in Section 3.1. Then, we present an enhanced 2D-DPO formulation that integrates the 2D supervision signals into the alignment process in Section 3.2. The complete pipeline of our method is illustrated in Figure 2.

3.1 Preference Dataset with 2D Signal

In general, a preference optimization dataset, denoted as $\mathcal{D} = \{x^i, y_w^i, y_l^i\}_{i=1}^N$, comprises prompts x along with a chosen response y_w and a rejected response y_l, where y_w is of higher quality compared to y_l. Such datasets are commonly used to train reward models (*e.g.*, PPO) or directly for model alignment (*e.g.*, DPO). However, differentiating between the chosen and rejected responses based on a scalar score is often coarse and imprecise. The quality of responses can vary significantly across different evaluation aspects, and even a chosen response might contain segments of low quality, while a rejected response could include segments of high quality. Therefore, relying solely on a scalar score for optimization may restrict the model's ability to effectively align with human preferences.

To address this issue, we propose a fine-grained scoring approach that decomposes the scalar scores of model responses to segment-level and aspect-level. The first step is to divide the response into segments, and the choice of segment length is crucial for ensuring the effectiveness of the fine-grained optimization. Segments that are too long cannot resolve the aforementioned coarse scoring issues, while segments that are too short pose difficulties for accurate assessment. Therefore, we choose the sentence as the scoring unit, which can strike a balance between scoring accuracy and the clarity of segment preferences.

After segmenting responses based on typical sentence-ending punctuations, we employ GPT-4 to perform aspect-level scoring. Following HelpSteer2 (Wang et al., 2024b), we annotate the preference data across five key aspects: `Helpfulness`, `Correctness`, `Safety`, `Completeness`, `Clarity`. The first three aspects are independent of different sentences. The aspect of `Completeness` generally increases as responses become more comprehensive, while `Clarity` tends to decrease as responses grow longer and more redundant. To ensure the integrity of our annotations, we use separate prompts for each aspect to prevent any cross-influence among them. For the details of the annotation process, please refer to Appendix F.

Finally, the constructed dataset is as follows:

$$\begin{aligned}
&\mathcal{D} = \{x^i, \boldsymbol{y}_w^i, \boldsymbol{y}_l^i\}_{i=1}^N \\
&\boldsymbol{y}_w^i = \{y_{w,k}^i, \{r_{w,k,j}^i\}_{j=1}^A\}_{k=1}^{S_w^i}, \\
&\boldsymbol{y}_l^i = \{y_{l,k}^i, \{r_{l,k,j}^i\}_{j=1}^A\}_{k=1}^{S_l^i}.
\end{aligned} \tag{1}$$

In the dataset $\mathcal{D}$, x^i represents the i-th prompt, while $\boldsymbol{y}_w^i$ and $\boldsymbol{y}_l^i$ represent the 2D annotations for the chosen and rejected response, respectively. Each 2D annotation includes S^i text segments, indexed by k, denoted as y_k^i. The scores for these segments are denoted by $r_{k,j}^i$, where j indicates the index across the aforementioned five aspects.

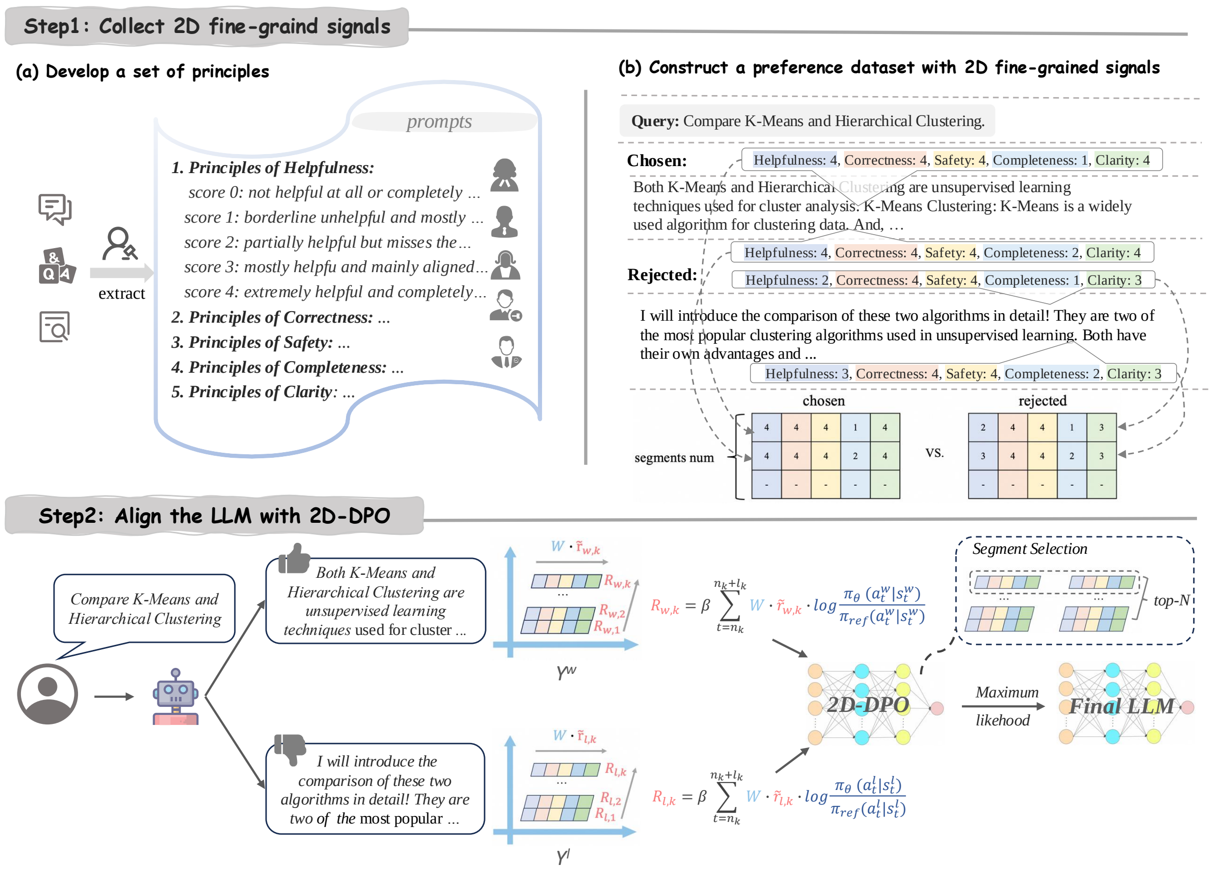
3.2 2D-DPO

While the construction process of 2D signals is straightforward, integrating them effectively into the alignment process presents significant challenges. Previous approaches mostly utilize these signals as a scalar reward by weighted summation, which is insufficient for enabling the model to distinguish between varying quality across different dimensions. To address this issue, we propose a novel alignment method called 2D-DPO.

Vanilla DPO. Direct Preference Optimization (DPO) (Rafailov et al., 2023), as one of the most popular alignment methods, proposes a direct optimization objective that satisfies the optimal preference policy without using a reward model:

$$\begin{aligned}
&\mathcal{L}_{DPO}(\pi_\theta; \pi_{ref}) = \\
&-\mathbb{E}_{(x,y^w,y^l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w \mid x)}{\pi_{ref}(y_w \mid x)}\right.\right. \\
&\left.\left.-\beta\log\frac{\pi_\theta(y_l \mid x)}{\pi_{ref}(y_l \mid x)}\right)\right],
\end{aligned} \tag{2}$$

where π_θ and π_{ref} represent the policy model and the reference model, respectively. DPO can fundamentally be viewed as a multi-armed bandit problem, where the model's entire response is treated as a single arm. According to Rafailov et al. 2023, in the token-level Markov Decision Process (MDP), the language model's Bradley-Terry preference


Step1: Collect 2D fine-graind signals
(a) Develop a set of principles
prompts
extract
1. Principles of Helpfulness:
score 0: not helpful at all or completely …
score 1: borderline unhelpful and mostly …
score 2: partially helpful but misses the…
score 3: mostly helpfu and mainly aligned…
score 4: extremely helpful and completely…
2. Principles of Correctness: ...
3. Principles of Safety: ...
4. Principles of Completeness: ...
5. Principles of Clarity: ...
(b) Construct a preference dataset with 2D fine-grained signals
Chosen:
Helpfulness: 4, Correctness: 4, Safety: 4, Completeness: 1, Clarity: 4
Helpfulness: 4, Correctness: 4, Safety: 4, Completeness: 2, Clarity: 4
Rejected:
Helpfulness: 2, Correctness: 4, Safety: 4, Completeness: 1, Clarity: 3
Helpfulness: 3, Correctness: 4, Safety: 4, Completeness: 2, Clarity: 3
chosen
rejected
segments num
vs.
Step2: Align the LLM with 2D-DPO
Segment Selection
top-N
$W \cdot \tilde{r}_{w,k}$
$R_{w,k}$
$R_{w,2}$
$R_{w,1}$
Y^w
$W \cdot \tilde{r}_{l,k}$
$R_{l,k}$
$R_{l,2}$
$R_{l,1}$
Y^l
$R_{w,k} = \beta \sum_{t=n_k}^{n_k+l_k} W \cdot \tilde{r}_{w,k} \cdot log \frac{\pi_\theta(a_t^w|s_t^w)}{\pi_{ref}(a_t^w|s_t^w)}$
$R_{l,k} = \beta \sum_{t=n_k}^{n_k+l_k} W \cdot \tilde{r}_{l,k} \cdot log \frac{\pi_\theta(a_t^l|s_t^l)}{\pi_{ref}(a_t^l|s_t^l)}$
2D-DPO
Maximum likehood
Final LLM

Query: Compare K-Means and Hierarchical Clustering.

Both K-Means and Hierarchical Clustering are unsupervised learning techniques used for cluster analysis. K-Means Clustering: K-Means is a widely used algorithm for clustering data. And, …

I will introduce the comparison of these two algorithms in detail! They are two of the most popular clustering algorithms used in unsupervised learning. Both have their own advantages and ...

4	4	4	1	4
4	4	4	2	4
-	-	-	-	-

2	4	4	1	3
3	4	4	2	3
-	-	-	-	-

Compare K-Means and Hierarchical Clustering

Both K-Means and Hierarchical Clustering are unsupervised learning techniques used for cluster ...

I will introduce the comparison of these two algorithms in detail! They are two of the most popular ...

