BEYOND PAIRWISE: EMPOWERING LLM ALIGNMENT WITH (RANKED) CHOICE MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

Alignment of large language models (LLMs) has predominantly relied on pairwise preference optimization, where annotators select the better of two responses to a prompt. While simple, this approach overlooks the opportunity to learn from richer forms of human feedback, such as multiwise comparisons and top-k rankings. We propose *Ranked Choice Preference Optimization* (RCPO), a unified framework that bridges preference optimization with (ranked) choice modeling via maximum likelihood estimation. The framework is flexible, supporting both utility-based and rank-based choice models. It subsumes several existing pairwise methods (e.g., DPO, SimPO), while providing principled training objectives for richer feedback formats. We instantiate this framework with two representative ranked choice models (Multinomial Logit and Mallows-RMJ). Empirical studies on Llama-3-8B-Instruct and Gemma-2-9B-it across AlpacaEval 2 and Arena-Hard benchmarks show that RCPO consistently outperforms competitive baselines. RCPO shows how directly leveraging ranked preference data, combined with the right choice models, yields more effective alignment. It offers a versatile and extensible foundation for incorporating (ranked) choice modeling into LLM training.

1 INTRODUCTION

Large language models (LLMs), a prominent form of generative AI, have rapidly transformed human-computer interaction, powering applications from open-ended dialogue (Thoppilan et al., 2022) and content creation (Brown et al., 2020) to code generation (Chen et al., 2021; Li et al., 2023) and healthcare decision support (Nori et al., 2023). A key driver of this success is *alignment*—training models to produce outputs that are factual, helpful, safe, and aligned with social norms.

Reinforcement learning from human feedback (RLHF) (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022) has emerged as the dominant paradigm for aligning AI systems, exemplified by ChatGPT and GPT-4 (Achiam et al., 2023). More recently, Direct Preference Optimization (DPO) Rafailov et al. (2023) has achieved comparable results through a simpler and more efficient objective, and has been used to fine-tune LLMs such as Llama-3 (Dubey et al., 2024) and Zephyr (Tunstall et al., 2023). The success of RLHF and DPO has spurred a surge of research, resulting in numerous extensions (Zhao et al., 2024; Winata et al., 2025).

Despite their differences, most alignment methods rely on *pairwise preference data*, where for each prompt x, a preferred response y_w and a dispreferred response y_l are selected by human annotators or AI judges. In practice, however, preference feedback is often richer than simple pairs. Annotators may provide partial rankings, top-k selections, or single-best judgments from a larger candidate set. Current approaches typically reduce this richer information to pairs—e.g., in training InstructGPT, Ouyang et al. (2022) collected rankings of K responses per prompt but converted them into all $\binom{K}{2}$ pairs; in many academic alignment studies (Meng et al., 2024; Chen et al., 2025; Zhao et al., 2024; Gupta et al., 2025), multiple responses are scored by a reward model, but only the highest- and lowest-scoring are kept. These reductions, while convenient for pairwise-based algorithms, risk distorting the original preference structure and discarding potentially valuable information.

To address this gap, we propose **R**anked **C**hoice **P**reference **O**ptimization (**RCPO**), a general framework that generalizes pairwise preference alignment to ranked choice feedback (see Figure 1).

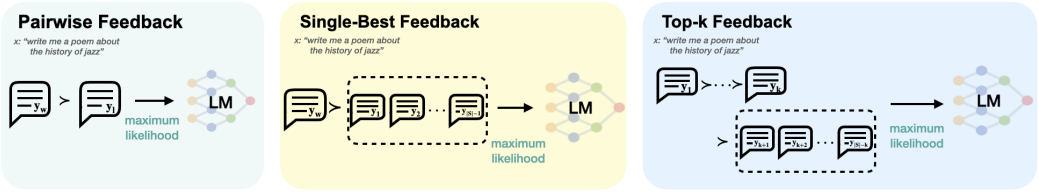


Figure 1: Ranked Choice Preference Optimization (RCPO)

Rather than restricting evaluators to comparing only two responses, RCPO presents a set of candidates and asks them to choose either the single-best or the top-k responses for a given prompt. Our approach is grounded in the theory of *(ranked) choice models*, particularly discrete choice modeling, which have been extensively studied in psychology, marketing, economics, and operations research. To the best of our knowledge, RCPO is among the first attempts to systematically apply (ranked) choice modeling to LLM alignment.

The main contributions of this paper are threefold:

(1) **Conceptual Framework**: We establish a systematic connection between LLM fine-tuning and choice modeling, showing that fine-tuning can be essentially reduced to maximum likelihood estimation (MLE) of choice models. Building on this insight, we develop RCPO as a principled extension of pairwise preference alignment that directly incorporates ranked choice feedback. RCPO is a general and flexible framework: any choice model that satisfies certain regularity conditions can be integrated. By preserving the richness of the original annotations – whether single-best or top-k preferences – RCPO avoids the information loss inherent in pairwise conversion and enables more faithful alignment with human intent.

(2) **Concrete Examples**: We showcase how two broad classes of choice models, i.e., utility- or rank-based, can be accommodated within the RCPO framework. We then instantiate RCPO with a representative model from each class: the Multinomial Logit (MNL) model (McFadden, 1972) for utility-based choices and the Mallows-RMJ model (Feng & Tang, 2022) for rank-based choices. For both models, we derive alignment objectives under single-best and top-k settings (see Table 1). We also use gradient analysis to shed some theoretical insights on the substances of these preference optimization methods.

Table 1: Various Preference Optimization Objectives in RCPO

Choice Model	Preference	Objective
MNL	Pairwise (DPO)	$-\log\sigma\left(f_\theta(x, y_w, y_l)\right)$
	Single-Best	$-\log\sigma\big(-\log\sum_{y_i\in S\setminus\{y_w\}}\exp\left(f_\theta(x, y_i, y_w)\right)\big)$
	Top-k Choice	$-\sum_{i=1}^{k}\log\sigma\big(-\log\sum_{y_j\in S\setminus\{y_1,\ldots,y_i\}}\exp\left(f_\theta(x, y_j, y_i)\right)\big)$
Mallows-RMJ	Pairwise	$-\log\phi(x)\cdot\sigma(f_\theta(x, y_l, y_w))$
	Single-Best	$-\log\phi(x)\cdot\sum_{y_i\in S\setminus\{y_w\}}\sigma(f_\theta(x, y_i, y_w))$
	Top-k Choice	$-\log\phi(x)\big(\sum_{i=1}^{k-1}(\lvert S\rvert - i)\,\sigma(f_\theta(x, y_{i+1}, y_i)) + \sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\sigma(f_\theta(x, y_j, y_k))\big)$

Notes: $f_\theta(x, y_1, y_2) := \beta\log\frac{\pi_\theta(y_1|x)}{\pi_{\text{ref}}(y_1|x)} - \beta\log\frac{\pi_\theta(y_2|x)}{\pi_{\text{ref}}(y_2|x)}$.

(3) **Experiments**: We evaluate RCPO on state-of-the-art instruction-tuned LLMs (Llama-3-8B-instruct and Gemma-2-9B-it) using widely adopted benchmarks (AlpacaEval 2 and Arena-Hard). Results consistently show that RCPO improves model performance and demonstrates flexibility across base models, preference feedback, and evaluation settings. We highlight the Mallows-RMJ-based preference optimization, which achieves strong results under both pairwise and ranked choice setups.

Collectively, these contributions position RCPO as a principled and practical framework for leveraging ranked choice feedback to advance the alignment of large language models.

2 A CHOICE BASED ALIGNMENT FRAMEWORK

2.1 RLHF AND DPO

We start by recapping the key concepts in RLHF and DPO. Let x denote an input prompt and y denote a candidate response. A language model is parameterized by a policy π_θ, where $\pi_\theta(y \mid x)$ represents the probability of generating response y given prompt x.

RLHF comprises three sequential phases. First, it fine-tunes a pre-trained LLM through supervised learning on supervised data and outputs a reference model π_{ref}. Second, RLHF fits a reward model $r^*(x, y)$, which can be a neural network itself, based on a separate pairwise preference dataset $\mathcal{D} = \{(x^{(i)}, y_w^{(i)}, y_l^{(i)})\}_{i=1}^N$. Here $x^{(i)}$ represents the prompt provided to annotator i, and $y_w^{(i)}$ and $y_l^{(i)}$ are the preferred and dis-preferred responses, respectively. Third, the LLM is further fine-tuned via reinforcement learning to maximize the regularized expected reward:

$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}}\big[\mathbb{E}_{y\sim\pi_\theta(y|x)}\big[r^*(x,y)\big] - \beta\,\mathrm{KL}\big(\pi_\theta(\cdot \mid x)\,\|\,\pi_{\text{ref}}(\cdot \mid x)\big)\big], \tag{1}$$

where $\beta > 0$ is a hyperparameter controlling the deviation from the reference policy π_{ref}.

While RLHF has shown impressive results, both reward model fitting and reinforcement learning require substantial computational effort. In this light, DPO analytically solves (1), yielding a closed-form relationship between the optimal policy and the reward model given by:

$$r^*(x,y) = \beta \log \tfrac{\pi_{\theta^*}(y|x)}{\pi_{\text{ref}}(y|x)} \;+\; \beta \log Z(x), \tag{2}$$

where $Z(x) = \sum_y \pi_{\text{ref}}(y \mid x)\,\exp(r^*(x,y)/\beta)$ is the partition function. As a result, based on a Bradley-Terry preference assumption on how the preferred/dis-preferred responses are generated, DPO consolidates the last two steps of RLHF into a direct optimization problem with the following loss function:

$$\min_{\pi_\theta} \mathcal{L}_{\text{DPO}}\big(\pi_\theta; \pi_{\text{ref}}\big) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\big[\log\sigma\big(\beta\log\tfrac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} \;-\; \beta\log\tfrac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\big)\big], \tag{3}$$

where $\sigma(\cdot)$ is the sigmoid function. In other words, DPO directly fine-tunes π_θ to match human pairwise preferences in a single step, thereby greatly reducing computational overhead.

2.2 (RANKED) CHOICE MODELING

Discrete Choice Models. We also introduce the key concepts for (ranked) choice modeling. First, a rich body of research across economics, marketing, and operations research has developed a variety of choice models to represent human preferences. Let $\mathcal{N} = \{y_1, y_2, \ldots, y_n\}$ denote the universe of items. A *discrete choice model* specifies the probability of selecting an *item* y from an *assortment* $S \subseteq \mathcal{N}$ under a given *context* x, denoted by $\mathbb{P}(y \mid S\,; x)$. This is an abstraction of many business and economics use cases. For instance, in retail settings, items typically correspond to products, assortments represent the sets of available products at the time of choice, and the context encompasses covariates such as prices, promotions, or product features. There are many ways to define a discrete choice model, which essentially reduce to specifying the probability function $\mathbb{P}(y \mid S\,; x)$.

Ranked Choice Models. Discrete choice models can be extended to richer feedback in the form of *ranked choices*, which is first formally studied by Feng & Tang (2022) to the best of our knowledge. Let $\mu^k = y_1 \succ y_2 \succ \cdots \succ y_k$ with $\{y_1, \ldots, y_k\} \subseteq S$ denote a top-k list of items from S. A ranked choice model defines $\mathbb{P}(\mu^k \mid S\,; x)$, which specifies the probability of observing the partial ranking μ^k under context x. This notion of ranked choices generalizes several common feedback structures, such as (i) pairwise comparisons; (ii) discrete choices (or multi-wise comparisons); (iii) listwise feedback (or full rankings over the items in any given assortment); (iv) top-k rankings over a full item set, to name a few. For example, when $k = 1$, a ranked choice model reduces to a discrete choice model.

Assumptions. In this paper, we will put (ranked) choice models in the context of LLM alignment, and focus on those that satisfy the following two assumptions:

- *Assumption A1: Reward sufficiency.* There exists a real-valued reward function $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and a mapping g such that, for every S and x,

$$\mathbb{P}(\mu^k \mid S\,; x) \;=\; g\big(\mu^k, S, \{\,r(x,y)\,\}_{y\in S}\big)\,, \quad \text{for every } \mu^k = y_1 \succ y_2 \succ \cdots \succ y_k \text{ within } S.$$

That is, the effect of x and S on ranked choice probabilities enters *only* through the item rewards $\{r(x,y)\}_{y\in S}$. Note that in the special case of discrete choices, the condition above can be simplified to $\mathbb{P}(y \mid S\,; x) = g(y, S, \{\, r(x,y') \,\}_{y'\in S})$, for every $y \in S$.

- *Assumption A2: MLE estimability.* Given observations $\mathcal{D} = \{(x_i, S_i, \mu_i^k)\}_{i=1}^N$, the rewards are identifiable up to the usual invariances (e.g., additive shift and positive scaling) and admit tractable log-likelihood function

$$\sum\nolimits_{i=1}^{N} \log g\big(\mu_i^k, S_i, \{\, r(x_i,y) \,\}_{y\in S_i}\big)\,.$$

Here the meaning of "tractable" will be clearer later. In short, the log-likelihood function should admit simple gradients to be passed to the training of π_θ.

As can be easily seen, many choice models satisfy the assumptions above. In Sections 2.4 and 2.5, we present two approaches: utility-based models and rank-based models.

2.3 RCPO: Connection between LLM Alignment and Ranked Choice Modeling

Our framework starts with a conceptual insight into LLM alignment: if we interpret a prompt x as the *context*, a candidate response y as an *item*, and a set of candidate responses as an *assortment* S, then every choice model offers a distinct way to incorporate annotators' preference feedback via an MLE objective. For instance, consider the case where $S = \{y_w, y_l\}$, and let g denote the Bradley-Terry choice rule. Then the probability that y_w is preferred over y_l is given by:

$$\mathbb{P}(y_w \mid \{y_w, y_l\}\,; x) = g(y_w, \{y_w, y_l\}, \{r(x,y_w), r(x,y_l)\}) = \sigma\big(r(x,y_w) - r(x,y_l)\big),$$

where σ denotes the sigmoid function. Substituting the reward function defined in (2) into the Bradley–Terry likelihood yields the DPO objective described in (3). This establishes that DPO is a special case of our formulation, where preferences follow the Bradley-Terry pairwise comparison model.

The RCPO Framework. Motivated by these connections, we introduce a general framework for preference optimization grounded in choice model theory. Specifically, we extend DPO from pairwise Bradley-Terry comparisons to *arbitrary* ranked choice models that satisfy the assumptions outlined above. In this framework, once the functional forms of $r(x,y)$ is specified, and the choice rule g is determined by a ranked choice model, the corresponding preference optimization procedure is defined by the following maximum likelihood estimation (MLE) objective:

$$\max_{\pi_\theta} \sum\nolimits_{i=1}^{N} \log g\left(\mu_i^k, S_i, \{r_{\pi_\theta}(x_i,y)\}_{y\in S_i}\right),$$

where $r_{\pi_\theta}(x,y)$ is a reward function derived from the policy π_θ.

Beyond the reward function (2) used in DPO, the literature has proposed many alternative definitions. For example, Wang et al. (2023) introduces f-divergence generalizations, while Meng et al. (2024) and Gupta et al. (2025) propose length-normalized log-likelihoods. These alternative reward formulations can likewise be incorporated into the RCPO framework. Consequently, methods such as R-DPO (Park et al., 2024), SimPO (Meng et al., 2024), and AlphaPO (Gupta et al., 2025) can also be viewed as special cases of RCPO. In particular, while these approaches adopt the Bradley–Terry choice rule, they differ in the functional form of the reward $r(x,y)$. Although our framework can accommodate a wide range of reward functions, for clarity, we restrict attention to the reward defined in (2) thereafter in the paper.

What *truly* demonstrates the versatility of RCPO is how it gives birth to new preference optimization methods. In this paper, we consider two prominent classes of choice models: utility-based models, which rely on the numerical magnitudes of the items' utilities, and rank-based models, which depend on the rankings of the items.

2.4 Utility-based Choice Models

The **random utility model (RUM)** is perhaps the most widely studied class of discrete choice models. Originally proposed by Thurstone (2017), it has been extensively developed in the economics and operations management literature (Anderson et al., 1992; Train, 2009). RUM assumes that every item

$y_i \in \mathcal{N}$ comes with a *utility*, which takes the form $u_{y_i} = \nu_{y_i} + \varepsilon_{y_i}$, where ν_{y_i} is the mean utility, and ε_{y_i} is an exogenous random utility shock term. In this paper, we also write u_{y_i} and ν_{y_i} in the form of $u_{y_i}(x)$ and $\nu_{y_i}(x)$ to emphasize that they can be context-dependent. Given the mean utility vector $\nu(x) = (\nu_{y_1}(x), \ldots, \nu_{y_n}(x))$ and a distribution f over the utility shocks $\boldsymbol{\varepsilon} = (\varepsilon_{y_1}, \ldots, \varepsilon_{y_n})$, the probability of choosing alternative y_i from an assortment $S \subseteq \mathcal{N}$ is

$$\mathbb{P}(y_i \mid S\,; x) = \int_{\boldsymbol{\varepsilon}} \mathbb{I}\{\nu_{y_i}(x) - \nu_{y_j}(x) > \varepsilon_{y_j} - \varepsilon_{y_i} \;\; \forall y_j \in S \setminus \{y_i\}\} f(\boldsymbol{\varepsilon})\, \mathrm{d}\boldsymbol{\varepsilon}. \tag{4}$$

RUMs are categorized by the distribution of their stochastic terms. The multinomial logit (MNL) model, introduced by McFadden (1972), assumes i.i.d. Gumbel noise. Alternatives include the probit model (joint normal distribution (Daganzo, 2014)), the nested logit model (correlated extreme value distributions (McFadden, 1980)), and the exponomial model (negative exponential distributions (Alptekinoğlu & Semple, 2016)).

The RUM can be extended to a ranked choice model, where the probability of observing the top-k ranking $\mu^k = y_1 \succ y_2 \succ \cdots \succ y_k$ from an assortment S is

$$\mathbb{P}(\mu^k \mid S\,; x) = \int_{\boldsymbol{\varepsilon}} \prod_{\ell=1}^{k} \mathbb{I}\Big\{\nu_{y_\ell}(x) - \nu_{y_j}(x) > \varepsilon_{y_j} - \varepsilon_{y_\ell},\ \forall y_j \in S \setminus \{y_1, \ldots, y_\ell\}\Big\}\, f(\boldsymbol{\varepsilon})\, \mathrm{d}\boldsymbol{\varepsilon}. \tag{5}$$

As (4) and (5) show, the choice probabilities depend only on the mean utility vector $\nu(x)$, which corresponds to the reward vector in our framework. Thus, any RUM that admits MLE estimation of $\nu(x)$ can be incorporated into LLM fine-tuning.

2.5 Rank-based Choice Models

While utility-based models operate on numerical utilities, rank-based models represent preferences as complete orderings over $\mathcal{N}$, depending only on the relative positions of items. See Jagabathula & Venkataraman (2022) for a survey of such models.

The **Mallows-type model** (Mallows, 1957; Fligner & Verducci, 1986) is among the most widely used classes of ranking models. Let $\mathfrak{S}_n$ be the set of permutations of $\mathcal{N}$. A Mallows-type model assigns a probability distribution over permutations $\mu \in \mathfrak{S}_n$ based on their distance from a central ranking μ_0:

$$\mathbb{P}_{\phi,\mu_0,d}(\mu) := \frac{\phi^{d(\mu_0,\mu)}}{\sum_{\mu'} \phi^{d(\mu_0,\mu')}}, \quad \mu \in \mathfrak{S}_n,$$

where $d(\cdot,\cdot)$ is a distance function between permutations, $\phi \in (0,1)$ is a dispersion parameter. Intuitively, rankings closer to μ_0 are exponentially more likely. Different distance choices yield different variants, such as Kendall's Tau (Mallows, 1957), Spearman's rank and footrule (Diaconis & Graham, 1977), Hamming distance (Bookstein et al., 2002), Cayley distance (Irurozki et al., 2018), and Reverse Major Index (Feng & Tang, 2022). To capture context dependence, we parameterize the central ranking and dispersion as functions of x, denoted $\mu_0(\cdot|x)$ and $\phi(x)$. For readability, we may suppress x in the notation when its dependence is unambiguous.

For a ranking $\mu \in \mathfrak{S}_n$ and an item y, denote $\mu^{-1}(y)$ as the position of y in μ (smaller means higher preference). Given an assortment $S \subseteq \mathcal{N}$ and a top-k ranking $\mu^k = y_1 \succ y_2 \succ \cdots \succ y_k$ with $\{y_1, \ldots, y_k\} \subseteq S$, the implied ranked choice probability of observing μ^k from S is given by

$$\mathbb{P}(\mu^k \mid S\,; x) = \sum\nolimits_{\mu \in \mathfrak{S}_n} \mathbb{P}_{\phi(x),\mu_0(\cdot|x),d}(\mu)\, \mathbb{I}\{\mu, \mu^k, S\}, \tag{6}$$

where $\mathbb{I}\{\mu, \mu^k, S\} = 1$ if μ ranks $\{y_1, \ldots, y_k\}$ in the specified order, and each of them is ranked above all remaining items in S, or more formally, $\mathbb{I}\{\mu, \mu^k, S\} := \mathbf{1}\Big\{\substack{\mu^{-1}(y_1)<\cdots<\mu^{-1}(y_k),\\ \mu^{-1}(y_\ell)<\mu^{-1}(y_j)\ \forall y_j \in S\setminus\{y_1,\ldots,y_k\},\ \forall \ell \in \{1,\ldots,k\}}\Big\}$. When $k = 1$, this reduces to a standard discrete choice probability. If $\phi(x)$ is known, the choice probability depends only on $\mu_0(\cdot|x)$, which can be represented as a vector of normalized ranks. Consequently, any Mallows-type model that supports MLE estimation of $\mu_0(\cdot|x)$ can be embedded within our framework.

3 Two Examples of The RCPO Framework

In this section, we consider two representative examples: the Multinomial Logit (MNL) model as a utility-based choice model, and the Mallows-RMJ model as a rank-based choice model. These models

are selected for the simplicity of their choice probabilities and the tractability of MLE. For each model, we derive the corresponding objectives for both single-best and top-k feedback, demonstrating the framework's flexibility across diverse preference structures.

3.1 Multinomial Logit Model (MNL)

Discrete Choice. If we take the random shock ε to be i.i.d. Gumbel, we recover the Multinomial Logit (MNL) model (McFadden, 1972), arguably the most widely used RUM. The corresponding choice probabilities in (4) admit simple closed from expressions, given by $\mathbb{P}(y_i|S;x) = e^{\nu_{y_i}(x)} / \sum_{j=1}^{|S|} e^{\nu_{y_j}(x)}$. As such, it naturally extends the Bradley-Terry model from pairwise comparisons to multi-item choice settings. By representing the mean utility as the reward defined in (2), we have the following theorem.

Theorem 1 (MNL-PO-Discrete) *Suppose the underlying single-best choice preference distribution follows MNL, the corresponding policy optimization objective is given by:*

$$\min_{\pi_\theta} \; -\mathbb{E}_{(x,S,y_w)\sim\mathcal{D}} \log\sigma\Big(-\log \sum\nolimits_{y_i\in S\setminus\{y_w\}} \exp\big(\beta\log\tfrac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)} - \beta\log\tfrac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}\big)\Big). \tag{7}$$

As in vanilla DPO, β controls the deviation from the reference policy. When empirically solving (7), the size of each prompt's assortment S is allowed to vary, enabling the sampling of additional responses, and thus finer-grained preferences, for certain prompts to further enhance alignment. Similar objectives appear in Ziegler et al. (2019) and Chen et al. (2024), where they are treated as softmax loss functions rather than being interpreted through the lens of discrete choice model.

Top-k Ranked Choice. As shown in Feng & Tang (2023), the MNL model also extends to a simple ranked choice model. Specifically, the probability that a top-k ranked choice $\mu^k = y_1 \succ y_2 \succ \ldots \succ y_k$ is chosen out of an assortment $S \subseteq \mathcal{N}$ is given by

$$\mathbb{P}(\mu^k \mid S\,; x) = \prod\nolimits_{i=1}^{k} \frac{e^{\nu_{y_i}(x)}}{\sum_{j=i}^{k} e^{\nu_{y_j}(x)} + \sum_{y_h\in S\setminus\{y_1,\ldots,y_k\}} e^{\nu_{y_h}(x)}}. \tag{8}$$

Hence we can write down the corresponding policy optimization objective as follows.