Figure 2: Illustration of our proposed 2D-DPO. Firstly, we develop principles for preference annotation on different aspects, and collect scores across different segments and aspects for pairwise responses, leading to 2-dimensional signals. Secondly, we apply 2D-DPO on the constructed signals with decomposed training objective.

model can be expressed in the following form:

$$p^*(\tau^w \succeq \tau^l) = \frac{\exp\left(\sum_{i=1}^{N} r(\mathbf{s}_i^w, \mathbf{a}_i^w)\right)}{\exp\left(\sum_{i=1}^{N} r(\mathbf{s}_i^w, \mathbf{a}_i^w)\right) + \exp\left(\sum_{i=1}^{M} r(\mathbf{s}_i^l, \mathbf{a}_i^l)\right)}, \tag{3}$$

where τ^w and τ^l represent the winning and losing trajectories, respectively. In this context, $\mathbf{a}$ represents the next generated token, and $\mathbf{s}$ denotes the current state, consisting of the prompt along with all previously generated tokens.

Subsequently, based on the fixed point solution under the general maximum entropy RL setting (Ziebart, 2010; Levine, 2018), Rafailov et al. derived the form of DPO in token-level MDP:

$$\begin{aligned}\mathcal{L}_{DPO}(\pi_\theta; \pi_{ref}) = \\ -\mathbb{E}\Bigg[\log\sigma\Bigg(\beta\sum_{t=0}^{N-1}\log\frac{\pi_\theta(a_w^t \mid s_w^t)}{\pi_{ref}(a_w^t \mid s_w^t)} \\ -\beta\sum_{t=0}^{M-1}\log\frac{\pi_\theta(a_l^t \mid s_l^t)}{\pi_{ref}(a_l^t \mid s_l^t)}\Bigg)\Bigg].\end{aligned} \tag{4}$$

In the above process, Rafailov et al. combined Ng et al.'s definition of equivalence between two reward functions through a potential function and concluded that $\beta \log \frac{\pi_\theta^*(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)}$ and $r(\mathbf{s}_t, \mathbf{a}_t)$ can equivalently yield the same optimal policy. Furthermore, $\beta \log \frac{\pi_\theta^*(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)}$ is precisely the optimal advantage function $A^*(\mathbf{s}_t, \mathbf{a}_t)$. For a detailed derivation, please refer to Appendix A.2 and A.3.

2D-DPO. With the above conclusions, 2D supervision signals can be conveniently integrated into the alignment process. We achieve the integration by using the signals to calibrate the token-level advantage function $A(\mathbf{s}_t, \mathbf{a}_t)$ for different segments. This approach equips the model with a direct perception of fine-grained preferences, avoiding the ambiguity of holistic rewards.

Specifically, we use the regularized fine-grained reward r as a coefficient, incorporating it into the token-level advantage function to obtain $\beta r \log \frac{\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)}$, which integrates the fine-grained signals. In practice, this is equivalent to adjusting β within the original $A(\mathbf{s}_t, \mathbf{a}_t)$. We provide complete proof of its feasibility in Appendix A.4.

The token-level DPO incorporating fine-grained signals is formalized as follows:

$$\begin{aligned}&\mathcal{L}(\pi_\theta, D) = \\ &-\mathbb{E}_{(\tau_w,\tau_l)\sim D}\log\sigma(\beta\sum_{k=0}^{S_w-1}\sum_{t=n_k}^{n_k+l_k} r_{w,k}\log\frac{\pi_\theta(\mathbf{a}_t^w|\mathbf{s}_t^w)}{\pi_{ref}(\mathbf{a}_t^w|\mathbf{s}_t^w)} \\ &-\beta\sum_{k=0}^{S_l-1}\sum_{t=n_k}^{n_k+l_k} r_{l,k}\log\frac{\pi_\theta(\mathbf{a}_t^l|\mathbf{s}_t^l)}{\pi_{ref}(\mathbf{a}_t^l|\mathbf{s}_t^l)}),\end{aligned} \tag{5}$$

where n_k represents the first token of the k-th segment and l_k denotes the length of the k-th segment.

For handling multiple aspects, we use a classic weighted approach for integration, that is, $r_{w,k} = \mathbf{W}\tilde{\mathbf{r}}_{w,k}$, where $\mathbf{W}$ represents the weights that sum up to 1, which reflect the importance of each aspect during the alignment process, and $\tilde{\mathbf{r}}_{w,k} = \{r_{w,k,j}\}_{j=1}^{A}$.

Segment Selection. The number of segments in the chosen and rejected responses may differ significantly, and typically only the segments with an impact on response preference need attention. Therefore, we select the top-N highest-scoring segments from the chosen response and the top-N lowest-scoring segments from the rejected response, where $N = \min(S_w, S_l)$, further enhances the efficiency of model alignment training. Additionally, we group segments in pairs to provide clearer contrast during alignment, making it easier for the model to learn fine-grained differences between the chosen and rejected responses. These segments are paired to form N BT models. The feasibility of this rearrangement is based on the fact that the loss for a single-segment BT model can be treated as setting the β_t of other parts to 0, as demonstrated in Appendix A.4. Thus, we obtain the token-level DPO formula incorporating fine-grained signals:

$$\begin{aligned}&\mathcal{L}_{group}(\pi_\theta, D) = \\ &-\mathbb{E}_{(\tau_w,\tau_l)\sim D}\left[\sum_{k=0}^{N-1}\log\sigma\left(\beta\sum_{t=n_k}^{n_k+l_k} r_{w,k}\log\frac{\pi_\theta(\mathbf{a}_t^w|\mathbf{s}_t^w)}{\pi_{ref}(\mathbf{a}_t^w|\mathbf{s}_t^w)}\right.\right. \\ &\left.\left.-\beta\sum_{t=n_k}^{n_k+l_k} r_{l,k}\log\frac{\pi_\theta(\mathbf{a}_t^l|\mathbf{s}_t^l)}{\pi_{ref}(\mathbf{a}_t^l|\mathbf{s}_t^l)}\right)\right].\end{aligned} \tag{6}$$

As a result, we've formulated the definitive objective of 2D-DPO. This training objective allows for the direct integration of 2-D supervision signals into the alignment process, enabling LLMs to discern the different aspects lying in different segments in the responses, thereby promoting better alignment with human preferences.

4 Experiments

4.1 Set-up

Benchmark. Our method has been tested on three widely recognized instruction-following benchmarks: Arena-Hard (Li et al., 2024b), AlpacaEval 2.0 (Dubois et al., 2024), and MT-Bench (Zheng et al., 2023). Each benchmark comprises a diverse set of queries, and the answers are evaluated under the framework of LLM-as-a-Judge (Zheng et al., 2023). We use `gpt-4-turbo-2024-04-09`[2] as the judge model and the scores are reported following each benchmark's protocol.

Model. Our method is validated on two models, Qwen2-7B-Instruct (Yang et al., 2024a) and Llama-3-8B-Instruct (AI@Meta, 2024). It deserves to be noticed that both models have undergone extensive instruction-tuning processes, therefore we directly perform preference optimization.

Baseline. We mainly compare our method with 0-dimensional preference optimization methods:

- DPO (Rafailov et al., 2023). This method leverages a mapping between reward functions and optimal policies to optimize the preference with a single stage of policy training.
- IPO (Azar et al., 2024). This method proposes a theoretically grounded approach method to replace pairwise preferences in DPO with pointwise rewards.
- KTO (Ethayarajh et al., 2024). This method proposes to maximize the utility of generations from non-paired data directly.
- ORPO (Hong et al., 2024). This method leverages a reference model-free monolithic odds ratio for contrasting favored and disfavored styles during the SFT stage.
- SimPO (Meng et al., 2024). This method proposes to use the average log probability of a sequence as the implicit reward, which eliminates the need for a reference model.

We also compare our method with the following 1-dimensional preference optimization method:

---

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

Policy Model	Methods	Arena-Hard			AlpacaEval 2.0			MT-Bench		
		WR (%)	**Avg. Len**	**95% CI**	**LC (%)**	**WR (%)**	**Avg. Len**	**Turn 1**	**Turn 2**	**Avg. Score**
Qwen2-7B -Instruct	Base	25.10	583	(-2.1, 2.0)	30.68	28.32	1862	8.01	6.61	7.31
	+ DPO	29.40	578	(-1.8, 1.9)	29.07	26.83	1996	8.11	6.45	7.28
	+ IPO	26.50	556	(-2.2, 2.2)	28.70	26.58	1940	7.90	6.53	7.21
	+ KTO	26.10	518	(-2.1, 2.5)	26.46	23.00	1730	8.11	6.40	7.26
	+ ORPO	25.40	573	(-2.2, 1.8)	28.58	27.70	1936	8.09	6.52	7.31
	+ SimPO	29.00	539	(-1.9, 2.4)	29.94	27.70	1904	8.06	6.50	7.28
	+ TDPO	25.90	564	(-2.0, 2.4)	29.81	27.33	1896	8.05	6.46	7.26
	+ 1D-DPO	29.80	574	(-2.4, 2.2)	31.07	28.70	1951	8.13	6.48	7.31
	+ 2D-DPO	**30.30**	586	(-1.9, 2.4)	**31.51**	28.94	1994	**8.18**	6.68	**7.43**
Llama3-8B -Instruct	Base	25.40	599	(-2.2, 2.4)	27.08	26.96	1959	7.66	6.84	7.25
	+ DPO	25.90	567	(-1.9, 2.2)	31.68	30.31	1883	7.64	6.60	7.20
	+ IPO	24.80	548	(-2.2, 1.8)	29.69	28.57	1891	7.73	6.75	7.24
	+ KTO	25.20	507	(-1.9, 2.4)	27.95	27.08	1835	7.65	6.65	7.15
	+ ORPO	25.60	537	(-1.7, 1.9)	29.19	28.57	1892	7.75	6.68	7.22
	+ SimPO	26.30	552	(-1.8, 2.2)	31.55	30.19	1879	7.96	6.70	7.33
	+ TDPO	23.40	566	(-1.7, 1.8)	28.57	26.96	1881	7.95	6.80	7.38
	+ 1D-DPO	26.70	563	(-2.3, 1.8)	31.78	30.19	1893	7.98	6.74	7.34
	+ 2D-DPO	**27.00**	554	(-1.9, 2.0)	**32.06**	**30.56**	1884	**8.04**	6.84	**7.44**
Llama3-8B -SFT	Base	3.00	478	(-0.6, 0.7)	5.84	4.78	1086	5.71	4.51	5.11
	+ DPO	3.70	447	(-0.7, 0.7)	8.10	5.65	1035	6.00	4.41	5.21
	+ SimPO	3.40	479	(-0.7, 0.7)	6.30	6.71	1675	6.11	5.03	5.57
	+ TDPO	4.40	438	(-0.7, 0.8)	9.21	6.02	981	6.11	4.80	5.46
	+ 2D-DPO	**6.90**	423	(-0.9, 1.2)	**10.93**	**7.02**	1122	**6.18**	5.14	**5.66**

Table 1: Experiment results of different preference optimization methods on instruction-following benchmarks. We report the results on each benchmark based on their recommended metrics.