Theorem 2 (MNL-PO-Topk) *Suppose the underlying top-k choice preference distribution follows (8), the corresponding policy optimization objective is given by:*

$$\min_{\pi_\theta} \; -\mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}} \sum\nolimits_{i=1}^{k} \log\sigma\Big(-\log \sum\nolimits_{y_j\in S\setminus\{y_1,\ldots,y_i\}} \exp\Big(\beta\log\tfrac{\pi_\theta(y_j|x)}{\pi_{\text{ref}}(y_j|x)} - \beta\log\tfrac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}\Big)\Big).$$

3.2 Mallows-RMJ Model

Discrete Choice. A notable challenge in applying Mallows-type models to (ranked) choice modeling is that the ranked choice probabilities in (6) are usually difficult to obtain, since the sum is taken over all permutations. In this regard, an exception is Feng & Tang (2022), who adopt a Mallows-type model using the Reverse Major Index (RMJ) as the distance function. They show that, unlike other Mallows-type models, the Mallows-RMJ distribution admits a closed-form expression for the choice probabilities derived from (6):

$$\mathbb{P}(y_i \mid S\,; x) = \frac{\phi(x)^{d(y_i,S)}}{1+\phi(x)+\cdots+\phi(x)^{|S|-1}}, \tag{9}$$

where $d(y_i, S) := \sum_{y_j\in S\setminus\{y_i\}} \mathbb{I}\{\mu_0^{-1}(y_i \mid x) > \mu_0^{-1}(y_j \mid x)\}$ equals the number of items in S that are ranked higher than y_i according to the global ranking $\mu_0(\cdot \mid x)$. In other words, $d(y_i, S)$ defines the *relative ranking position* of item y_i within the assortment S, so that its choice probability decays exponentially with its rank position in S.

A notable feature of this model is its exclusive dependence on *ordinal information*. For instance, when the assortment size is restricted to two, the model reduces to the classic *noisy comparison* model, in which the superior item is chosen with a fixed probability $1/(1+\phi(x))$, independent of the absolute difference between the options. This is in contrast to the Multinomial Logit (MNL) model, where choice probabilities are directly tied to the cardinal utilities of items. This reliance on relative

rankings, rather than precise utility estimates, provides the Mallows-RMJ model a form of *robustness* against model misspecification and noise in preference feedback. Such robustness may help explain the model's favorable empirical performance observed in our experiments (Section 4).

We next derive the corresponding policy optimization objectives. By normalizing the negative rank as the reward defined in (2), i.e., $-\mu_0^{-1}(y|x) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x)$, we have the following.

Theorem 3 (Mallows-RMJ-PO-Discrete) *Suppose the underlying single-best choice preference distribution follows (9), the corresponding policy optimization objective is given by:*

$$\min_{\pi_\theta} -\mathbb{E}_{(x,S,y_w)\sim\mathcal{D}}\Big[\log\phi(x)\cdot\sum\nolimits_{y_i\in S\setminus\{y_w\}}\mathbb{I}\{\beta\log\tfrac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}-\beta\log\tfrac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}<0\}\Big]. \tag{10}$$

We also derive the alignment objective for the pairwise comparison case, with details in Appendix C.

Top-k Ranked Choice. The top-k ranked choice probability under the Mallows-RMJ model is also given by a simple expression:

$$\mathbb{P}(\mu^k \mid S\,; x) = \tfrac{\psi(|S|-k,\phi(x))}{\psi(|S|,\phi(x))}\cdot\phi(x)^{d(\mu^k,S)}, \tag{11}$$

where $d(\mu^k,S) = \sum_{i=1}^{k-1}\mathbb{I}\left\{\mu_0^{-1}(y_i\mid x)>\mu_0^{-1}(y_{i+1}\mid x)\right\}(|S|-i) + \sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\mathbb{I}\left\{\mu_0^{-1}(y_k\mid x)>\mu_0^{-1}(y_j\mid x)\right\}$ and $\psi(n,\phi(x)) = \prod_{i=1}^{n}(1+\ldots+\phi(x)^{i-1})$. The corresponding policy optimization objective is thus given by the result below.

Theorem 4 (Mallows-RMJ-PO-Topk) *Suppose the underlying top-k choice preference distribution follows (11), the corresponding policy optimization objective is given by:*

$$\begin{aligned}\min_{\pi_\theta}\ -\ \mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}}\Bigg[\ \log\phi(x)\Bigg(&\sum_{i=1}^{k-1}(|S|-i)\,\mathbb{I}\Big\{\beta\log\tfrac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}-\beta\log\tfrac{\pi_\theta(y_{i+1}|x)}{\pi_{\text{ref}}(y_{i+1}|x)}<0\Big\}\\ &+\sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\mathbb{I}\Big\{\beta\log\tfrac{\pi_\theta(y_k|x)}{\pi_{\text{ref}}(y_k|x)}-\beta\log\tfrac{\pi_\theta(y_j|x)}{\pi_{\text{ref}}(y_j|x)}<0\Big\}\Bigg)\Bigg].\end{aligned} \tag{12}$$

To make Mallows-RMJ objectives practical for LLM training, we also overcome two challenges along the way: estimating the unknown dispersion parameter $\phi(x)$ and handling the step functions in the objectives that hinder optimization. For the first challenge, we follow a similar approach to Chen et al. (2025) and adapt the entropy proxy of $-\log\phi(x)$. For the second challenge, we smooth the step functions by replacing them with sigmoid approximations, which preserve the preference-based structure while yielding smoother, more informative gradients. More details are in Appendix D. The resulting objectives are summarized in Table 1.

3.3 Gradient Analysis

To gain a deeper mechanistic understanding of the RCPO framework, we analyze the gradient structure of its loss functions. We focus here on the gradient of the Mallows-RMJ-PO-Top-k objective as a representative example. Full derivations and gradient expressions for other settings are deferred to Appendix F. The gradient with respect to model parameters θ is given by:

$$\begin{aligned}&\nabla_\theta\mathcal{L}_{\text{Mallows-RMJ-PO-Top}k}(\pi_\theta)\\ &=\beta\,\mathbb{E}\Bigg[\underbrace{-\log\phi(x)}_{\substack{\text{greater weight for}\\ \text{low-dispersion prompts}}}\Bigg(\sum_{i=1}^{k-1}\underbrace{(|S|-i)}_{\substack{\text{greater weight for}\\ \text{higher ranks}}}\underbrace{\sigma\big(f_\theta(x,y_{i+1},y_i)\big)\big(1-\sigma\big(f_\theta(x,y_{i+1},y_i)\big)\big)}_{\substack{\text{greater weight when}\\ \text{rewards are similar}}}\\ &\quad\times\Bigg(\underbrace{\nabla_\theta\log\pi_\theta(y_{i+1}\mid x)}_{\text{discourage }y_{i+1}}-\underbrace{\nabla_\theta\log\pi_\theta(y_i\mid x)}_{\text{encourage }y_i}\Bigg)\\ &\quad+\sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\underbrace{\sigma\big(f_\theta(x,y_j,y_k)\big)\big(1-\sigma\big(f_\theta(x,y_j,y_k)\big)\big)}_{\text{comparison difficulty}}\Bigg(\underbrace{\nabla_\theta\log\pi_\theta(y_j\mid x)}_{\text{discourage }y_j}-\underbrace{\nabla_\theta\log\pi_\theta(y_k\mid x)}_{\text{encourage }y_k}\Bigg)\Bigg)\Bigg],\end{aligned}$$

where the expectation is taken over ranked choice triples $(x, S, \mu^k) \sim \mathcal{D}$. Intuitively, this gradient update increases the likelihood of higher-ranked responses while decreasing that of lower-ranked ones. The magnitude of the update is amplified when: (i) the prompt context exhibits low dispersion (i.e., more confident preferences); (ii) the response occurs in a higher-ranked position; and (iii) the reward estimates are close, indicating a more informative comparison. As such, these properties drive the model to sharpen distinctions near the top of the ranking and better align its outputs with fine-grained preference structures.

4 Experiments

In this section, we follow a setup similar to that in Meng et al. (2024) and evaluate our methods on widely used benchmarks. Additional experimental details are provided in Appendix G.

4.1 Experimental Setup

We adopt Llama-3-8B-Instruct and Gemma-2-9B-it as our fine-tuning bases, as both are widely used flagship instruction-tuned models that represent the state-of-the-art. We generate multiple responses to each prompt in the UltraFeedback dataset (Cui et al., 2023) and use the Skywork-Reward-V2-Llama-3.1-8B reward model to provide feedback, which achieves state-of-the-art performance on seven major reward model benchmarks at the time of writing this paper. We refer readers to Appendix G regarding details of constructing the ranking-based preference dataset.

Evaluation. We assess our fine-tuned models on two widely used instruction-following benchmarks: AlpacaEval 2.0 (Dubois et al., 2024) and Arena-Hard-v0.1 (Li et al., 2024). AlpacaEval 2.0 comprises 805 questions drawn from distinct datasets. Performance is measured by the win rate (WR) of model outputs against reference answers generated by GPT-4-Turbo. We also report the length-controlled win rate (LC), which adjusts WR to control for output length. Arena-Hard-v0.1 consists of 500 well-defined technical problem-solving prompts and evaluates models using WR against GPT-4-0314. Previous works (Meng et al., 2024) have demonstrated that Arena-Hard-v0.1 achieves stronger model separability than AlpacaEval 2.0. For both benchmarks, we use GPT-4.1-mini as the judge, replacing the default GPT-4-Turbo due to its improvements (OpenAI, 2025). To enhance cross-judge robustness, we additionally employ GPT-5-mini as the judge on Arena-Hard-v0.1. The results are in Appendix G.3. We defer hyperparameter tuning details to Appendix G.1 and decoding details to Appendix G.2.

4.2 Results

Alignment performance. All eight choice-based methods—three existing and five newly proposed RCPO variants—clearly outperform other preference optimization baselines across all evaluation metrics. Notably, the best-performing RCPO method, Mallows-RMJ-PO-Top2, surpasses the strongest non-RCPO baseline, IPO, by 4.00 points on AlpacaEval LC, 19.5 points on AlpacaEval WR, and 6.2 points on Arena-Hard WR. These gains demonstrate the strong potential of choice modeling as a principled framework for aligning LLMs with human preferences.

Impact of feedback structure. Fixing the reward function as (2), we find that training on top-2 feedback generally leads to better performance than top-1. This reflects the benefit of richer feedback structure. On the other hand, we envision that there is an implicit trade-off between informativeness of feedback and the intrinsic noise/error in the data. Therefore, the performance may not be always increasing in the length of feedback. That is why we stop at Top-2 before going even further, such as the full rankings.

Impact of choice models. The selection of the choice model can have a substantial impact on performance. Fixing the reward function as (2), Mallows-RMJ performs strongly across feedback types. In the pairwise setting, it outperforms all baselines on AlpacaEval WR and Arena-Hard WR, even surpassing MNL models trained on richer top-2 feedback. Its performance further improves with discrete or top-2 choice data.

Taken together, our results highlight the promising potential of leveraging broader feedback structures, when paired with appropriate choice models, to achieve more effective alignment.

Table 2: Evaluation Results for Llama-3-8B-Instruct.

	AlpacaEval 2		Arena-Hard
Method	**LC (%)**	**WR (%)**	**WR (%)**
Base Model	24.76 (0.42)	24.40 (1.44)	23.6 (21.9, 25.4)
CPO (Xu et al., 2024)	32.25 (0.31)	34.18 (1.59)	31.3 (29.5, 33.0)
IPO (Azar et al., 2024)	37.95 (0.33)	33.51 (1.63)	31.0 (29.2, 32.6)
ORPO (Hong et al., 2024)	35.01 (0.39)	28.72 (1.52)	27.5 (25.5, 29.3)
RRHF (Yuan et al., 2023)	31.04 (0.21)	25.36 (1.47)	27.1 (25.3, 28.9)
SLiC-HF (Zhao et al., 2023)	31.04 (0.21)	25.36 (1.47)	26.9 (25.2, 28.9)
KTO (Ethayarajh et al., 2024)	32.55 (0.35)	29.70 (1.54)	25.8 (24.0, 27.7)
DPO (Rafailov et al., 2023)	41.24 (0.35)	40.24 (1.66)	32.6 (30.6, 34.7)
R-DPO (Park et al., 2024)	39.88 (0.34)	38.01 (1.63)	32.3 (30.1, 34.3)
SimPO (Meng et al., 2024)	**44.15** (0.25)	38.84 (1.58)	33.5 (31.3, 35.8)
Mallows-RMJ-PO-Pairwise	39.33 (0.28)	48.71 (1.67)	36.5 (34.3, 38.6)
MNL-PO-Discrete	41.33 (0.29)	48.08 (1.68)	35.6 (33.6, 37.4)
Mallows-RMJ-PO-Discrete	39.19 (0.28)	51.17 (1.67)	36.3 (34.6, 37.9)
MNL-PO-Top2	40.12 (0.22)	47.69 (1.67)	35.8 (33.2, 38.2)
Mallows-RMJ-PO-Top2	41.95 (0.26)	**53.01** (1.68)	**37.2** (35.0, 39.4)

Notes: The last eight rows correspond to preference optimization methods within the RCPO framework. Among these, the first three (DPO, R-DPO, SimPO) are from prior work, and the remaining five are new variants in this paper. Standard errors (in parentheses) follow AlpacaEval 2 metrics, and 95% confidence interval (in parentheses) follow Arena-Hard WR.

Robustness check on Gemma-2-9B-it. As shown in Table 3, the Mallows-RMJ-PO-Top2 achieves the best results on AlpacaEval WR and Arena-Hard WR while remaining competitive on LC, highlighting strong generalization across base models.

Table 3: Evaluation Results for Gemma-2-9B-it.

	AlpacaEval 2		Arena-Hard
Method	**LC (%)**	**WR (%)**	**WR (%)**
Base Model	46.74 (0.15)	31.81 (1.58)	43.3 (41.3, 45.7)
SimPO (Meng et al., 2024)	54.11 (0.17)	47.23 (1.68)	57.4 (55.3, 59.6)
DPO (Rafailov et al., 2023)	**58.01** (0.18)	56.13 (1.66)	59.9 (57.6, 62.2)
Mallows-RMJ-PO-Top2	55.64 (0.11)	**59.82** (1.65)	**60.9** (58.9, 62.6)

Notes: Standard errors (in parentheses) follow AlpacaEval 2 metrics, and 95% CIs (in parentheses) follow Arena-Hard WR.

5 Conclusion

In this work, we propose Ranked Choice Preference Optimization (RCPO), a general framework that connects preference optimization with choice model estimation. By leveraging maximum likelihood, RCPO unifies pairwise, single-best, and top-k preference data within a principled formulation. Examples of both utility- and rank-based choice models, together with empirical evaluations, demonstrate that RCPO preserves richer feedback and improves alignment over pairwise-only methods. We hope this work provides a foundation for integrating more advanced choice models into LLM alignment and inspires future exploration of richer preference signals.

Ethics statement This work adheres to the ICLR Code of Ethics. All authors have read and explicitly acknowledged compliance with the Code of Ethics during the submission process. Our study does not involve human subjects, personal or sensitive data, or any procedures that would raise ethical concerns. Therefore, we believe that this work poses no ethical risks.

Reproducibility statement We have made significant efforts to ensure reproducibility. Complete proofs of all theoretical results and detailed derivations of the gradient analysis presented in Section 3 are provided in Appendix E and Appendix F, respectively. The full experimental details, including the computing environment, dataset construction process, the use of open-source models, hyperparameter settings, and evaluation protocols, are provided in the Appendix G. In addition, we provide the codes and training scripts in the supplementary materials. Together, these materials allow independent researchers to fully reproduce our results.

REFERENCES

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

Aydın Alptekinoğlu and John H Semple. The exponomial choice model: A new alternative for assortment and price optimization. *Operations Research*, 64(1):79–93, 2016.

Simon P Anderson, Andre De Palma, and Jacques-Francois Thisse. *Discrete choice theory of product differentiation*. MIT press, 1992.

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

Gerardo Berbeglia, Agustín Garassino, and Gustavo Vulcano. A comparative empirical study of discrete choice models in retail operations. *Management Science*, 68(6):4005–4023, 2022.

Jose Blanchet, Guillermo Gallego, and Vineet Goyal. A markov chain approximation to choice modeling. *Operations Research*, 64(4):886–905, 2016.

Abraham Bookstein, Vladimir A Kulyukin, and Timo Raita. Generalized hamming distance. *Information Retrieval*, 5(4):353–375, 2002.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Sebastian Bruch, Masrour Zoghi, Michael Bendersky, and Marc Najork. Revisiting approximate metric optimization in the age of deep neural networks. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pp. 1241–1244, 2019.

Haoxian Chen, Hanyang Zhao, Henry Lam, David Yao, and Wenpin Tang. Mallowspo: Fine-tune your llm with preference dispersions. In *The Thirteenth International Conference on Learning Representations*, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yuxin Chen, Junfei Tan, An Zhang, Zhengyi Yang, Leheng Sheng, Enzhi Zhang, Xiang Wang, and Tat-Seng Chua. On softmax direct preference optimization for recommendation. *Advances in Neural Information Processing Systems*, 37:27463–27489, 2024.

Vincent Conitzer, Rachel Freedman, Jobst Heitzig, Wesley H Holliday, Bob M Jacobs, Nathan Lambert, Milan Mossé, Eric Pacuit, Stuart Russell, Hailey Schoelkopf, et al. Social choice should guide ai alignment in dealing with diverse human feedback. *arXiv preprint arXiv:2404.10271*, 2024.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377*, 2023.

Carlos Daganzo. *Multinomial probit: the theory and its application to demand forecasting*. Elsevier, 2014.

Jessica Dai and Eve Fleisig. Mapping social choice theory to rlhf. *arXiv preprint arXiv:2404.13038*, 2024.

Persi Diaconis and Ronald L Graham. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 39(2):262–268, 1977.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

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

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

Vivek F Farias, Srikanth Jagabathula, and Devavrat Shah. A nonparametric approach to modeling choice with limited data. *Management science*, 59(2):305–322, 2013.

Yifan Feng and Yuxuan Tang. On a mallows-type model for (ranked) choices. *Advances in Neural Information Processing Systems*, 35:3052–3065, 2022.

Yifan Feng and Yuxuan Tang. A mallows-type model for preference learning from (ranked) choices. *Available at SSRN 4539900*, 2023.

Michael A Fligner and Joseph S Verducci. Distance based ranking models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):359–369, 1986.

Guillermo Gallego, Richard Ratliff, and Sergey Shebalov. A general attraction model and sales-based linear program for network revenue management under customer choice. *Operations Research*, 63 (1):212–232, 2015.

Guillermo Gallego, Huseyin Topaloglu, et al. *Revenue management and pricing analytics*, volume 209. Springer, 2019.

Luise Ge, Daniel Halpern, Evi Micha, Ariel D Procaccia, Itai Shapira, Yevgeniy Vorobeychik, and Junlin Wu. Axioms for ai alignment from human feedback. *Advances in Neural Information Processing Systems*, 37:80439–80465, 2024.

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

José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *NIPS*, volume 27, pp. 918–926, 2014.

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

Saffron Huang, Divya Siddarth, Liane Lovitt, Thomas I Liao, Esin Durmus, Alex Tamkin, and Deep Ganguli. Collective constitutional ai: Aligning a language model with public input. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, pp. 1395–1417, 2024.

Ekhine Irurozki, Borja Calvo, and Jose A Lozano. Sampling and learning mallows and generalized mallows models under the cayley distance. *Methodology and Computing in Applied Probability*, 20(1):1–35, 2018.

Srikanth Jagabathula and Ashwin Venkataraman. *Nonparametric Estimation of Choice Models*, pp. 177–209. Springer International Publishing, Cham, 2022. ISBN 978-3-031-01926-5. doi: 10.1007/978-3-031-01926-5_8. URL https://doi.org/10.1007/978-3-031-01926-5_8.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023.

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

Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, et al. Lipo: Listwise preference optimization through learning-to-rank. *arXiv preprint arXiv:2402.01878*, 2024.

David JC MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.

Colin L Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.

Daniel McFadden. Conditional logit analysis of qualitative choice behavior. 1972.

Daniel McFadden. Econometric models for probabilistic choice among products. *Journal of Business*, pp. S13–S29, 1980.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Abhilash Mishra. Ai alignment and social choice: Fundamental limitations and policy implications. *arXiv preprint arXiv:2310.16048*, 2023.

Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*, 2023.

OpenAI. Introducing GPT-4.1 in the API. https://openai.com/blog/introducing-gpt-4-1-in-the-api, 2025. Accessed: 2025-07-30.

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

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. *arXiv preprint arXiv:2403.19159*, 2024.

Mahendra Prasad. Social choice and the value alignment problem. *Artificial intelligence safety and security*, pp. 291–314, 2018.

Tao Qin, Tie-Yan Liu, and Hang Li. A general approximation framework for direct optimization of information retrieval measures. *Information retrieval*, 13(4):375–397, 2010.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18990–18998, 2024.

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

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

Louis L Thurstone. A law of comparative judgment. In *Scaling*, pp. 81–92. Routledge, 2017.

Kenneth E Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.

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

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. *arXiv preprint arXiv:2309.16240*, 2023.

Genta Indra Winata, Hanyang Zhao, Anirban Das, Wenpin Tang, David D Yao, Shi-Xiong Zhang, and Sambit Sahu. Preference tuning with human feedback on language, speech, and vision tasks: A survey. *Journal of Artificial Intelligence Research*, 82:2595–2661, 2025.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*, 2024.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback. *Advances in Neural Information Processing Systems*, 36:10935–10950, 2023.

Hanyang Zhao, Genta Indra Winata, Anirban Das, Shi-Xiong Zhang, David D Yao, Wenpin Tang, and Sambit Sahu. Rainbowpo: A unified framework for combining improvements in preference optimization. *arXiv preprint arXiv:2410.04203*, 2024.

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

Tan Zhi-Xuan, Micah Carroll, Matija Franklin, and Hal Ashton. Beyond preferences in ai alignment. *Philosophical Studies*, pp. 1–51, 2024.

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

APPENDIX

A THE USE OF LARGE LANGUAGE MODELS (LLMS)

Large Language Models are used to assist in editing and polishing the writing. The authors take full responsibility for all content of the paper, and acknowledge that LLMs are not eligible for authorship.

B RELATED WORK

Preference optimization. DPO's success has attracted significant research attention in the LLM alignment community, yielding numerous variants with alternative objectives. These include formulations based on ranking objectives (Yuan et al., 2023; Song et al., 2024; Liu et al., 2024), pairwise comparisons employing other preference models (Chen et al., 2025), other implicit reward formulations (Wang et al., 2023; Meng et al., 2024; Gupta et al., 2025), and approaches leveraging binary feedback on individual prompt-response pairs (Ethayarajh et al., 2024). Distinct from prior studies, we investigate novel forms of preference feedback that enable alignment methods to directly leverage richer choice-based signals.

Discrete choice modeling. Research in marketing, economics, and operations research has studied various specifications of discrete choice models, including the multinomial logit model (McFadden, 1972), the general attraction model (Gallego et al., 2015), the Markov chain choice model (Blanchet et al., 2016), rank-based choice models (Farias et al., 2013), among others. These models play a critical role in informing key operational decisions such as inventory management, assortment planning, pricing, and matching optimization. For comprehensive overviews, we refer readers to Train (2009), Gallego et al. (2019), and Berbeglia et al. (2022). To the best of our knowledge, we are the first to apply discrete choice model theory in LLM alignment, enabling principled use of richer preference feedback beyond pairwise comparisons.

Social choice and AI alignment. Social choice theory is a field of study that deals with the aggregation of individual preferences to form a collective decision. Current approaches to LLM alignment involves the experimental studies (Huang et al., 2024), conceptual frameworks (Prasad, 2018; Mishra, 2023; Dai & Fleisig, 2024; Conitzer et al., 2024; Zhi-Xuan et al., 2024), and axiomatic frameworks (Ge et al., 2024). In contrast to the standard social choice setting (Dai & Fleisig, 2024), where voters provide full rankings over all alternatives, we focus on aggregating ranked choice preferences.