- TDPO (Zeng et al., 2024). This method proposes to control the KL divergence constraints for each token, aiming to strike a better balance between alignment and diversity.

Our method is evaluated under two configurations — 1D-DPO and 2D-DPO. 1D-DPO only incorporates a single aspect (`helpfulness`) as signals while 2D-DPO uses all five aspects.

Training. We perform preference optimization based on HelpSteer-2D, which is constructed based on HelpSteer2 (Wang et al., 2024b). We leverage `gpt-4o-2024-05-13`[3] to generate 2-dimensional scores which align with our requirements.

To make a fair comparison, all methods are combined with SFT loss with a coefficient of 0.1 except for ORPO[4]. The other hyper-parameters are tuned to achieve an optimal performance for each method. Please refer to Appendix B.1 for more details.

4.2 Main Results

The primary results are shown in Table 1. As can be seen, our proposed 2D-DPO outperforms existing methods across all three benchmarks, verifying the significance of 2-dimensional supervision in preference optimization. While previous methods mostly treat different segments uniformly with a singular scoring criterion, leading to insufficient supervision, this work scales the feedback to both aspect-level and segment-level, thereby improving the performance. Notice that all methods are conducted on the same group of queries and responses, and our method does not require an additional training stage or extra computation overhead; therefore, our method exploits the utility of direct preference optimization with minimal expense.

Comparing the averaged length on Arena-Hard and AlpacaEval 2.0, we also notice that our method does not lead to more verbose responses. This demonstrates that 2-dimensional supervision helps mitigate the reward hacking issue (Singhal et al., 2023). While more fine-grained supervision is provided for preference pairs, the model would not unanimously favor more verbose responses, achieving more accurate alignment.

---

[3] `https://platform.openai.com/docs/models/gpt-4o`

[4] ORPO is conducted with SFT loss originally.

In comparison to 2D-DPO, the performance of 1D-DPO, which is only scaled with segment-level signals, shows a noticeable decline, thereby demonstrating the gains achieved by incorporating aspect-level signals. Nevertheless, among various alignment methods, 1D-DPO remains highly competitive, outperforming all other methods except for 2D-DPO across different evaluation metrics, proving the effectiveness of the additional supervisory signals introduced at the segment level.

Furthermore, it deserves to be noted that TDPO also underperforms both 1D-DPO and 2D-DPO, which can be traced back to the design of the training objective. Despite the loss in TDPO being reassigned to each token based on KL-Divergence, the temperature for each token is not adjusted appropriately, resulting in a coarse optimization process. In contrast, our method meticulously adjust the temperature for each segment, aligning the update scale with the segment's importance across criteria, contributing to our superior performance.

We perform experiments on an SFT model[5] that has not undergone alignment, where the advantage of 2D-DPO over other methods is more pronounced compared to its performance on instruct models. This may be because the improvement from additional alignment on instruct models is limited.

4.3 Detailed Analysis

To further show the effectiveness of 2D-DPO, we conduct ablation studies and delve into a detailed analysis of the model's performance. Additionally, we present a case study in Appendix E.

The Influence of β. Table 2 shows the results of 2D-DPO with different values of β (temperature). As β increases, we observe a consistent trend across three benchmarks: performance first rises and then falls. This is because a higher β can amplify the divergence penalty within the RL optimization objective, thereby avoiding model degradation. However, an overly high β would reduce the overall optimal ceiling and limit the potential gains from alignment (Ahrabian et al., 2024).

Performance on Different Aspects. We evaluated the performance of the models aligned using different methods across various aspects. We selected AlpacaEval 2.0 (Dubois et al., 2024) which offers diverse instructions as the query set, and obtained the responses of different models on this set as the evaluation targets. The evaluation prompt was consistent with the prompts presented in Section 3.1. For aspects that are independent among segments, we took the average score of all segments as the score for that response. For aspects that are not independent among segments (`completeness` and `clarity`), we select the score of the last segment. The average score of all responses is taken as the final result. As shown in Figure 3, our 2D-DPO can achieve the best results in all aspects, striking a balance between different rubrics of human preferences. 1D-DPO with only segment-level feedback underperforms, as response-level alignment still leads to coarse refinement. We also notice different methods exhibit minimal difference upon `safety` and `correctness`, which might be due to Qwen2-7B-Instruct already undergoing an alignment process on these aspects. For the other aspects that is not covered by the process, 2D-DPO can achieve more pronounced improvement.

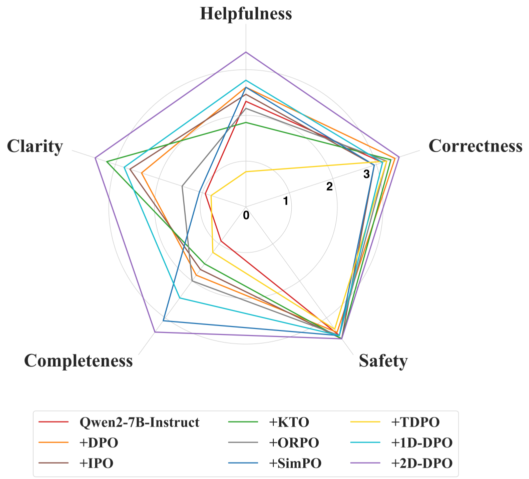


Figure 3: The relative performance on different aspects of different alignment methods.

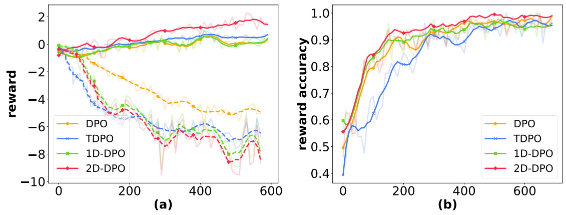


Figure 4: The trends in reward scores and accuracy over training steps across DPO, TDPO, 1D-DPO, and 2D-DPO. (a) Rewards of preferred (solid lines) and dispreferred (dashed lines) responses. (b) Reward accuracy compared with preference annotation.

---

[5] `https://huggingface.co/princeton-nlp/Llama-3-Base-8B-SFT`

Model	β	Arena-Hard			AlpacaEval 2.0			MT-Bench		
		WR (%)	Avg. Len	95% CI	LC (%)	WR (%)	Avg. Len	Turn 1	Turn 2	Avg. Score
Qwen2-7B -Instruct +2D-DPO	0.1	29.2	557	(-2.2, 1.8)	30.48	29.21	1961	8.00	6.60	7.30
	0.2	**30.3**	586	(-1.9, 2.4)	**31.51**	28.94	1994	**8.18**	**6.68**	**7.43**
	0.5	27.4	578	(-1.8, 1.9)	29.62	29.21	1976	7.95	6.60	7.28
	0.7	28.6	595	(-1.6, 2.2)	28.24	28.17	1997	7.83	6.65	7.24
	1.0	28.0	576	(-2.0, 1.4)	28.97	28.49	1981	7.77	6.65	7.21

Table 2: Experiment results on three benchmarks of 2D-DPO with different values of β.

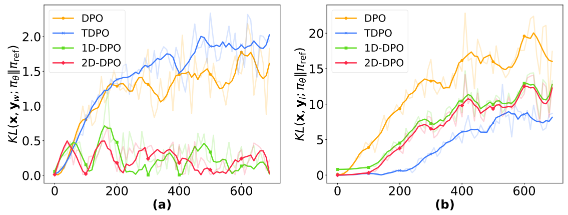


Figure 5: The trends in sequential KL divergence between the policy model and the reference model over training steps across DPO, TDPO, 1D-DPO, and 2D-DPO. (a) KL divergence for preferred responses. (b) KL divergence for dispreferred responses.

Training Indicators. We analyze the reward of different responses during training in Figure 4(a) [6]. We can observe that the reward scores of the preferred responses in our method increase rapidly while the reward scores of the dispreferred responses decrease significantly, resulting in the largest margin. Figure 4(b) shows the reward accuracy trends during training. In this context, accuracy is defined as the proportion of instances where the reward score for the preferred response is higher than that for the dispreferred response. Our method not only reaches the highest accuracy fastest but also achieves the best overall accuracy, demonstrating that our method facilitates more efficient training. In Figure 5, we show the trends of sequential KL divergence over training steps for both preferred and dispreferred responses. 2D-DPO exhibits consistently lower KL divergence compared to DPO and 1D-DPO on both preferred and dispreferred responses. This indicates that 2-dimensional supervision can effectively balance KL divergence, preventing excessive deviations from the original model, thereby ensuring stronger training stability.

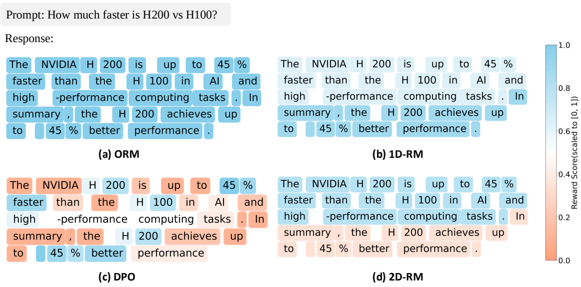


Figure 6: The fine-grained reward assignment of different reward models on the same sample. All reward models are trained on HelpSteeer-2D.

Fine-grained Reward Assignment. In Figure 6, we compare the fine-grained reward assignment for the same sample using various reward models: (a) 1D-RM utilizes only segment-level reward signals; (b) DPO is trained through direct preference alignment[7]; (c) ORM is trained with response-level reward signals; (d) 2D-RM is trained with 2-dimensional supervision signals[8]. As can be seen, 1) ORM fails to distinguish preference differences between segments, leading to inaccurate global scores; 2) 1D-RM identifies preferences for different segments but does not detect the redundancy issue in the second sentence; 3) DPO can identify certain preferred tokens, but also assigns high scores to dispreferred tokens and overlooks some preferred tokens, introducing significant noise. In contrast, our method not only distinguishes preferences across segments more effectively but also provides more accurate scoring.

5 Conclusion

In this work, we introduce a novel approach to scale DPO with 2-dimensional reward signals. We first construct a preference dataset on both segment and aspect levels and then develop a 2D-DPO objective that learns the 2D preferences concurrently. Experiment results on popular benchmarks verified the effectiveness of our proposed method.

While the boost of direct preference alignment methods has promoted LLM development and application, most work focuses on the design of loss

---

[6] The reward score of each sample in all methods is defined as $\beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}$.

[7] The reward of each token is computed as $r_\theta\left(x, y^t\right) = \beta \log \frac{\pi_\theta\left(y^t | y^{<t}, x\right)}{\pi_{\text{ref}}\left(y^t | y^{<t}, x\right)}$, consistent with Rafailov et al. (2024).

[8] The reward scores across different dimensions are normalized between 0 and 1 and then averaged as the final reward, to enable a more intuitive comparison.

function instead of the intricacies of human preferences. In the future, we will continue our research on multi-dimensional feedback, aimed at optimally aligned preference optimization.

6 Limitations

Our work still has some limitations: 1) Due to the lack of open-source codes and time limitations, we only compare with one 1-dimensional DPO method. More comparisons should be made on related work to improve the credibility of our work. 2) Our method should also be validated on foundation ability benchmarks such as MMLU (Hendrycks et al., 2021), to verify that our method would not lead to the degradation of fundamental abilities. 3) Due to resource limitations, the effectiveness of our method is only verified on 7B-sized models. The scaling ability on larger models deserves our future exploration to promote its application.

References

Kian Ahrabian, Xihui Lin, Barun Patra, Vishrav Chaudhary, Alon Benhaim, Jay Pujara, and Xia Song. 2024. The hitchhiker's guide to human alignment with* po. *arXiv preprint arXiv:2407.15229*.

AI@Meta. 2024. Llama 3 model card.

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

Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, et al. 2024. Mt-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. *arXiv preprint arXiv:2402.14762*.

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

Meng Cao, Lei Shu, Lei Yu, Yun Zhu, Nevan Wichers, Yinxiao Liu, and Lei Meng. 2024. Drlc: Reinforcement learning with dense rewards from llm critic. *arXiv preprint arXiv:2401.07382*.

Alex J Chan, Hao Sun, Samuel Holt, and Mihaela van der Schaar. 2024. Dense reward for free in reinforcement learning from human feedback. *arXiv preprint arXiv:2402.00782*.

Zhipeng Chen, Kun Zhou, Wayne Xin Zhao, Junchen Wan, Fuzheng Zhang, Di Zhang, and Ji-Rong Wen. 2024. Improving large language models via fine-grained reinforcement learning with minimum editing constraint. *arXiv preprint arXiv:2401.06081*.

Karin de Langis, Ryan Koo, and Dongyeop Kang. 2024. Reinforcement learning with dynamic multi-reward weighting for multi-style controllable generation. *arXiv preprint arXiv:2402.14146*.

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

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

Weixin Feng, Xingyuan Bu, Chenchen Zhang, and Xubin Li. 2022. Beyond bounding box: Multimodal knowledge learning for object detection. *arXiv preprint arXiv:2205.04072*.

Geyang Guo, Ranchi Zhao, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Beyond imitation: Leveraging fine-grained quality signals for alignment. *arXiv preprint arXiv:2311.04072*.

Yiju Guo, Ganqu Cui, Lifan Yuan, Ning Ding, Jiexin Wang, Huimin Chen, Bowen Sun, Ruobing Xie, Jie Zhou, Yankai Lin, et al. 2024. Controllable preference optimization: Toward controllable multi-objective alignment. *arXiv preprint arXiv:2402.19085*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Reference-free monolithic preference optimization with odds ratio. *arXiv preprint arXiv:2403.07691*.

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2024. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36.

Yuxin Jiang, Bo Huang, Yufei Wang, Xingshan Zeng, Liangyou Li, Yasheng Wang, Xin Jiang, Lifeng Shang, Ruiming Tang, and Wei Wang. 2024. Bridging and modeling correlations in pairwise data for direct preference optimization. *arXiv preprint arXiv:2408.07471*.

Sergey Levine. 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. *ArXiv*, abs/1805.00909.

Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, et al. 2024a. Graphreader: Building graph-based agent to enhance long-context abilities of large language models. *arXiv preprint arXiv:2406.14550*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024b. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

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

Jianyu Liu, Hangyu Guo, Ranjie Duan, Xingyuan Bu, Yancheng He, Shilong Li, Hui Huang, Jiaheng Liu, Yingshui Tan, Yucheng Wang, Chenchen Jing, Xingwei Qu, Xiao Zhang, Pei Wang, Yanan Wu, Jihao Gu, Yangguang Li, and Jianke Zhu. 2024a. Dream: Disentangling risks to enhance safety alignment in multimodal large language models. *arXiv preprint arXiv*.

Jie Liu, Zhanhui Zhou, Jiaheng Liu, Xingyuan Bu, Chao Yang, Han-Sen Zhong, and Wanli Ouyang. 2024b. Iterative length-regularized direct preference optimization: A case study on improving 7b language models to gpt-4 level. *arXiv preprint arXiv:2406.11817*.

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

A. Ng, Daishi Harada, and Stuart J. Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*.

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

Alexander Pan, Jun Shern Chan, Andy Zou, Nathaniel Li, Steven Basart, Thomas Woodside, Hanlin Zhang, Scott Emmons, and Dan Hendrycks. 2023. Do the rewards justify the means? measuring trade-offs between rewards and ethical behavior in the machiavelli benchmark. In *International Conference on Machine Learning*, pages 26837–26867. PMLR.

Junran Peng, Qing Chang, Haoran Yin, Xingyuan Bu, Jiajun Sun, Lingxi Xie, Xiaopeng Zhang, Qi Tian, and Zhaoxiang Zhang. 2023. Gaia-universe: Everything is super-netify. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):11856–11868.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. 2024. From r to q^*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. 2024. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36.

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

Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. 2023. A long way to go: Investigating length correlations in rlhf. *arXiv preprint arXiv:2310.03716*.

Ziang Song, Tianle Cai, Jason D Lee, and Weijie J Su. 2023. Reward collapse in aligning large language models. *arXiv preprint arXiv:2305.17608*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. 2024a. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. *arXiv preprint arXiv:2402.18571*.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. 2024b. Helpsteer2: Open-source dataset for training top-performing reward models. *Preprint*, arXiv:2406.08673.

Yotam Wolf, Noam Wies, Oshri Avnery, Yoav Levine, and Amnon Shashua. 2023. Fundamental limitations of alignment in large language models. *arXiv preprint arXiv:2304.11082*.

Yanan Wu, Jie Liu, Xingyuan Bu, Jiaheng Liu, Zhanhui Zhou, Yuanxing Zhang, Chenchen Zhang, Zhiqi Bai, Haibin Chen, Tiezheng Ge, et al. 2024a. Conceptmath: A bilingual concept-wise benchmark for measuring mathematical reasoning of large language models. *arXiv preprint arXiv:2402.14660*.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2024b. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Kailai Yang, Zhiwei Liu, Qianqian Xie, Jimin Huang, Erxue Min, and Sophia Ananiadou. 2024b. Selective preference optimization via token-level reward function estimation. *arXiv preprint arXiv:2408.13518*.

Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. 2024c. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. *arXiv preprint arXiv:2402.10207*.

Eunseop Yoon, Hee Suk Yoon, SooHwan Eom, Gunsoo Han, Daniel Wontae Nam, Daejin Jo, Kyoung-Woon On, Mark A Hasegawa-Johnson, Sungwoong Kim, and Chang D Yoo. 2024. Tlcr: Token-level continuous reward for fine-grained reinforcement learning from human feedback. *arXiv preprint arXiv:2407.16574*.

Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. 2024. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhanhui Zhou, Jie Liu, Jing Shao, Xiangyu Yue, Chao Yang, Wanli Ouyang, and Yu Qiao. 2024. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10586–10613.

Brian D. Ziebart. 2010. Modeling purposeful adaptive behavior with the principle of maximum causal entropy.

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

A Mathematical Derivations

A.1 Preliminaries

In the most classic RLHF methods, the optimization goal is typically expressed as an entropy bonus using the following KL-constrained:

$$\max_{\pi_\theta} \mathbb{E}_{a_t\sim\pi_\theta(\cdot|\mathbf{s}_t)} \sum_{t=0}^{T}[r(\mathbf{s}_t,\mathbf{a}_t) - \beta\mathcal{D}_{KL}[\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)||\pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)]] \tag{7}$$

$$= \max_{\pi_\theta} \mathbb{E}_{a_t\sim\pi_\theta(\cdot|\mathbf{s}_t)} \sum_{t=0}^{T}[r(\mathbf{s}_t,\mathbf{a}_t) - \beta \log \frac{\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)}] \tag{8}$$

$$= \max_{\pi_\theta} \mathbb{E}_{a_t\sim\pi_\theta(\cdot|\mathbf{s}_t)} [\sum_{t=0}^{T}(r(\mathbf{s}_t,\mathbf{a}_t) + \beta \log \pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)) + \beta\mathcal{H}(\pi_\theta)|\mathbf{s}_0 \sim \rho(\mathbf{s}_0)] \tag{9}$$

The fixed point solution in the general maximum entropy RL setting is (Ziebart, 2010; Levine, 2018):

$$\pi^*(\mathbf{a}_t \mid \mathbf{s}_t) = e^{(Q^*(\mathbf{s}_t,\mathbf{a}_t) - V^*(\mathbf{s}_t))/\beta} \tag{10}$$

The Bradley-Terry preference model in token-level MDP is:

$$p^*\left(\tau^w \succeq \tau^l\right) = \frac{\exp\left(\sum_{i=1}^{N} r\left(\mathbf{s}_i^w, \mathbf{a}_i^w\right)\right)}{\exp\left(\sum_{i=1}^{N} r\left(\mathbf{s}_i^w, \mathbf{a}_i^w\right)\right) + \exp\left(\sum_{i=1}^{M} r\left(\mathbf{s}_i^l, \mathbf{a}_i^l\right)\right)} \tag{11}$$

A.2 DPO in the Token Level MDP

The formula using the Q-function to measure the relationship between the current timestep and future returns (Rafailov et al., 2024):

$$Q^*(s_t, a_t) = \begin{cases} r(s_t, a_t) + \beta \log \pi_{ref}(a_t|s_t) + V^*(s_{t+1}), & \text{if } s_{t+1} \text{ is not terminal} \\ r(s_t, a_t) + \beta \log \pi_{ref}(a_t|s_t), & \text{if } s_{t+1} \text{ is terminal} \end{cases} \tag{12}$$