C MALLOWS-RMJ IN PAIRWISE SETUP

Pairwise Setup. When this model is restricted to the pairwise comparison setting, the preference probability simplifies to

$$\mathbb{P}\left(y_w \succ y_l \,; x\right) = \frac{\phi(x)^{\mathbb{I}\{\mu_0^{-1}(y_w|x) > \mu_0^{-1}(y_l|x)\}}}{1+\phi(x)}. \tag{13}$$

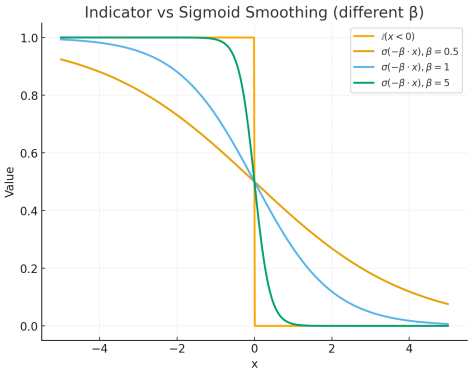
This formulation aligns with the widely studied noisy pairwise comparison models in the computer science literature, where only two items are compared at a time, and the preferred one is selected with a fixed probability that does not depend on the specific pair. The pairwise probability in (13) leads to our following optimization objective.

Theorem 5 (Mallows-RMJ-PO-Pairwise) *Suppose the underlying pairwise preference distribution follows (13), the corresponding policy optimization objective is given by:*

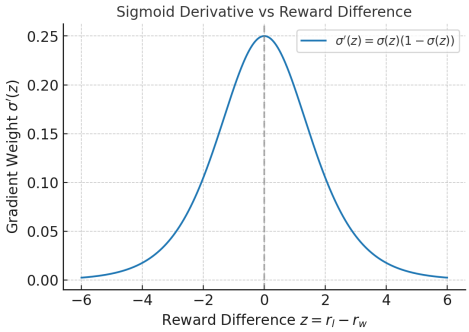
$$\min_{\pi_\theta} -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\log\phi(x)\cdot\mathbb{I}\{\beta\log\tfrac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\tfrac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} < 0\}\Big]. \tag{14}$$

D PRACTICAL SCHEME OF MALLOWS-RMJ METHODS

To make Mallows-RMJ-based objectives practical for LLM training, we address two key challenges: estimating the dispersion parameter $\phi(x)$ and stabilizing optimization via smooth approximations.



(a) Indicator with its sigmoid approximation.



(b) Derivative of sigmoid function.

A Token-level Entropy-based Proxy of $-\log\phi(x)$ for Any Mallows-Type Models. Chen et al. (2025) studies using Mallows-θ and Mallows-ϕ model to conduct alignment on pairwise preference data. They propose a direct approach to estimate the dispersion parameter $\phi(x)$ without any pretraining or learning.

The idea is to qualitatively relate $\phi(x)$ to the empirical output distribution of the LLM. Intuitively, when $-\log(\phi(x))$ is large, preferences are highly concentrated and the next-token distribution collapses to a point mass, whereas when $-\log(\phi(x))$ approaches zero, the distribution becomes uniform. On the other hand, Shannon's entropy $H(X) = 0$ when X is a point mass, and $H(X) = \log n$ when X is uniform on n points. Motivated by this observation, they propose:

$$-\log\left(H(\pi(\cdot \mid x))/\log n\right), \tag{15}$$

as a proxy to $-\log\phi(x)$, where $\pi(\cdot \mid x)$ can be either the pretrained LM model or the SFT model. Furthermore, they approximate the entropy term in (15) via a realization of a sequence of $N = \max(|Y_w|, |Y_l|)$ tokens $\{Y_w^i, Y_l^i\}_{i=1,\ldots,N}$ given the prompt X:

$$H(\pi(\cdot \mid X)) \approx \frac{1}{2}\sum_{i=1}^{N-1}\left[H(Y^{i+1} \mid Y^i = Y_w^i) + H(Y^{i+1} \mid Y^i = Y_l^i)\right], \tag{16}$$

which can be easily computed by the logits of the model given the output data. In this case, $n = V^N$, where V is the token size. This is also related to the predictive entropy (Hernández-Lobato et al., 2014; MacKay, 1992) of the next-token predictions.

Finally, the authors validate that this entropy-based estimator closely matches the true dispersion in a synthetic experiment setup.

In this paper, we adopt Chen et al. (2025) to approximate dispersion using Shannon entropy. While their method assumes pairwise comparisons, we extend it to multiple responses. Specifically, for a prompt X with response set $S = \{Y_1, \ldots, Y_{|S|}\}$, we approximate $-\log(\phi(X))$ as

$$-\log\left(\frac{1}{|S|\log n}\sum\nolimits_{i=1}^{|S|}\sum\nolimits_{j=1}^{N-1} H(Y^{j+1} \mid Y^j = Y_i^j)\right),$$

where $H(\cdot|\cdot)$ denotes the conditional Shannon entropy, which can be directly computed from the model's logits. Here, Y_i^j is the j_{th} token of response i, $N = \max(|Y_1|, \ldots, |Y_{|S|}|)$, and $n = V^N$ with vocabulary size V.

Sigmoid-Smoothed Objectives. The objectives in (10), (12), and (14) cannot be directly optimized with gradient-based methods, since they involve step functions that are either discontinuous or flat almost everywhere, yielding zero gradients for most values of θ. To make these objectives amenable to efficient optimization with preference data, we replace the indicator functions with a sigmoid approximation. This smoothing technique is widely adopted in the machine learning literature (see, e.g., (Qin et al., 2010; Bruch et al., 2019)). Specifically, we approximate $\mathbb{I}\{x < 0\}$ using $\sigma(-\beta x)$, where β serves as a hyperparameter that controls the tightness of the approximation. An illustration is provided in Figure 2a.

This design brings two significant benefits:

(i) *Computational benefits*: The original indicator-based objectives are discontinuous at zero, making gradient-based training unstable. The sigmoid function smooths these objectives, facilitating efficient optimization.

(ii) *Soft and robust penalties*: The sigmoid function offers a continuous, differentiable alternative to the indicator, yielding a "soft" loss that reflects the magnitude of preference violations rather than just their direction. For example, in the loss function (10), under the indicator function, only the relative ordering between $\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}$ and $\beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}$ matters—i.e., the loss is zero as long as the preferred response scores higher. In contrast, the sigmoid penalty decreases smoothly as the score gap widens, encouraging the model to not only rank preferred responses above dispreferred ones, but to do so confidently. For instance, even if the inequality holds, a small margin will still incur non-trivial loss, while a large margin will yield a smaller loss. This is conceptually similar to incorporating a margin term in the loss function, as seen in prior works (Zhao et al., 2023; Meng et al., 2024; Azar et al., 2024; Hong et al., 2024).

Overall, this smoothing approach allows us to retain the structure of preference-based training while enabling more stable and informative gradient signals during optimization. The final objectives are summarized in Table 1.

E PROOFS

Proof of Theorem 1. The *Multinomial Logit* (MNL) model (McFadden, 1972) is one of the most widely used utility-based discrete choice models. It assumes that each alternative $y_i \in S$ is associated with a latent utility $u_{y_i}(x) = \nu_{y_i}(x) + \epsilon_i$, where $\nu_{y_i}(x)$ is a deterministic component and ϵ_i follows an independent Gumbel distribution. Under this assumption, the choice probability of selecting alternative y_w from assortment S takes the closed form

$$\mathbb{P}(y_w \mid S\,; x) = \frac{\exp(\nu_{y_w}(x))}{\sum_{y_i \in S} \exp(\nu_{y_i}(x))}.$$

By identifying the deterministic utility $\nu_y(x)$ as the reward function defined in (2), the normalization constant $Z(x)$ cancels out, and we are left with:

$$\mathbb{P}_{\pi_\theta}(y_w \mid S\,; x) = \frac{e^{\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}}}{\sum_{y_i \in S} e^{\beta \log \frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}}}.$$

Maximizing the likelihood yields to the following objective:

$$\min_{\pi_\theta} \; -\mathbb{E}_{(x,S,y_w)\sim\mathcal{D}} \left[\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log \sum_{y_i \in S} e^{\beta \log \frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}} \right],$$

which is equivalent to the objective in (7). □

Proof of Theorem 2. Following Feng & Tang (2023), the probability that an individual selects a top-k choice $\mu^k = y_1 \succ y_2 \succ \ldots \succ y_k$ out of an assortment $S \subseteq \mathcal{N}$ is

$$\mathbb{P}(\mu^k \mid S\,; x) = \prod_{i=1}^{k} \frac{e^{\nu_{y_i}(x)}}{\sum_{j=i}^{k} e^{\nu_{y_j}(x)} + \sum_{y_h \in S \setminus \{y_1,\ldots,y_k\}} e^{\nu_{y_h}(x)}}.$$

By identifying the deterministic utility $\nu_y(x)$ as the reward function defined in (2), we have the optimal RLHF policy satisfies

$$\mathbb{P}_{\pi_\theta}(\mu^k \mid S\,; x) = \prod_{i=1}^{k} \frac{e^{\beta \log \frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}}}{\sum_{j=i}^{k} e^{\beta \log \frac{\pi_\theta(y_j|x)}{\pi_{\text{ref}}(y_j|x)}} + \sum_{y_h \in S \setminus \{y_1,\ldots,y_k\}} e^{\beta \log \frac{\pi_\theta(y_h|x)}{\pi_{\text{ref}}(y_h|x)}}}.$$

To maximize the likelihood estimation, our objective becomes:

$$\min_{\pi_\theta} -\mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}}\left[\sum_{i=1}^{k}\beta\log\frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}-\sum_{i=1}^{k}\log\Big(\sum_{j=i}^{k}e^{\beta\log\frac{\pi_\theta(y_j|x)}{\pi_{\text{ref}}(y_j|x)}}+\sum_{y_h\in S\setminus\{y_1,\ldots,y_k\}}e^{\beta\log\frac{\pi_\theta(y_h|x)}{\pi_{\text{ref}}(y_h|x)}}\Big)\right],$$

which is equivalent to the objective in Theorem 2. □

Proof of Theorem 3. We consider optimizing the following objective:

$$\max_{\pi_\theta}\mathbb{E}_{x\sim\mathcal{D}}\Big[\mathbb{E}_{y\sim\pi_\theta(y|x)}\big[-\mu_0^{-1}(y\mid x)\big]-\beta\,\mathrm{KL}\big(\pi_\theta(\cdot\mid x)\,\|\,\pi_{\text{ref}}(\cdot\mid x)\big)\Big], \tag{17}$$

As shown in section A.1 of Rafailov et al. (2023), the optimum of such a KL-constrained reward maximization objective has the form of

$$\pi_\theta(y\mid x)=\tfrac{1}{Z(x)}\pi_{\text{ref}}(y\mid x)\exp\left(-\tfrac{\mu_0^{-1}(y|x)}{\beta}\right),$$

where $Z(x)=\sum_y\pi_{\text{ref}}(y\mid x)\,\exp(-\frac{1}{\beta}\,\mu_0^{-1}(y\mid x))$ is the partition function. By moving terms, we have

$$-\mu_0^{-1}(y\mid x)=\beta\log\tfrac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}+\beta\log Z(x). \tag{18}$$

Combining (9) and (18), we have the optimal RLHF policy $\pi_\theta(\cdot\mid x)$ for (17) satisfies

$$\mathbb{P}_{\pi_\theta}(y_w\mid S\,;x)=\frac{\phi(x)^{\sum_{y_i\in S\setminus\{y_w\}}\mathbb{I}\{-\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}-\beta\log Z(x)>-\beta\log\frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}-\beta\log Z(x)\}}}{1+\phi(x)+\cdots+\phi(x)^{|S|-1}}.$$

Maximizing the likelihood leads to the following objective:

$$\begin{aligned}&\min_{\pi_\theta}-\mathbb{E}_{(x,S,y_w)\sim\mathcal{D}}\left[\log\frac{\phi(x)^{\sum_{y_i\in S\setminus\{y_w\}}\mathbb{I}\{-\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}>-\beta\log\frac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}\}}}{1+\phi(x)+\cdots+\phi(x)^{|S|-1}}\right]\\ =&\min_{\pi_\theta}-\mathbb{E}_{(x,S,y_w)\sim\mathcal{D}}\left[\sum_{y_i\in S\setminus\{y_w\}}\mathbb{I}\{-\beta\log\tfrac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}>-\beta\log\tfrac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}\}\log\phi(x)-C(x)\right],\end{aligned}$$

where $C(x)=\log(1+\phi(x)+\cdots+\phi(x)^{|S|-1})$ is constant with respect to the policy and thus does not affect the optimal solution. This formulation is equivalent to the objective in (10). □