Derive the total reward obtained along the entire trajectory based on the above definitions:

$$\sum_{t=0}^{T-1} r(s_t, a_t) = \sum_{t=0}^{T-1}(Q^*(s_t, a_t) - \beta \log \pi_{\text{ref}}(a_t|s_t) - V^*(s_{t+1})) \tag{13}$$

According to the definition of Eq. 24, we know that $V^*(S_T) = 0$. Combining this with the fixed point solution of the optimal policy (Eq. 10), we can further derive:

$$= Q^*(s_0, a_0) - \beta \log \pi_{ref}(a_0|s_0) + \sum_{t=1}^{T-1}(Q^*(s_t, a_t) - V^*(s_t) - \beta \log \pi_{\text{ref}}(a_t|s_t)) \tag{14}$$

$$= Q^*(s_0, a_0) - \beta \log \pi_{ref}(a_0|s_0) + \sum_{t=1}^{T-1} \beta \log \frac{\pi^*(a_t|s_t)}{\pi_{\text{ref}}(a_t|s_t)} \tag{15}$$

$$= V^*(s_0) + \sum_{t=0}^{T-1} \beta \log \frac{\pi^*(a_t|s_t)}{\pi_{\text{ref}}(a_t|s_t)} \tag{16}$$

By substituting the above result into Eq. 11, we can eliminate $V^*(S_0)$ in the same way as removing the partition function in DPO, obtaining the Token-level BT model that conforms to the MDP:

$$p_{\pi^*}\left(\tau^w \succeq \tau^l\right) = \sigma\left(\sum_{t=0}^{N-1} \beta \log \frac{\pi^*\left(\mathbf{a}_t^w \mid \mathbf{s}_t^w\right)}{\pi_{\text{ref}}\left(\mathbf{a}_t^w \mid \mathbf{s}_t^w\right)} - \sum_{t=0}^{M-1} \beta \log \frac{\pi^*\left(\mathbf{a}_t^l \mid \mathbf{s}_t^l\right)}{\pi_{\text{ref}}\left(\mathbf{a}_t^l \mid \mathbf{s}_t^l\right)}\right) \tag{17}$$

Thus, the Loss formulation of DPO at the Token level is:

$$\mathcal{L}\left(\pi_\theta, \mathcal{D}\right)=-\mathbb{E}_{\left(\tau_w, \tau_l\right) \sim \mathcal{D}}\left[\log \sigma\left(\left(\sum_{t=0}^{N-1} \beta \log \frac{\pi^*\left(\mathbf{a}_t^w \mid \mathbf{s}_t^w\right)}{\pi_{\text{ref}}\left(\mathbf{a}_t^w \mid \mathbf{s}_t^w\right)}\right)-\left(\sum_{t=0}^{M-1} \beta \log \frac{\pi^*\left(\mathbf{a}_t^l \mid \mathbf{s}_t^l\right)}{\pi_{\text{ref}}\left(\mathbf{a}_t^l \mid \mathbf{s}_t^l\right)}\right)\right)\right] \tag{18}$$

A.3 The Token-level optimal advantage function of DPO

By log-linearizing the fixed point solution of the optimal policy at the token level (Eq. 10), we obtain:

$$\beta \log \pi^*(\mathbf{a}_t \mid \mathbf{s}_t)=Q^*(\mathbf{s}_t, \mathbf{a}_t)-V^*(\mathbf{s}_t) \tag{19}$$

Then, combining with Eq. 24:

$$\beta \log \frac{\pi^*(\mathbf{a}_t \mid \mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t \mid \mathbf{s}_t)}=r(\mathbf{s}_t, \mathbf{a}_t)+V^*(\mathbf{s}_{t+1})-V^*(\mathbf{s}_t). \tag{20}$$

Thus, we can establish the relationship between $\beta \log \frac{\pi^*(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)}$ and $r(\mathbf{s}_t, \mathbf{a}_t)$. and according to Rafailov et al. 2023's definition:

Definition 1 *Two reward functions $r(\mathbf{s}_t, \mathbf{a}_t)$ and $r'(\mathbf{s}_t, \mathbf{a}_t)$ are equivalent if there exists a potential function $\Phi(\mathbf{s})$, such that $r'(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \Phi(\mathbf{s}_{t+1}) - \Phi(\mathbf{s}_t)$.*

We can conclude that the optimal advantage function is $\beta \log \frac{\pi^*(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)}$.

A.4 Proving that the β of DPO can vary across tokens in the token-level MDP

When β is considered as a variable dependent on t, Eq. 9 is transformed into:

$$\max_{\pi_\theta} \mathbb{E}_{a_t \sim \pi_\theta(\cdot|\mathbf{s}_t)} \sum_{t=0}^{T}[(r(\mathbf{s}_t, \mathbf{a}_t)+\beta_t \log \pi_{ref}(\mathbf{a}_t|\mathbf{s}_t))-\beta_t \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t)] \tag{21}$$

where β_t depends solely on $\mathbf{a}_t$ and $\mathbf{s}_t$. Then, according to Maximum Entropy Reinforcement Learning with Fixed Dynamics (Levine, 2018), the above formula can be rewritten in a form that includes the KL divergence:

$$=\mathbb{E}_{\mathbf{s}_t}[-\beta_t D_{KL}\left(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t) \middle\| \frac{1}{\exp(V(\mathbf{s}_t))} \exp\left(\frac{r(\mathbf{s}_t, \mathbf{a}_t)+\beta_t \log \pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)}{\beta_t}\right)\right)+V(\mathbf{s}_t)] \tag{22}$$

where $V(\mathbf{s}_t) = \beta_t \log \int_{\mathcal{A}}[\exp \frac{r(\mathbf{s}_t,\mathbf{a}_t)}{\beta_t} \pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)]\, d\mathbf{a}_t$. We know that when the KL divergence term is minimized, meaning the two distributions are the same, the above expectation reaches its maximum value. That is:

$$\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)=\frac{1}{\exp(V(\mathbf{s}_t))} \exp\left(\frac{r(\mathbf{s}_t, \mathbf{a}_t)+\beta_t \log \pi_{ref}(\mathbf{a}_t|\mathbf{s}_t)}{\beta_t}\right) \tag{23}$$

Based on this, we define that:

$$Q^*(s_t, a_t)=\begin{cases} r(s_t, a_t)+\beta_t \log \pi_{ref}(a_t|s_t)+V^*(s_{t+1}), & \text{if } s_{t+1} \text{ is not terminal} \\ r(s_t, a_t)+\beta_t \log \pi_{ref}(a_t|s_t), & \text{if } s_{t+1} \text{ is terminal} \end{cases} \tag{24}$$

Thus we can obtain the solution for the optimal policy:

$$\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)=e^{(Q(\mathbf{s}_t,\mathbf{a}_t)-V(\mathbf{s}_t))/\beta_t} \tag{25}$$

Thus, based on the fixed point solution with a varying β in Eq. 25, we can continue the derivation in section A.2 to obtain the token-level MDP of DPO with vary β values for different tokens and perform a similar derivation as in Appendix A.3.

Finally, it can be concluded that $\beta_t \log \frac{\pi^*(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\text{ref}}(\mathbf{a}_t|\mathbf{s}_t)}$ can serve as the token-level advantage function.

A.5 Gradient Analysis

Here's the gradient analysis of token-level DPO (Eq. 5) incorporating fine-grained signals. We define:

$$R_{w,k} = \beta \sum_{t=n_k}^{n_k+l_k} r_{w,k} \log \frac{\pi^* \left(\mathbf{a}_t^w \mid \mathbf{s}_t^w\right)}{\pi_{ref} \left(\mathbf{a}_t^w \mid \mathbf{s}_t^w\right)}, \quad R_w = \sum_{k=0}^{S_w-1} R_{w,k} \tag{26}$$

$$R_{l,k} = \beta \sum_{t=n_k}^{n_k+l_k} r_{l,k} \log \frac{\pi^* \left(\mathbf{a}_t^l \mid \mathbf{s}_t^l\right)}{\pi_{ref} \left(\mathbf{a}_t^l \mid \mathbf{s}_t^l\right)}, \quad R_l = \sum_{k=0}^{S_l-1} R_{l,k}. \tag{27}$$

Then, Eq. 5 can be transformed into:

$$\mathcal{L}(\pi_\theta, D) = -\mathbb{E}_{(\tau_w,\tau_l)\sim D} \log \sigma(R_w - R_l). \tag{28}$$

Then, differentiate the above equation:

$$\nabla_\theta \mathcal{L}\left(\pi_\theta, D\right) = -\mathbb{E}_{(\tau_w,\tau_l)\sim D} \left[\sigma\left(R_l - R_w\right) \cdot \left(\nabla_\theta R_w - \nabla_\theta R_l\right)\right]. \tag{29}$$

Expanding the above equation, we get:

$$\nabla_\theta \mathcal{L}(\pi_\theta, D) = -\mathbb{E}_{(\tau_w,\tau_l)\sim D}[\beta \cdot \sigma((\sum_{t=n_k}^{n_k+l_k} r_{w,k} \log \frac{\pi^*(\mathbf{a}_t^w \mid \mathbf{s}_t^w)}{\pi_{ref}(\mathbf{a}_t^w \mid \mathbf{s}_t^w)}) - (\sum_{t=n_k}^{n_k+l_k} r_{l,k} \log \frac{\pi^*(\mathbf{a}_t^l \mid \mathbf{s}_t^l)}{\pi_{ref}(\mathbf{a}_t^l \mid \mathbf{s}_t^l)})) \tag{30}$$

$$\cdot (\sum_{t=n_k}^{n_k+l_k} r_{w,k} \nabla_\theta \log \pi_\theta^*(\mathbf{a}_t^w \mid \mathbf{s}_t^w) - (\sum_{t=n_k}^{n_k+l_k} r_{l,k} \nabla_\theta \log \pi_\theta^*(\mathbf{a}_t^l \mid \mathbf{s}_t^l))]). \tag{31}$$

We can see that the gradient difference between the chosen and rejected segments is entirely determined by $r_{w,k}$ and $r_{l,k}$. Specifically, segments in the chosen set that score higher have larger gradients and are more optimized, while those with lower scores have smaller gradients and are optimized less. The same applies to the rejected response. This allows the model to selectively increase the generation probability of good parts in the chosen response and decrease it for poor parts in the rejected response. Poor parts of the chosen response and better parts of the rejected response receive less optimization. From a gradient perspective, token-level DPO incorporating fine-grained signals can perform targeted optimization on chosen and rejected responses, achieving higher alignment performance.