Proof of Theorem 4. Following (11) and a similar discussion as in the proof of Theorem 3, we have the optimal RLHF policy $\pi_\theta(\cdot\mid x)$ for (17) satisfies

$$\mathbb{P}_{\pi_\theta}(\mu^k\mid S\,;x)=\tfrac{\psi(|S|-k,\phi(x))}{\psi(|S|,\phi(x))}\cdot\phi(x)^{d_{\pi_\theta}(\mu^k,S)},$$

where the exponent term $d(\mu^k,S)$ is

$$\begin{aligned}d_{\pi_\theta}(\mu^k,S)=&\sum_{i=1}^{k-1}\mathbb{I}\left\{-\beta\log\tfrac{\pi_\theta(y_i|x)}{\pi_{\text{ref}}(y_i|x)}>-\beta\log\tfrac{\pi_\theta(y_{i+1}|x)}{\pi_{\text{ref}}(y_{i+1}|x)}\right\}\cdot(|S|-i)+\\ &\sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\mathbb{I}\left\{-\beta\log\tfrac{\pi_\theta(y_k|x)}{\pi_{\text{ref}}(y_k|x)}>-\beta\log\tfrac{\pi_\theta(y_j|x)}{\pi_{\text{ref}}(y_j|x)}\right\}.\end{aligned}$$

Maximizing the likelihood leads to the following objective:

$$\begin{aligned}&\min_{\pi_\theta}-\mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}}\left[\log(\tfrac{\psi(|S|-k,\phi(x))}{\psi(|S|,\phi(x))}\cdot\phi(x)^{d_{\pi_\theta}(\mu^k,S)})\right]\\ =&\min_{\pi_\theta}-\mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}}\left[\log\tfrac{\psi(|S|-k,\phi(x))}{\psi(|S|,\phi(x))}+d_{\pi_\theta}(\mu^k,S)\log\phi(x)\right],\end{aligned}$$

where $\log\frac{\psi(|S|-k,\phi(x))}{\psi(|S|,\phi(x))}$ is constant with respect to the policy and thus does not affect the optimal solution. This formulation is equivalent to the objective in (12). □

Proof of Theorem 5. Theorem 5 is a special case of Theorem 3 with $S=\{y_w,y_l\}$. □

F GRADIENT ANALYSIS

Let $f_\theta(x, y_1, y_2) := \beta \log \frac{\pi_\theta(y_1|x)}{\pi_{\text{ref}}(y_1|x)} - \beta \log \frac{\pi_\theta(y_2|x)}{\pi_{\text{ref}}(y_2|x)}$ for shorthand notation. We next provide an analysis of various preference optimization models to shed light on their training procedures.

F.1 MNL

In this section, we derive the gradients of DPO, MNL-PO, and RankedMNL-PO, and then compare their update mechanisms.

F.1.1 DPO

Recap the DPO gradient below:

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta) = -\beta\, \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Bigg[\underbrace{\sigma\big(f_\theta(x, y_l, y_w)\big)}_{\text{higher weight when reward estimate is wrong}} \Big(\underbrace{\nabla_\theta \log \pi_\theta(y_w \mid x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi_\theta(y_l \mid x)}_{\text{decrease likelihood of } y_l} \Big) \Bigg].$$

F.1.2 MNL-PO-DISCRETE

The gradient of $\mathcal{L}_{\text{MNL-PO-Discrete}}$ with respect to parameters θ takes the following formulation:

$$\nabla_\theta \mathcal{L}_{\text{MNL-PO-Discrete}}(\pi_\theta) =$$
$$-\beta \mathbb{E}\Bigg[\underbrace{\sigma\Bigg(\log \sum_{y_i \in S\setminus\{y_w\}} \exp(f_\theta(x, y_i, y_w))\Bigg)}_{\text{higher weight when reward deviates from preference}} \cdot \Bigg[\nabla_\theta \log \pi_\theta(y_w|x) - \sum_{y_i \in S\setminus\{y_w\}} \underbrace{\frac{\nabla_\theta \log \pi_\theta(y_i|x)}{\sum_{y_j \in S\setminus\{y_w\}} \exp(f_\theta(x, y_j, y_i))}}_{\text{higher weight when reward is larger}} \Bigg] \Bigg],$$

where the expectation is with respect to $(x, S, y_w) \sim \mathcal{D}$. As MNL-PO-Discrete is a special case of MNL-PO-Topk with ranking length $k = 1$, we defer its derivation to the next section.

F.1.3 MNL-PO-TOPK

The gradient of $\mathcal{L}_{\text{MNL-PO-Topk}}$ with respect to parameters θ takes the following formulation:

$$\nabla_\theta \mathcal{L}_{\text{MNL-PO-Topk}}(\pi_\theta) =$$
$$-\beta \mathbb{E}\Bigg[\underbrace{\sum_{i=1}^{k}}_{\substack{\text{ranked preference} \\ y_1 \succ \ldots \succ y_i | S\,; x}} \underbrace{\sigma\Bigg(\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\Bigg)}_{\text{higher weight when reward deviates from preference}} \cdot \Bigg[\nabla_\theta \log \pi_\theta(y_i|x) - \sum_{\substack{y_j \in \\ S\setminus\{y_1,\ldots,y_i\}}} \underbrace{\frac{\nabla_\theta \log \pi_\theta(y_j|x)}{\sum_{\substack{y_{j'} \in \\ S\setminus\{y_1,\ldots,y_i\}}} \exp(f_\theta(x, y_{j'}, y_j))}}_{\text{higher weight when reward is larger}} \Bigg] \Bigg],$$

where the expectation is with respect to $(x, S, \mu^k) \sim \mathcal{D}$.

The gradient of the MNL-PO-Topk loss increases the likelihood of the chosen responses while decreasing the likelihood of all unchosen responses. Specifically, each preference relation $(y_1 \succ \ldots \succ y_i \mid S\,; x)$ is weighted by the degree to which the implicit reward deviates from the observed preference. Furthermore, MNL-PO-Topk differentiates among the gradients of unchosen responses: the gradient for an unchosen response y_j is scaled by $\frac{1}{\sum_{y_{j'} \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_{j'}, y_j))} = \frac{\exp(\beta \log \frac{\pi_\theta(y_j|x)}{\pi_{\text{ref}}(y_j|x)})}{\sum_{y_{j'} \in S\setminus\{y_1,\ldots,y_i\}} \exp(\beta \log \frac{\pi_\theta(y'_j|x)}{\pi_{\text{ref}}(y'_j|x)})}$. This factor captures the relative reward of y_j compared with the other unchosen responses.

Derivation. Recall that $f_\theta(x, y_1, y_2) = \beta \log \frac{\pi_\theta(y_1|x)}{\pi_{\text{ref}}(y_1|x)} - \beta \log \frac{\pi_\theta(y_2|x)}{\pi_{\text{ref}}(y_2|x)}$. The MNL-PO-Topk loss takes the following form:

$$\mathcal{L}_{\text{MNL-PO-Topk}}(\pi_\theta) = -\mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}} \left[\sum_{i=1}^{k} \log \sigma\Big(-\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i)) \Big) \right]$$

The gradient of $f_\theta(x, y_1, y_2)$ can be formulated as:

$$\nabla_\theta f_\theta(x, y_1, y_2) = \beta(\nabla_\theta \log \pi_\theta(y_1|x) - \nabla_\theta \log \pi_\theta(y_2|x)) \tag{19}$$

Using properties of the sigmoid function that $\sigma'(x) = \sigma(x)(1 - \sigma(x)) = \sigma(x)\sigma(-x)$ and thus $((\log \sigma(x))' = \frac{1}{\sigma(x)} \times \sigma(x)\sigma(-x) = \sigma(-x)$, we have:

$$\begin{aligned}
&\nabla_\theta \mathcal{L}_{\mathrm{MNL-PO-Topk}}(\pi_\theta) \\
&= -\mathbb{E}\left[\sum_{i=1}^{k} \nabla_\theta \log \sigma\left(-\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\right)\right] \\
&= \mathbb{E}\left[\sum_{i=1}^{k} \sigma\left(\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\right) \cdot \nabla_\theta \log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\right] \\
&\qquad\qquad ((\log\sigma(x))' = \sigma(-x)) \\
&= \mathbb{E}\left[\sum_{i=1}^{k} \sigma\left(\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\right) \cdot \frac{\sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x,y_j,y_i)) \cdot \nabla_\theta f_\theta(x,y_j,y_i)}{\sum_{y'_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x,y_{j'},y_i))}\right] \\
&= -\beta\mathbb{E}\left[\sum_{i=1}^{k} \sigma\left(\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\right) \cdot \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \frac{\nabla_\theta \log \pi_\theta(y_i|x) - \nabla_\theta \log \pi_\theta(y_j|x)}{\sum_{y_{j'} \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x,y_{j'},y_i) - f_\theta(x,y_j,y_i))}\right] \\
&\qquad\qquad \text{(Eq. (19))} \\
&= -\beta\mathbb{E}\left[\sum_{i=1}^{k} \sigma\left(\log \sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x, y_j, y_i))\right) \cdot \left[\nabla_\theta \log \pi_\theta(y_i|x) - \sum_{\substack{y_j \in \\ S\setminus\{y_1,\ldots,y_i\}}} \frac{\nabla_\theta \log \pi_\theta(y_j|x)}{\sum_{y_{j'} \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x,y_{j'},y_j))}\right]\right]
\end{aligned}$$

The last equation is because:

$$\sum_{\substack{y_j \in \\ S\setminus\{y_1,\ldots,y_i\}}} \frac{1}{\sum_{\substack{y'_j \in \\ S\setminus\{y_1,\ldots,y_i\}}} \exp(f_\theta(x,y_{j'},y_i) - f_\theta(x,y_j,y_i))} = \frac{\sum_{y_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x,y_j,y_i))}{\sum_{y'_j \in S\setminus\{y_1,\ldots,y_i\}} \exp(f_\theta(x,y'_j,y_i))} = 1, \forall\, i.$$

F.1.4 Comparison of Gradient Updates

DPO updates the model by increasing the likelihood of the preferred response y_w and decreasing that of the dispreferred response y_l. The update weight is larger when the model's reward estimate disagrees with the preference. This ensures that learning focuses on correcting mistakes, especially in cases where the model is misaligned with the observed preference.

MNL-PO-Discrete compares the preferred response y_w against all other alternatives $y_i \in S \setminus \{y_w\}$. The update weight grows when the reward deviates from the preference, and it is further adjusted according to the relative reward magnitudes across the choice set. As a result, the gradient not only enforces the winner against individual competitors but also reflects the overall reward distribution of the choice set.

MNL-PO-Topk extends this to ranked preferences $y_1 \succ \ldots \succ y_k \mid S$. The gradient sequentially enforces each ranking position by comparing $y_i,\ i = 1, \ldots, k$ against the remaining alternatives. Similar to MNL-PO-Discrete, the update is stronger when the reward diverges from the observed preference and when the competitor's reward is larger. This design enables the model to learn the entire preference ranking rather than only the top choice, sharpening distinctions across multiple ranking positions.

F.2 Mallows-RMJ

In this section, we derive the gradients of Mallows-RMJ-PO-Pairwise, Mallows-RMJ-PO-Discrete, and Mallows-RMJ-PO-Topk, and then compare their update mechanisms.

F.2.1 MALLOWS-RMJ-PO-PAIRWISE

The Mallows-RMJ-PO-Pairwise loss takes the following form:

$$\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Pairwise}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\phi(x)\cdot\sigma(f_\theta(x,y_l,y_w))\right].$$

The gradient of $\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Pairwise}}$ with respect to parameters θ takes the following formulation:

$$\begin{aligned}
&\nabla_\theta\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Pairwise}}(\pi_\theta)\\
&= -\mathbb{E}\left[\log\phi(x)\;\sigma'\big(f_\theta(x,y_l,y_w)\big)\;\nabla_\theta f_\theta(x,y_l,y_w)\right]\\
&= \beta\mathbb{E}\Bigg[\underbrace{-\log\phi(x)}_{\text{larger weight for less dispersed}}\;\underbrace{\sigma\big(f_\theta(x,y_l,y_w)\big)\big(1-\sigma\big(f_\theta(x,y_l,y_w)\big)\big)}_{\text{larger weight when reward estimates are closer}}\cdot\Big[\underbrace{\nabla_\theta\log\pi_\theta(y_l|x)}_{\text{decrease likelihood of }y_l}-\underbrace{\nabla_\theta\log\pi_\theta(y_w|x)}_{\text{increase likelihood of }y_w}\Big]\Bigg],
\end{aligned}$$

where the expectation is with respect to $(x, y_w, y_l) \sim \mathcal{D}$. See Figure 2b for an illustration of $\sigma'(\cdot)$.