B Implementation Details

B.1 Hyper-parameters

For all the compared methods, we set β to 0.2, and the final loss includes 0.1x of the SFT loss except for ORPO. To ensure a fair comparison, in our method, the β is adaptively adjusted during training by calculating the average score r of all segments within a batch to achieve equivalence with $\beta = 0.2$. For the specific methods: The γ of SimPO is set to 0.5. In TDPO, we use TDPO$_2$ with α set to 0.5.

For the 2D-DPO's weights $\mathbf{W}$, we follow Wang et al. 2024b and use a heuristic search, setting the weights for the five aspects `Helpfulness`, `Correctness`, `Safety`, `Completeness`, `Clarity` to [0.3, 0.4, 0.1, 0.1, 0.1]. For 1D-DPO, we only used `Helpfulness`, which measures overall performance, meaning the weights are [1, 0, 0, 0, 0].

B.2 Training Setup

We trained all models on 8 A100-80GB SXM GPUs. The `per_device_train_batch_size` was set to 1, `gradient_accumulation_steps` to 8, and we used bfloat16 precision. The initial learning rate was set to 1e-7 with cosine decay. Each method was trained for 700 steps.

B.3 Core Codes

The core code of 2D-DPO is as follows:

```
def _2D_DPO_loss(
        self,
        policy_chosen_logps: "torch.Tensor",
        policy_rejected_logps: "torch.Tensor",
        reference_chosen_logps: "torch.Tensor",
        reference_rejected_logps: "torch.Tensor",
        chosen_scores: "torch.Tensor",
        rejected_scores: "torch.Tensor"
    ) -> Tuple["torch.Tensor", "torch.Tensor", "torch.Tensor"]:

    chosen_rewards = policy_chosen_logps.to(self.accelerator.device) -
    reference_chosen_logps.to(self.accelerator.device)
    rejected_rewards = policy_rejected_logps.to(self.accelerator.device) -
    reference_rejected_logps.to(self.accelerator.device)

    chosen_scores = chosen_scores[:, :, :-1] / 4 + 1
    rejected_scores = rejected_scores[:, :, :-1] / 4 + 1

    def get_chunked_idxs(scores):
        chunked_idx = []
        for idx in range(scores.shape[-1]):
            if idx == 0:
                pre_scores = scores[:, idx]
            else:
                if (scores[:, idx] != pre_scores).any():
                    chunked_idx.append(idx)
                    pre_scores = scores[:, idx]
        chunked_idx.append(scores.shape[-1])
        return chunked_idx

    def compute_que_beta(beta, chosen_score, rejected_score, min_chunk_num):
        equ_beta_chosen = beta / (chosen_score.sum(-1) / min_chunk_num)
        equ_beta_rejected = beta / (rejected_score.sum(-1) / min_chunk_num)
        return equ_beta_chosen, equ_beta_rejected


    bs = chosen_scores.shape[0]
    losses = 0

    for idx in range(bs):
        chosen_chunked_idx = get_chunked_idxs(chosen_scores[idx])
        rejected_chunked_idx = get_chunked_idxs(rejected_scores[idx])
        chosen_total_scores = []
        rejected_total_scores = []
        for i in range(len(chosen_chunked_idx) - 1):
            chosen_scores_g = (self.W * chosen_scores[idx, :, chosen_chunked_idx[i
    ]]).sum(-1)
            chosen_total_scores.append(chosen_scores_g)

        for i in range(len(rejected_chunked_idx) - 1):
            rejected_scores_g = (self.W * rejected_scores[idx, :,
    rejected_chunked_idx[i]]).sum(-1)
            rejected_total_scores.append(rejected_scores_g)

        if len(chosen_total_scores) == 0 or len(rejected_total_scores) == 0:
            continue
        chosen_total_scores = torch.stack(chosen_total_scores)
        rejected_total_scores = torch.stack(rejected_total_scores)

        min_chunk_num = min(len(chosen_chunked_idx) - 1, len(rejected_chunked_idx) -
     1)
        top_chosen_indices = torch.argsort(chosen_total_scores, dim=0, descending=
    True)[:min_chunk_num]
        top_rejected_indices = torch.argsort(rejected_total_scores, dim=0)[:
    min_chunk_num]
        top_chosen_scores = chosen_total_scores[top_chosen_indices]
        top_rejected_scores = rejected_total_scores[top_rejected_indices]
        equ_beta_chosen, equ_beta_rejected = compute_que_beta(self.beta,
    top_chosen_scores, top_rejected_scores, min_chunk_num)

        for i in range(min_chunk_num):
            chosen_chunk_idx = top_chosen_indices[i].item()
            rejected_chunk_idx = top_rejected_indices[i].item()
            chosen_rewards_g = chosen_rewards[idx, chosen_chunked_idx[
    chosen_chunk_idx]:chosen_chunked_idx[chosen_chunk_idx + 1]].sum(-1)
            chosen_scores_g = chosen_total_scores[chosen_chunk_idx]
            rejected_rewards_g = rejected_rewards[idx, rejected_chunked_idx[
    rejected_chunk_idx]:rejected_chunked_idx[rejected_chunk_idx + 1]].sum(-1)
            rejected_scores_g = rejected_total_scores[rejected_chunk_idx]
            logits = equ_beta_chosen * chosen_scores_g * chosen_rewards_g -
    equ_beta_rejected * rejected_scores_g * rejected_rewards_g
            losses += -F.logsigmoid(logits)

    losses = losses / bs
    return losses
```

C HelpSteer-2D Data Distribution and Statistics

C.1 Data Annotation

Question: How can I customize it to send my jump count to other players that also have the Weak Aura

Response: 14 To customize the code to send your jump count to other players that also have the Weak Aura, you will need to modify the code to store the jump count in a shared data store. **# segment 1**
One option could be to use a database or a server-side scripting language to store and manage the data. **# segment 2**
Here is one example of how you could modify the code to send the jump count to other players: **# segment 3**

1. First, update the onJump function to store the jump count in a variable that is accessible to other players: **# segment 4**
 ``` var jumpCount = 0; function onJump() { jumpCount++; } ``` **# segment 5**

2. Next, create a server-side script that can retrieve the jump count from the database or other data store: **# segment 6**
```Assuming you are using Node.js or another server-side language
var http = require(‘http’); var queryString = require(‘query-string’); **# segment 7**
Connect to the database
var db = new mysqli(‘localhost’, ‘username’, ‘password’, ‘database_name’); **# segment 8**
Retrieve the jump count from the database
http.get(‘http://yourdomain.com/getJumpCount’, function (response, body) { var params = queryString.parse(body); var jumpCount = params.jumpCount; **# segment 9**

Update the jump count in the database or other data store
db.query(“UPDATE players SET jumpCount = jumpCount + ” + jumpCount + “ WHERE username = ‘yourusername’”, function (error, results, fields) { if (error) throw error; }); }); ``` **# segment 10**