F.2.2 MALLOWS-RMJ-PO-DISCRETE

The Mallows-RMJ-PO-Discrete loss takes the following form:

$$\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Discrete}}(\pi_\theta) = -\mathbb{E}_{(x,S,y_w)\sim\mathcal{D}}\log\phi(x)\cdot\sum\nolimits_{y_i\in S\setminus\{y_w\}}\sigma(f_\theta(x,y_i,y_w))\,.$$

The gradient of $\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Discrete}}$ with respect to parameters θ takes the following formulation:

$$\begin{aligned}
&\nabla_\theta\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Discrete}}(\pi_\theta)\\
&= -\mathbb{E}\Bigg[\log\phi(x)\sum_{y_i\in S\setminus\{y_w\}}\sigma'\big(f_\theta(x,y_i,y_w)\big)\;\nabla_\theta f_\theta(x,y_i,y_w)\Bigg]\\
&= \beta\mathbb{E}\Bigg[\underbrace{-\log\phi(x)}_{\text{larger weight for less dispersed}}\sum_{y_i\in S\setminus\{y_w\}}\underbrace{\sigma\big(f_\theta(x,y_i,y_w)\big)\big(1-\sigma\big(f_\theta(x,y_i,y_w)\big)\big)}_{\text{larger weight when reward estimates are closer}}\cdot\Big[\underbrace{\nabla_\theta\log\pi_\theta(y_i|x)}_{\text{decrease likelihood of }y_i}-\underbrace{\nabla_\theta\log\pi_\theta(y_w|x)}_{\text{increase likelihood of }y_w}\Big]\Bigg],
\end{aligned}$$

where the expectation is with respect to $(x, S, y_w) \sim \mathcal{D}$.

F.2.3 MALLOWS-RMJ-PO-TOPK

The Mallows-RMJ-PO-Topk loss takes the following form:

$$\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Topk}}(\pi_\theta) = -\mathbb{E}_{(x,S,\mu^k)\sim\mathcal{D}}\Bigg[\log\phi(x)\Big(\sum_{i=1}^{k-1}(|S|-i)\,\sigma(f_\theta(x,y_{i+1},y_i))\;+\sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\sigma(f_\theta(x,y_j,y_k))\Big)\Bigg].$$

The gradient of $\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Topk}}$ with respect to parameters θ takes the following formulation:

$$\begin{aligned}
&\nabla_\theta\mathcal{L}_{\mathrm{Mallows-RMJ-PO-Topk}}(\pi_\theta)\\
&= -\mathbb{E}\Bigg[\log\phi(x)\Big(\sum_{i=1}^{k-1}(|S|-i)\,\sigma'(f_\theta(x,y_{i+1},y_i))\;\nabla_\theta f_\theta(x,y_{i+1},y_i)\;+\sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\sigma'(f_\theta(x,y_j,y_k))\;\nabla_\theta f_\theta(x,y_j,y_k)\Big)\Bigg]\\
&= \beta\mathbb{E}\Bigg[\underbrace{-\log\phi(x)}_{\substack{\text{larger weight for}\\ \text{less dispersed}}}\Bigg(\sum_{i=1}^{k-1}\underbrace{(|S|-i)}_{\substack{\text{larger weight for}\\ \text{top position}}}\underbrace{\sigma\big(f_\theta(x,y_{i+1},y_i)\big)\big(1-\sigma\big(f_\theta(x,y_{i+1},y_i)\big)\big)}_{\text{larger weight when reward estimates are closer}}\Big(\underbrace{\nabla_\theta\log\pi_\theta(y_{i+1}\,|\,x)}_{\text{decrease likelihood of }y_{i+1}}-\underbrace{\nabla_\theta\log\pi_\theta(y_i\,|\,x)}_{\text{increase likelihood of }y_i}\Big)\\
&\qquad+\sum_{y_j\in S\setminus\{y_1,\ldots,y_k\}}\underbrace{\sigma\big(f_\theta(x,y_j,y_k)\big)\big(1-\sigma\big(f_\theta(x,y_j,y_k)\big)\big)}_{\text{larger weight when reward estimates are closer}}\Big(\underbrace{\nabla_\theta\log\pi_\theta(y_j\,|\,x)}_{\text{decrease likelihood of }y_j}-\underbrace{\nabla_\theta\log\pi_\theta(y_k\,|\,x)}_{\text{increase likelihood of }y_k}\Big)\Bigg)\Bigg],
\end{aligned}$$

where the expectation is with respect to $(x, S, \mu^k) \sim \mathcal{D}$.

F.2.4 Comparison of Gradient Updates

Mallows-RMJ-PO-Pairwise focuses on pairwise comparisons between a preferred response y_w and a dispreferred response y_l. The gradient increases the likelihood of y_w while decreasing that of y_l, with stronger updates when preferences are less dispersed and when the two responses have similar rewards. This concentrates learning on the harder, more informative comparisons.

Mallows-RMJ-PO-Discrete generalizes this idea to a set of responses. The update pushes up the likelihood of the preferred response against all alternatives in $S \setminus \{y_w\}$. Again, the updates are stronger when preferences are concentrated and when the competing responses are close in reward, ensuring sharper separation between the winner and its competitors.

Mallows-RMJ-PO-Topk extends to ranked choice feedback. Here the gradient simultaneously enforces the full ranking: higher-ranked responses are promoted while lower-ranked ones are suppressed. The update is amplified for top positions, for less dispersed preferences, and for cases where neighboring rewards are close. This design emphasizes both the most critical ranking positions and the harder comparisons, yielding sharper alignment to the preference order.

G Experimental Details

Our method is implemented by modifying the `DPO Trainer` and `DPO Config` in the `TRL` library. We include the modified codes and training scripts in the supplementary materials.

Computation Environment All experiments are implemented in Python 3.10.16 with PyTorch 2.6.0 on a server with 7 NVIDIA H100 GPUs each with 80 GB memory, equipped with Ubuntu 22.04.2 LTS.

Dataset We provide descriptions of the training dataset and evaluation benchmarks in Tables 5 and 6, respectively. We construct a ranking-based preference set for ranked choice training in the following way. We generate five distinct responses for each prompt in the UltraFeedback dataset (Cui et al., 2023) using a sampling temperature of 0.8. We then score and rank these five responses with the Skywork-Reward-V2-Llama-3.1-8B reward model. In the pairwise setup, we adopt the literature's tradition and use the top- and bottom- ranked responses as the assortment. For ranked choice training, we truncate each full ranking to generate the required data.

Open Source Models The Huggingface IDs of the base models and reward model used in our experiments are listed in Table 4.

Table 4: Base Models and Reward Models Used in Experiments.

Model	Huggingface ID
Llama-3-8B-Instruct	meta-llama/Meta-Llama-3-8B-Instruct
Gemma-2-9B-it	google/gemma-2-9b-it
Skywork-Reward-V2-Llama-3.1-8B	Skywork/Skywork-Reward-V2-Llama-3.1-8B

G.1 Training hyper-parameter tuning

By default of `TRL`, we use `adamw_torch` optimizer.

Llama-3-8B-Instruct We adopt a global batch size of 112, a maximum sequence length of 4096, and a cosine learning rate schedule for one epoch across all training settings. We retrieve the latest models of the baseline methods; their Huggingface IDs are listed in Table 7. The hyperparameters we used for training are in Table 8.

Table 5: Overview of the UltraFeedback Dataset (Cui et al., 2023).

Item	Description
Data Scale	∼64K prompts
Prompt Sources	UltraChat, ShareGPT, Evol-Instruct, TruthfulQA, FalseQA, FLAN
Prompt Types	Open-ended dialogue Instruction-following (writing, coding, summarization, translation) Factual QA, adversarial/false QA Standard NLP tasks (classification, reasoning, reading comprehension)
Response Source	Generated by base model
Ranking Method	Scored by Skywork-V2 (`Skywork-Reward-V2-Llama-3.1-8B`)

Table 6: Evaluation details for AlpacaEval 2 (Dubois et al., 2024) and Arena-Hard (Li et al., 2024).

	# Exs.	Baseline Model	Judge Model	Scoring Type	Metric
AlpacaEval 2	805	`GPT-4-Turbo`	`GPT-4.1-mini`	Pairwise comparison	LC & raw win rate
Arena-Hard	500	`GPT-4-0314`	`GPT-4.1-mini`	Pairwise comparison	Win rate

Notes: The baseline model refers to the model compared against. `GPT-4 Turbo` corresponds to `GPT-4-Preview-1106`. `GPT-4.1-mini` corresponds to `GPT-4.1-mini-2025-04-14`.

Table 7: List of Baseline Models Used in Experiments on Llama-3-8B-instruct.

Baseline Method	Huggingface ID
SimPO (Meng et al., 2024)	princeton-nlp/Llama-3-Instruct-8B-SimPO-v0.2
DPO (Rafailov et al., 2023)	princeton-nlp/Llama-3-Instruct-8B-DPO-v0.2
RDPO (Park et al., 2024)	princeton-nlp/Llama-3-Instruct-8B-RDPO-v0.2
CPO (Xu et al., 2024)	princeton-nlp/Llama-3-Instruct-8B-CPO-v0.2
IPO Azar et al. (2024)	princeton-nlp/Llama-3-Instruct-8B-IPO-v0.2
ORPO (Hong et al., 2024)	princeton-nlp/Llama-3-Instruct-8B-ORPO-v0.2
RRHF (Yuan et al., 2023)	princeton-nlp/Llama-3-Instruct-8B-RRHF-v0.2
SLiC-HF (Zhao et al., 2023)	princeton-nlp/Llama-3-Instruct-8B-SLiC-HF-v0.2
KTO (Ethayarajh et al., 2024)	princeton-nlp/Llama-3-Instruct-8B-KTO-v0.2

Gemma-2-9B-it We adopt a global batch size of 112, a maximum sequence length of 2048, and a cosine learning rate schedule for one epoch. We retrieve the latest models of the baseline methods; their Huggingface IDs are listed in Table 9. The hyperparameters we used for training are in Table 10.

G.2 Decoding Hyperparameters

For AlpacaEval 2.0, we adopt the default template for evaluators provided by AlpacaEval. For Llama-3-8B-Instruct settings, we adopt the following fixed generation config : max new tokens 4096, temperature 0.7 and top-p 0.1. For Gemma-2-9B-it settings, we adopt the following fixed generation config: max new tokens 4096, temperature 0.5 and top-p 1.0. For Arena-Hard-v0.1, we use the default greedy decoding for all settings and methods.

G.3 Robustness Check

To enhance cross-judge robustness, we additionally employ GPT-5-mini-2025-08-07 as the judge on Arena-Hard. We report the win rate and 95% confidence interval. The results are shown in Table 11.

H Qualitative Examples

In this section, we present a series of examples for direct comparisons between our RCPO and benchmarks, as shown in Table 12–15. These tables showcase the qualitative examples of model responses.

Table 8: Hyperparameters for training on Llama-3-8B-Instruct.

Method	β	Learning Rate	Warmup
Mallows-RMJ-PO-Pairwise	0.03	3×10^{-7}	0.2
MNL-PO-Discrete	0.01	5×10^{-7}	0.1
Mallows-RMJ-PO-Discrete	0.03	3×10^{-7}	0.2
MNL-PO-Top2	0.01	5×10^{-7}	0.1
Mallows-RMJ-PO-Top2	0.01	3×10^{-7}	0.1

Table 9: List of baseline models used in experiments on Gemma-2-9B-it.

Baseline Method	Huggingface ID
SimPO (Meng et al., 2024)	princeton-nlp/gemma-2-9b-it-SimPO
DPO (Rafailov et al., 2023)	princeton-nlp/gemma-2-9b-it-DPO

Table 10: Hyperparameters for training on Gemma-2-9B-it.

Method	β	Learning Rate	Warmup
Mallows-RMJ-PO-Top2	0.01	5×10^{-7}	0.2

Table 11: Evaluation Results for Llama-3-8B-Instruct and Gemma-2-9B-it (Judged by GPT-5-mini).

Base	**Method**	**WR (%)**	**95% CI**
Llama-3-8B-Instruct	Base Model	20.9	[19.3, 22.7]
	CPO (Xu et al., 2024)	23.7	[22.1, 25.4]
	IPO (Azar et al., 2024)	23.2	[21.6, 24.7]
	ORPO (Hong et al., 2024)	22.1	[20.5, 23.8]
	RRHF (Yuan et al., 2023)	22.3	[20.7, 23.8]
	SLiC-HF (Zhao et al., 2023)	23.1	[21.5, 24.7]
	KTO (Ethayarajh et al., 2024)	21.1	[19.4, 22.5]
	DPO (Rafailov et al., 2023)	25.9	[24.7, 27.3]
	R-DPO (Park et al., 2024)	24.9	[22.8, 26.4]
	SimPO (Meng et al., 2024)	27.2	[25.4, 28.7]
	Mallows-RMJ-PO-Pairwise	**27.4**	[25.8, 28.9]
	MNL-PO-Discrete	25.1	[23.1, 27.0]
	Mallows-RMJ-PO-Discrete	25.4	[23.8, 27.3]
	MNL-PO-Top2	24.9	[23.6, 26.5]
	Mallows-RMJ-PO-Top2	25.8	[24.1, 27.5]
Gemma-2-9B-it	Base Model	38.2	[35.9, 40.3]
	SimPO (Meng et al., 2024)	47.0	[45.3, 49.0]
	DPO (Rafailov et al., 2023)	49.4	[47.3, 51.6]
	Mallows-RMJ-PO-Top2	**51.0**	[48.7, 52.8]

Table 12: Qualitative Example (SAT Word)

Question: *Find the answer that best describes the underlined SAT word. Select the correct option and explain the meaning of the underlined word.*
Despite the cacophony, the student tried to study.
A. Loud sounds B. Difficult subject C. Late hour D. Low lighting

Mallows-RMJ-PO-Top-2	**DPO**
The correct answer is A. Loud sounds. The word "*cacophony*" refers to a harsh, discordant sound or mixture of sounds, often unpleasant to the ear. In this context, "despite the cacophony" means that despite the loud and unpleasant noise, the student tried to study. **Example:** The construction site outside my window created a cacophony of hammering, drilling, and shouting, but I managed to focus on my work. The other options don't fit the context: **B. Difficult subject:** The word "cacophony" doesn't relate to a challenging academic subject. **C. Late hour:** While it's possible to study at a late hour, the word "cacophony" doesn't describe the time of day. **D. Low lighting:** This option is unrelated to the concept of loud, unpleasant sounds.	**The correct answer is A. Loud sounds.** The word "*cacophony*" refers to a harsh, discordant sound or a mixture of loud, unpleasant noises. In this context, "despite the cacophony" means that despite the loud and unpleasant sounds (e.g., noise from construction, traffic, etc.) surrounding the student, they still tried to study. **Example:** If a student is trying to study in a busy city street or a noisy café, the cacophony of sounds from cars honking, people talking, and music playing might be distracting, but the student is trying to focus despite these unpleasant sounds.

Notes: Compared with DPO, the response generated by Mallows-RMJ-PO-Top-2 also explains why the other options are incorrect.

Table 13: Qualitative Example (Excel Test)

Question: *Please write the Excel function name associated with each description.*
- *Returns the number of days between two dates*
- *Returns the starting position of a text string within another text string*
- *Returns the number in the middle of the set of given numbers*

Mallows-RMJ-PO-Top-2	**DPO**

Here are the Excel function names associated with each description:
 	
1. **Returns the number of days between two dates:** `DATEDIF` or `DAYS`
   - `DATEDIF` is a more flexible function that can calculate the difference between two dates in various units (days, months, years, etc.), while `DAYS` is a simpler function that returns the exact number of days between two dates.
2. **Returns the starting position of a text string within another text string:** `FIND` or `SEARCH`
   - `FIND` is case-sensitive, while `SEARCH` is not.
3. **Returns the number in the middle of the set of given numbers:** `MEDIAN`
   - This function returns the middle value in a set of numbers, assuming the set is sorted in ascending order. If the set has an even number of values, `MEDIAN` returns the average of the two middle values.
	Here are the Excel function names associated with each description:
	
	1. **Returns the number of days between two dates:** `DATEDIF` or `DAYS`
	However, if you want a single function that returns the exact number of days between two dates, you can use a custom formula: `=A2-A1+1` (assuming A1 and A2 are the start and end dates).
	2. **Returns the starting position of a text string within another text string:** `FIND` or `SEARCH`
	   - `FIND` is case-sensitive, while `SEARCH` is not.
	3. **Returns the number in the middle of the set of given numbers:** `AVERAGE` or `MEDIAN`
	   - If you want to return the exact middle value (in case of an even number of values), you can use `=AVERAGE(MIDIF(A1:A10,1, COUNT(A1:A10)))` (assuming A1:A10 is the range of numbers), where `MIDIF` is a custom array formula.
	   - Alternatively, you can use `=INDEX(A1:A10,MATCH(2,IF(A1:A10 <MIN(A1:A10)+MAX(A1:A10), ROW(A1:A10),""),0))` (assuming A1:A10 is the range of numbers).
	   - For a simpler approach, you can use `=AVERAGE(A1:A10)` and then round to the nearest integer if needed.
	   - For a more robust approach, use `=MEDIAN(A1:A10)`.

Notes: In the Excel test, Mallows-RMJ-PO-Top-2 answered all three questions correctly, whereas DPO incorrectly calculated the mean in the last question using the average function.

Table 14: Qualitative Example (Open Question)

Question: *Which libraries are the best for developing deep learning scripts in Python?*

Mallows-RMJ-PO-Top-2	**DPO**
For developing deep learning scripts in Python, you'll want to focus on libraries that provide efficient implementations of neural networks, automatic differentiation, and optimization algorithms. Here are some of the most popular and widely-used libraries: 1. **TensorFlow (TF)**: An open-source library developed by Google. Provides a Python API, supports both CPU and GPU acceleration, and has a large community of developers. **Pros:** Wide range of pre-built estimators, flexible architecture, good documentation. **Cons:** Steeper learning curve, can be slow for small models. 2. **PyTorch**: Known for its dynamic computation graph and automatic differentiation. Popular among researchers who prefer a more Pythonic API. **Pros:** Easier to learn, dynamic computation graph, fast prototyping. **Cons:** Less mature than TensorFlow, fewer pre-built estimators. 3. **Keras**: High-level neural networks API written in Python. Can run on top of TensorFlow, PyTorch, or Theano. **Pros:** Easy to use, rapid prototyping, supports multiple backends. **Cons:** Limited control over low-level details, not suitable for complex models. 4. **Theano**: Library from the University of Montreal, focused on automatic differentiation and symbolic math. **Pros:** Strong support for symbolic math, research-friendly, high-performance. **Cons:** Steep learning curve, less user-friendly. 5. **Microsoft Cognitive Toolkit (CNTK)**: A commercial-grade DL library from Microsoft, known for high-performance computing and large-scale support. **Pros:** High-performance, supports large-scale models, good documentation. **Cons:** Commercial license required for some features, less widely used. 6. **Hugging Face Transformers**: Focused on NLP and transformer-based models, provides pre-trained models and simple interfaces. **Pros:** Easy to use, large community, pre-trained models. **Cons:** Limited to NLP and transformer-based models. **When choosing a library, consider:** * Your project's specific requirements (large datasets/complex models: TensorFlow, PyTorch; simple/rapid prototyping: Keras, Hugging Face). * Your team's expertise (familiarity makes adoption easier). * Computational resources (if limited, Keras or Hugging Face can run on CPU/GPU). * Community support (active communities provide better docs/tutorials). Ultimately, it's a good idea to experiment with different libraries to determine which best fits your needs and preferences. You can also use libraries like **scikit-learn**, which provides a unified interface for various ML algorithms, including some DL techniques.	For developing deep learning scripts in Python, you'll want to focus on libraries that provide efficient implementations of neural networks and related algorithms. Here are some of the most popular and widely used libraries: 1. **TensorFlow (TF)**: An open-source library developed by Google. Popular for deep learning, especially for large-scale projects. Provides a wide range of tools and APIs for building and training neural networks. 2. **PyTorch**: Another open-source library, developed by Facebook's AI Research Lab (FAIR). Known for its dynamic computation graph, automatic differentiation, and ease of use. Particularly popular among researchers and developers who prefer a Pythonic API. 3. **Keras**: A high-level neural networks API written in Python, capable of running on top of TensorFlow, PyTorch, or Theano. Provides an easy-to-use interface for building and experimenting with deep learning models, suitable for beginners and experts. 4. **SciPy**: While not exclusively a deep learning library, SciPy's `scipy.optimize` and `scipy.signal` modules provide useful tools for optimization and signal processing, which are essential for many deep learning algorithms. For specific tasks, you may also want to consider the following libraries: 5. **OpenCV (Computer Vision)**: For tasks such as image and video processing, feature extraction, and object detection. 6. **NLTK (Natural Language Toolkit)** and **spaCy**: For natural language processing (NLP) tasks, such as text processing, tokenization, and language modeling. 7. **PyCaffe (optional)**: A Python wrapper for the Caffe deep learning framework, useful for deploying pre-trained models. **When choosing a library, consider the following factors:** * **Ease of use:** If you're new to deep learning, Keras or PyTorch might be a better choice due to their more Pythonic APIs. * **Performance:** TensorFlow and PyTorch are generally more performant for large-scale deep learning tasks. * **Specific requirements:** If you need to work with computer vision or NLP, OpenCV and NLTK/spaCy might be more suitable. * **Community support:** All of these libraries have large communities and extensive documentation, but TensorFlow and PyTorch have a more extensive ecosystem. Here's a rough guideline: • For beginners: Keras or PyTorch. • For large-scale deep learning: TensorFlow or PyTorch. • For specific tasks (computer vision, NLP): OpenCV, NLTK/spaCy, or library-specific options (e.g., OpenCV for vision). Remember, you can always use multiple libraries together to achieve your goals. Happy deep learning!

Notes: The response generated by Mallows-RMJ-PO-Top-2 gives more details of pros and cons when listing the options compared to that generated by DPO.

Table 15: Qualitative Example (Equation Solving)

Question: *Solve this equation.* $x^3 - 4x^2 + 6x - 24 = 0$

Mallows-RMJ-PO-Pairwise

A nice cubic equation! Let's try to factor it:

1. Start with the given equation:

$$x^3 - 4x^2 + 6x - 24 = (x - 3)(x^2 - x - 8) = 0$$

2. Factor the quadratic inside:

$$x^2 - x - 8 = (x - 2)(x + 4) = 0$$

3. So we have:

$$(x - 3)(x - 2)(x + 4) = 0$$

4. This tells us that either:
   - $x - 3 = 0 \Rightarrow x = 3$
   - $x - 2 = 0 \Rightarrow x = 2$
   - $x + 4 = 0 \Rightarrow x = -4$

Therefore, the solutions to the equation are

$$x = 3, \quad x = 2, \quad x = -4.$$

DPO

A nice cubic equation! To solve this equation, we can try various methods. Here are a few approaches:

1. **Method 1: Factoring**
   Let's see if we can factor the left-hand side of the equation:

   $$x^3 - 4x^2 + 6x - 24 = (x - 3)(x^2 - x - 8) = 0$$

   Now, we need to factor the quadratic expression inside:

   $$x^2 - x - 8 = (x - 2)(x + 4) = 0$$

   So, we have:

   $$(x - 3)(x - 2)(x + 4) = 0$$

   This tells us that:

   $$x = 3, \quad x = 2, \quad x = -4$$

2. **Method 2: Numerical Methods or Formula**
   If factoring doesn't work, we can use numerical methods or the cubic formula. For example:

   $$x = \frac{-b \pm \sqrt{b^2 - 3ac}}{3a} + \frac{(2c \pm \sqrt{(b^2 - 3ac)(3a - 2b) + c^3})}{3a^2}$$

   In this case, $a = 1$, $b = -4$, $c = 6$. Plugging in the values gives messy expressions, and after simplification, we don't get a neat closed form. The cubic formula often leads to complicated or approximate solutions.

Conclusion:
The most straightforward approach is factoring, which yields the solutions:

$$x = 3, \quad x = 2, \quad x = -4$$

These are the exact solutions to the equation:

$$x^3 - 4x^2 + 6x - 24 = 0$$

Notes: For the equation-solving task, Mallows-RMJ-PO-Pairwise provided the correct solution, whereas DPO gave an incorrect answer (method 2).