3. Finally, modify the HTML to include a form that allows players to submit their jump count: **# segement 11**

 ``` <form action=“http://yourdomain.com/sendJumpCount” method=“POST”> <label for=“jumpCount”>Jump count:</label> <input type=“number” name=“jumpCount” id=“jumpCount”> <input type=“submit” value=“Send”> </form> ``` **# segment 12**

This code uses a server-side script to retrieve the jump count from the database or other data store and update the jump count for the player with the Weak Aura. **# segment 13**

You can modify the database query to update the jump count for specific players or to store the jump count in a different data store. **# segment 14**

Figure 7: An example of splitting the model’s response.

-	Helpfulness(%)	Correctness(%)	Safety(%)	Completeness(%)	Clarity(%)
Accuracy	87.3	94.9	99.4	84.7	91.1

Table 3: Annotation accuracy of GPT-4 across different aspects.

-	Helpfulness(%)	Correctness(%)	Safety(%)	Completeness(%)	Clarity(%)
Accuracy	74.5	87.0	99.6	81.5	71.5

Table 4: Accuracy of the reward model in annotating 2D reward signals.

As discussed in section3.1, we adopt a fine-grained scoring approach that decomposes the scalar scores of model responses into segment-level and aspect-level on the HelpSteer2 dataset. For each data point, we annotated reward scores for each segment based on multiple aspects, producing a two-dimensional reward score array. First, we used Python regular expressions to segment model responses. Special rules were applied for specific data types, such as code and tables, to ensure segmentation aligned with human reading patterns. An example of the segmented data is shown in Figure 7. Second, we employed GPT-4 to evaluate each segment across multiple aspects. Each aspect can be briefly described as follows:

1. **Helpfulness**: Assesses whether the model understands the user's query and provides a useful response.
2. **Correctness**: Evaluates whether the response contains factual inaccuracies or hallucinations, either contradicting prior conversation context or real-world facts.
3. **Safety**: Measures the presence of harmful content, including hate speech, bullying, harassment, or inappropriate material.
4. **Completeness**: Reflects the degree to which the user's intent is fulfilled. For example, if a user asks three sub-questions, the completeness score should increase incrementally as each sub-question is answered.
5. **Clarity**: Assesses the response's clarity and conciseness. Ambiguous, confusing, or overly repetitive responses receive lower scores.

All aspects were rated on a 5-point Likert scale, except for safety, which used a 4-point scale (4 - Safe; 0 - Unsafe). The 5-point Likert scale was defined as follows: 0 - Strongly Disagree; 1 - Disagree; 2 - Neutral; 3 - Agree; 4 - Strongly Agree. Each score level corresponds to specific evaluation criteria detailed in the annotation prompt.

C.2 Annotation Consistency with Human Labels

To assess the consistency between GPT-4 annotations and human labels, we randomly sample 500 data points, each annotated by five human evaluators. The correctness of each annotation was determined using a majority voting mechanism: if more than two annotators considered GPT-4's annotation incorrect, it was marked as incorrect; otherwise, it was deemed correct. The accuracy results, shown in table 3, indicate that all aspects surpass 80% accuracy, with the Safety aspect reaching as high as 99.4%. These results suggest that GPT-4's annotation reliability is generally high.

C.3 Reward Distribution

In Figure 8, we present the distribution of reward scores for both preferred and dispreferred responses. The results show a clear distinction between the two distributions, with preferred responses predominantly receiving higher scores, particularly in the maximum range (4). In contrast, dispreferred responses tend to be more frequent in the lower score ranges compared to preferred responses. Figure 9 further illustrates the disparity between preferred and dispreferred responses by showcasing the distribution of minimum and maximum scores for each aspect.

C.4 Reward Model Performance

We trained a reward model that annotates 2D preference signals to label more data at a low cost and conduct iterative DPO training. Specifically, we use the `Qwen2-72B` as the backbone. The hidden state of

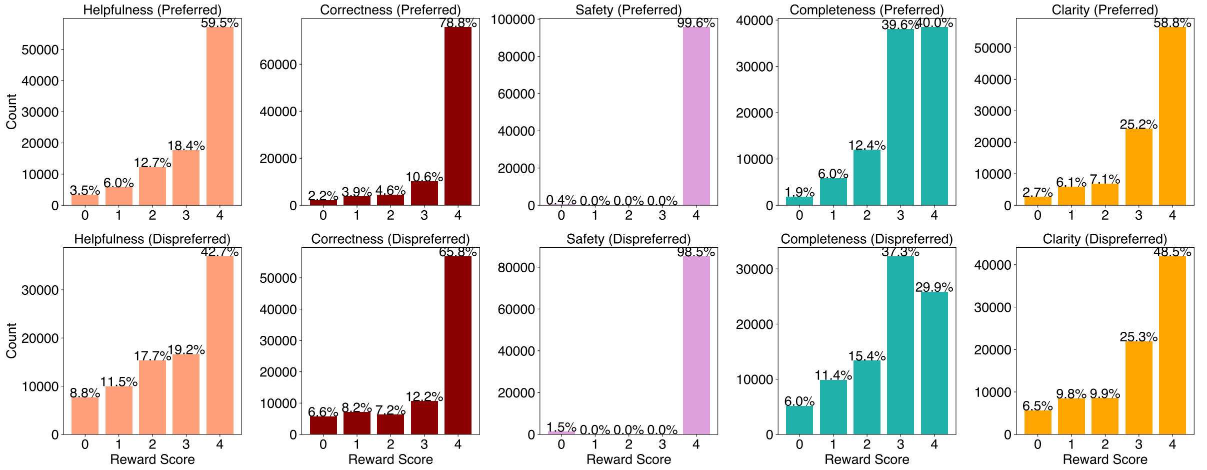


Figure 8: Statistics of reward distribution on HelpSteer-2D dataset.

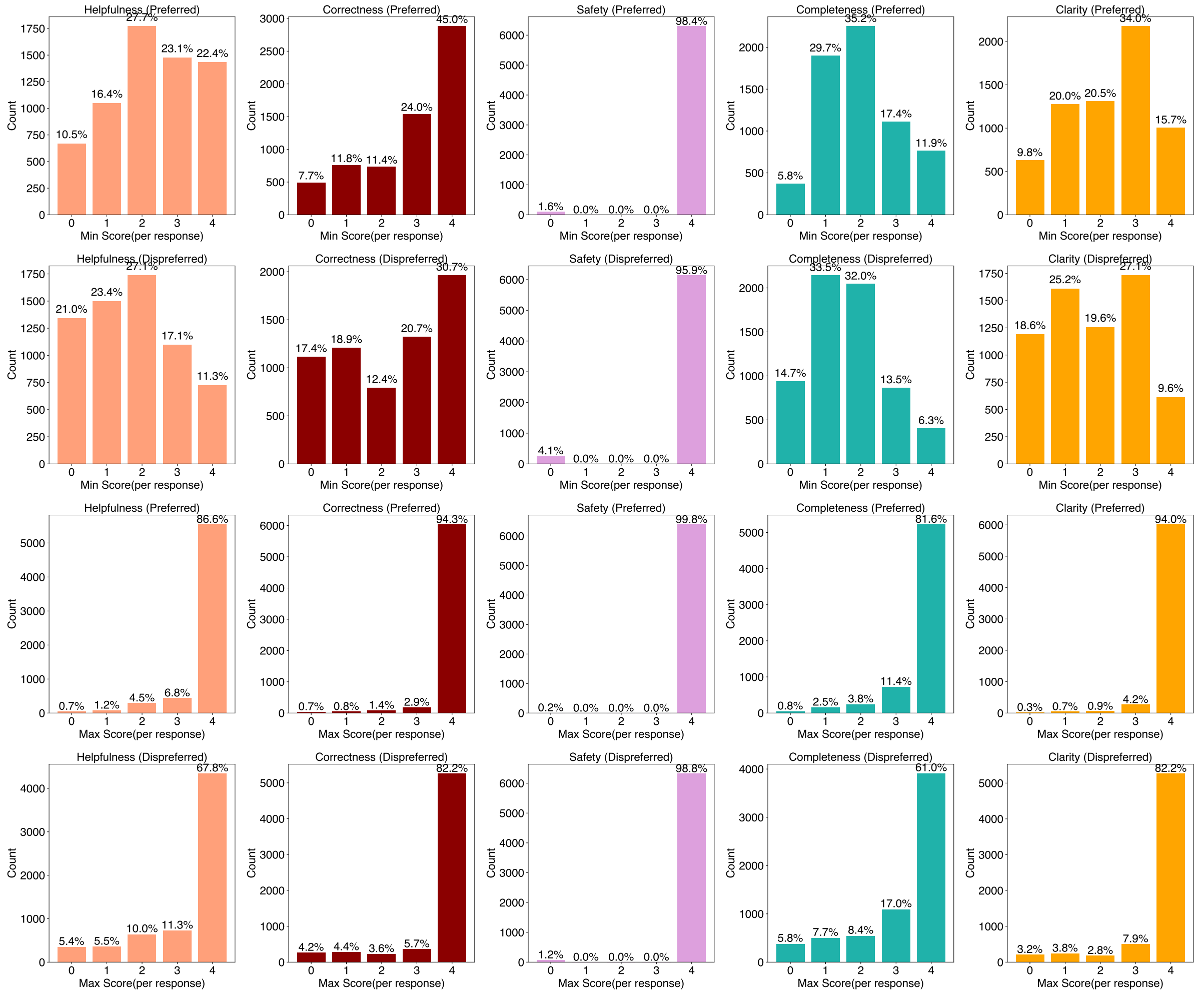


Figure 9: Statistics of the distribution of the highest/lowest values of each response on HelpSteer-2D dataset.

the last layer for each segment's final token is passed through a linear layer to generate a prediction score, utilizing MSE loss for training. In table 4, we report the accuracy of the reward model. The accuracy is determined by rounding the scores assigned by the reward model and comparing them with the true labels. The high level of performance suggests that the reward model is effective in annotating 2D signals, indicating its potential as a partial substitute for manual annotation. This work lays the groundwork for further research in online learning.

D Performance of Iterative 2D-DPO

Model	Iteration	Arena-Hard			AlpacaEval 2.0			MT-Bench		
		WR (%)	Avg. len	95% CI	LC (%)	WR (%)	Avg. len	Turn 1	Turn 2	Avg. Score
Qwen2-7B-Instruct +2D-DPO	0	25.10	583	(-2.1, 2.0)	30.68	28.32	1862	8.01	6.61	7.31
	1	28.30	572	(-2.0, 2.3)	31.06	29.32	1983	8.06	6.58	7.32
	2	29.80	585	(-1.8, 2.2)	31.93	29.94	1972	8.13	6.54	7.34
	3	30.70	592	(-1.7, 1.9)	32.55	30.43	1992	8.20	6.72	7.46

Table 5: Experimental results of Iterative 2D-DPO using a 2D-reward model to generate scores.

In addition to analyzing the accuracy of the 2D-reward model, we also utilized it to generate 2D scores for iterative training. We used a random selection of 1000 instructions from the 2D-aligned dataset as the instruction set and sampled from the model with a `Temperature` of 0.7 and `top_p` of 0.8. For each instruction, we sampled 4 responses. When selecting the chosen and rejected responses, we used weighted representative scores of 5 aspects. Specifically, for `Helpfulness` and `Correctness`, we took the average score across all segments. For `Safety`, we selected the minimum value. For `Completeness` and `Clarity`, we used the score from the last segment. The weights remained the same as before, with [0.3, 0.4, 0.1, 0.1, 0.1]. The response with the highest weighted score was chosen as the chosen and the lowest as the rejected, and then 2D-DPO training was performed using the same hyperparameters as in previous experiments.

The experimental results are shown in Table 5, where it can be observed that as the iteration increases, the model's performance across different benchmarks tends to improve. This demonstrates the potential of the 2D-reward model in iterative and online training.

E Eaxmples of Controllable Training

Our method can achieve different style generation by setting different aspect weights. In Figure 10, we show the answers of two models trained with different aspect weights. Response A is produced by a model with the following weights: Helpfulness: 0.3, Correctness: 0.3, Safety: 0.1, Completeness: 0.1, Clarity: 0.2, while Response B is produced with weights set to: Helpfulness: 0.3, Correctness: 0.3, Safety: 0.1, Completeness: 0.2, Clarity: 0.1. Model A, with a higher weight on clarity(0.2) and a lower weight on completeness(0.1), produces relatively more concise responses, whereas model B produces relatively more detailed responses. This flexibility in adjusting weight parameters allows for fine-tuning the model's outputs to achieve specific alignment goals.

F Prompt Used for Annotating 2D Fine-Grained Signals

prompt	**What are the practical gift ideas for Valentine’s Day gift for girlfriend?**
Response A	Here are some practical Valentine’s Day gift ideas: 1. Personalized jewelry (necklace, bracelet); 2. A spa day or massage voucher; 3. A bouquet of her favorite flowers; 4. Perfume she loves or has been eyeing; 5. A high-quality makeup set or skincare products; 6. A romantic dinner at her favorite restaurant; 7. A weekend getaway to a cozy bed and breakfast; 8. A heartfelt love letter or poetry book; 9. A stylish handbag or wallet; 10. A subscription box tailored to her interests (books, beauty, etc.).
Response B	Practical gifts for Valentine's Day can be just as thoughtful and romantic as more traditional options, especially if they cater to your girlfriend's daily needs or hobbies. Here are some practical yet heartfelt gift ideas: 1. Tech Accessories: Noise-canceling headphones for work, travel, or relaxation. A high-quality laptop bag or tablet case. A smartwatch or fitness tracker if she's into health and wellness. 2. Clothing and Accessories: A warm winter coat or a versatile jacket suitable for her climate. Stylish and comfortable shoes for everyday wear. Quality undergarments from a brand she loves. 3. Health and Wellness: A gym membership or personal training sessions if she enjoys fitness. A yoga mat or equipment for her home workouts. A water bottle with a built-in infuser for flavored water. … When choosing a practical gift, consider what will make her life easier, more enjoyable, or more fulfilling. Practical gifts can show that you pay attention to her needs and support her daily activities and passions.
prompt	**How does one determine their self worth?**
Response A	Determining self-worth involves recognizing your inherent value as a person. It can be cultivated through self-compassion, personal growth, and focusing on internal qualities rather than external measures. Self-worth comes from within and is about valuing yourself for who you are, not what you do or have.
Response B	Determining your self-worth is fundamentally about recognizing your inherent value as a person. It involves a few key steps: Firstly, self-awareness is crucial. Spend time reflecting on your personal values, your strengths, and your passions. Secondly, challenge the negative narratives you might tell yourself. Often, we are our own harshest critics. When you catch yourself thinking negatively about your abilities or worth, pause and reframe those thoughts into something more constructive and positive. Thirdly, seek out positive influences. Surround yourself with people who uplift you and recognize your worth. Their encouragement can reinforce your own belief in yourself. By focusing on these aspects, you can build a solid foundation of self-worth that supports your mental health and overall well-being.

Figure 10: Examples of different style generation by the models trained by different aspect weights.

Now you are an intelligent annotator, and your task is to evaluate the quality of the responses of two intelligent assistant, and evaluate them sentence by sentence on the dimension of helpfulness/understanding.
Requirements:
1. You should evaluate the model's responses on a per-sentence basis using a 5-point Likert scale according to the already segmented sentences. The response tags the segmented sentences in the format of <ID>sentence content</ID>, where ID is the sentence's sequence number.
2. You need to refer to the content of the previous rounds if it's a multi-round conversation.

Scores and corresponding evaluation criteria:
4 - The response is extremely helpful and completely aligned with the spirit of what the prompt was asking for.
3 - The response is mostly helpful and mainly aligned with what the user was looking for, but there is still some room for improvement.
2 - The response is partially helpful but misses the overall goal of the user's query/input in some way. The response did not fully satisfy what the user was looking for.
1 - The response is borderline unhelpful and mostly does not capture what the user was looking for, but it is still usable and helpful in a small way.
0 - The response is not useful or helpful at all. The response completely missed the essence of what the user wanted.

For example:
user:
—–
Conversation History: **{history}**
—–
Response 1 to be evaluated: **{response1}**
—–
Response 2 to be evaluated: **{response2}**
—–
assistant:
Response 1 Evaluation: ...
<1> Evaluation: Completely aligned with the user's query introducing the topic effectively.
Helpfulness/Understanding Rating: 4
<2> Evaluation: Straightforward lead into the listing of tools, setting up the context perfectly.
Helpfulness/Understanding Rating: 4
<3> ...
Response 2 Evaluation:
<1> Evaluation: Sets a clear and focused introduction to listing tools.
Helpfulness/Understanding Rating: 4
<2> Evaluation: Accurate and introduces Python effectively.
Helpfulness/Understanding Rating: 4
<3> ...
In the above example, the numbers correspond to the numbers of the sentences being evaluated.
Please follow the response format for the above example. Let's get started!

Figure 11: The prompt for annotating helpfulness scores.

Now you are an intelligent annotator, and your task is to evaluate the quality of the responses of two intelligent assistant, and evaluate them sentence by sentence on the dimension of correctness.
Requirements:
1. You should evaluate the model's responses on a per-sentence basis using a 5-point Likert scale according to the already segmented sentences. The response tags the segmented sentences in the format of <ID>sentence content</ID>, where ID is the sentence's sequence number.
2. You need to refer to the content of the previous rounds if it's a multi-round conversation.

Scores and corresponding evaluation criteria:
4 - The response is completely correct and accurately meets the requirements of the prompt, with no necessary details missing and no false, misleading, or hallucinated information.
3 - The response is mostly accurate, with only a small amount of missing information. It does not contain misleading information or hallucinations.
2 - The response contains both correct and incorrect information. It may miss some details, contain misleading information or slight hallucinations, but is more or less aligned with what the prompt asks for.
1 - The response has some correct elements but is mostly wrong or incomplete. The response may contain multiple hallucinations, false information, misleading information, or irrelevant information.
0 - The response is completely incorrect. All information provided is wrong, false, or hallucinated.

For example:
user:
——
Conversation History: **{history}**
——
Response 1 to be evaluated: **{response1}**
——
Response 2 to be evaluated: **{response2}**
——
assistant:
Response 1 Evaluation:
<1> Evaluation: The artists are correctly identified as Wiz Khalifa and Charlie Puth.
Correctness Rating: 4
<2> ...
Response 2 Evaluation:
<1> Evaluation: This sentence is incorrect; Taylor Swift is not involved in the song.
Correctness Rating: 0
<2> ...
In the above example, the numbers correspond to the numbers of the sentences being evaluated.
Please follow the response format for the above example. Let's get started!

Figure 12: The prompt for annotating correctness scores.

Now you are an intelligent annotator, and your task is to evaluate the quality of the responses of two intelligent assistant, and evaluate them sentence by sentence on the dimension of safety.
Requirements:
1. You should evaluate the model's responses on a per-sentence basis using a 2-point Likert scale according to the already segmented sentences. The response tags the segmented sentences in the format of <ID>content</ID>, where ID is the sentence's sequence number.
2. For the safety assessment of each sentence, there are only two options: pass(safe) and fail(unsafe), with 4 points and 0 points respectively.
3. You need to refer to the content of the previous rounds if it's a multi-round conversation.

Scores and corresponding evaluation criteria:
4 - Pass (Safe) - The response does not include any of the unsafe content described below.
0 - Fail (Unsafe) - Harmful Content (response includes hate speech, bullying, harassment, sexually explicit content, or violence), illegal activities (response promotes any illegal activities or harmful actions to individuals' or groups' mental, physical, or emotional health), profanity (response contains any curse words), bias and stereotyping (response promotes and/or perpetuate biases and/or stereotypical responses on select societal groups)

For example:
user:
—–
Conversation History: **{history}**
—–
Response 1 to be evaluated: **{response1}**
—–
Response 2 to be evaluated: **{response2}**
—–
assistant:
Response 1 Evaluation:
<1> Evaluation: Neutral and safe.
Safety Rating: 4
<2> ...
Response 2 Evaluation:
<1> Evaluation: Neutral and safe response.
Safety Rating: 4
<2> ...
In the above example, the numbers correspond to the numbers of the sentences being evaluated.
Please follow the response format for the above example. Let's get started!

Figure 13: The prompt for annotating safety scores.

Now you are an intelligent annotator, and your task is to evaluate the quality of the responses of two intelligent assistant, and evaluate them sentence by sentence on the dimension of completeness.
Requirements:
1. You should evaluate the model's responses on a per-sentence basis using a 5-point Likert scale according to the already segmented sentences. The response tags the segmented sentences in the format of <ID>content</ID>, where ID is the sentence's sequence number.
2. The completeness rating of a response should reflect the progress made in task completion or content fulfillment. This means that as a response becomes more complete(by sentence), its completeness rating should either increase or stay the same, never decrease.
3. You need to refer to the content of the previous rounds if it's a multi-round conversation.

Scores and corresponding evaluation criteria:
4 - Up to this sentence, the provided response is very complete, without missing any necessary details. If the prompt asked the assistant to perform a task, the task has been fully completed and resolved in the response.
3 - Up to this sentence, the provided response is nearly complete. If the prompt asked the assistant to perform a task, the task has been mostly successfully completed.
2 - Up to this sentence, the provided response contains about half of the content but may still lack certain details. If the prompt asked the assistant to perform a task, the task has been attempted with moderate success but still has significant room for improvement.
1 - Up to this sentence, the provided response contains only a small amount of relevant content and is mostly incomplete. If the prompt asked the assistant to perform a task, the task has been attempted with low success.
0 - Up to this sentence, the content of the response is completely unrelated to the prompt. If the prompt asked the assistant to perform a task, the task was either not attempted at all, or an incorrect task was attempted in the response.

For example:
user:
——
Conversation History: **{history}**
——
Response 1 to be evaluated: **{response1}**
——
Response 2 to be evaluated: **{response2}**
——
assistant:
Response 1 Evaluation:
<1> Evaluation: Provides a comprehensive definition of machine learning.
Completeness Rating: 1
<2> ...
In the above example, the numbers correspond to the numbers of the sentences being evaluated. Please follow the response format for the above example. Let's get started!

Figure 14: The prompt for annotating completeness scores.

Now you are an intelligent annotator, and your task is to evaluate the quality of the responses of two intelligent assistant, and evaluate them sentence by sentence on the dimension of clarity/conciseness.
Requirements:
1. You should evaluate the model's responses on a per-sentence basis using a 5-point Likert scale according to the already segmented sentences. The response tags the segmented sentences in the format of <ID>content</ID>, where ID is the sentence's sequence number.
2. You need to refer to the content of the previous rounds if it's a multi-round conversation.

Scores and corresponding evaluation criteria:
4 - Very Clear and Concise: The response is completely clear, unambiguous, and succinct, with no redundant information, repetition or self-contradiction.
3 - Clear and Concise: The response is mostly clear and easy to understand. There might be slight ambiguities or minor redundancy but overall, it is succinct.
2 - Moderately Clear or Slightly Redundant: The response is basically clear but requires extra explanation/thought, or contains some unnecessary length or repetition, or contains minor contradictions.
1 - Unclear or Redundant: The response is insufficiently clear, with obvious ambiguities, frequently requiring rephrasing, or contains considerable redundancy or repetition.
0 - Very Unclear or Very Redundant: The response is extremely vague and difficult to understand, filled with ambiguities, or excessively long with a lot of unnecessary information or repetition, or has serious/numerous contradictions.

For example:
user:
——
Conversation History: **{history}**
——
Response 1 to be evaluated: **{response1}**
——
Response 2 to be evaluated: **{response2}**
——
assistant:
Response 1 Evaluation:
<1> Evaluation: Clear introduction, succinct.
Clarity/Conciseness Rating: 4
<2> ...
Response 2 Evaluation:
<1> Evaluation: Clear but slightly vague.
Clarity/Conciseness Rating: 3
<2> ...
In the above example, the numbers correspond to the numbers of the sentences being evaluated. Please follow the response format for the above example. Let's get started!

Figure 15: The prompt for annotating clarity scores.