VPO: Leveraging the Number of Votes in Preference Optimization

Jae Hyeon Cho, Minkyung Park, and Byung-Jun Lee*

Korea University
bonin147@korea.ac.kr, swwwjkl538@korea.ac.kr, byungjunlee@korea.ac.kr

Direct Preference Optimization (DPO) trains a language model using human preference data, bypassing the explicit reward modeling phase of Reinforcement Learning from Human Feedback (RLHF). By iterating over sentence pairs in a preference dataset, DPO enhances generation quality by increasing the likelihood of producing preferred sentences over less favored ones. Preference datasets, typically labeled with votes or scores, provide valuable insights into whether a sentence pair exhibits a clear preference or remains controversial. However, existing methods do not fully utilize this information. In this article, we propose a technique that leverages user voting data to better align language models with diverse subjective preferences. We use the Bayesian Minimum Mean Square Error (Bayesian MMSE) estimator to model the probability that one generation is preferred over another. Using this estimated probability as a target, we introduce the Vote-based Preference Optimization (VPO) framework, which incorporates the number of votes on both sides to distinguish between controversial and clearly preferred generation pairs. Furthermore, we demonstrate that previous algorithms, such as DPO and Identity Preference Optimization (IPO), can be extended using the proposed framework, termed VDPO and VIPO. Our experiments demonstrate that these proposed algorithms outperform various existing methods, including their base algorithms. Additionally, our framework can be applied to reward modeling, demonstrating that our approach is compatible with the broader RLHF pipeline.

1. Introduction

In general-domain applications of language models (LMs), the model should be aligned with human values, such as helpfulness, honesty, and harmlessness. Pretraining and supervised fine-tuning (SFT) enable the development of models with notable capabilities across a wide range of natural language processing (NLP) tasks (Wei et al. 2021; Wang et al. 2023). However, additional training using pairwise preference data is often utilized to further align the model with human values.

Preference alignment methods, such as reinforcement learning from human feedback (RLHF; Stiennon et al. 2020; Ouyang et al. 2022) and direct preference optimization

* Corresponding author.

Action Editor: Xuanjing Huang. Submission received: 15 April 2025; revised version received: 23 September 2025; accepted for publication: 19 October 2025.

https://doi.org/10.1162/COLI.a.579

(DPO; Rafailov et al. 2023), have shown significant success in enhancing the human usability of language models. Consequently, these preference optimization processes are now considered essential in the development of state-of-the-art large LMs (Achiam et al. 2023; Gemini Team et al. 2023).

Given pairwise preference data with labels indicating which response is preferred, RLHF trains a reward model to align with these preferences, enabling the evaluation of a language model's outputs. Subsequently, the language model is trained using a reinforcement learning algorithm to maximize the expected reward of its generated responses. In contrast, DPO provides an alternative approach by directly adjusting the generation probabilities of the language model based on preference labels. This method eliminates the need for a separate reward modeling phase, thereby reducing computational costs.

However, we note that the current labels in pairwise preference datasets may provide limited information in these processes. In many cases, given a prompt with two responses, the preference for one over the other may be clear, while in others it may be more controversial. We posit that side information associated with each response—such as votes or scores—offers valuable cues about the degree of certainty or disagreement in human judgments. However, this additional information has been largely overlooked in prior work on preference alignment.

In this paper, we introduce a simple yet effective method (Figure 1) to better utilize the rich side information inherent in human preference datasets. Our approach models the underlying target preference probability using the Bayesian Minimum Mean Square

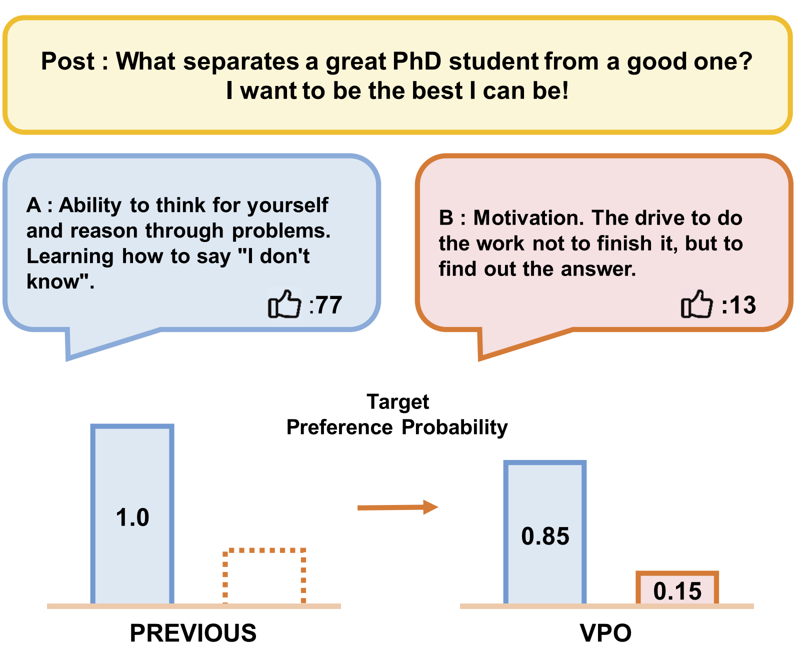

Figure 1
While previous methods trained models to generate responses based on majority preference (e.g., A), human preferences are subjective, making responses like B also desirable. Our proposed framework, VPO, utilizes additional information to capture a more nuanced understanding of these preferences.

Error (MMSE) estimator, enabling the model to distinguish between learning from clear-cut examples (those with a high vote/score gap) and contentious cases (those with a low vote/score gap). We term this framework as Vote-based Preference Optimization (VPO), and extend established algorithms such as DPO and Identity Preference Optimization (IPO; Azar et al. 2024) into VDPO and VIPO, respectively, demonstrating the broad applicability of our approach. Furthermore, we demonstrate that our method can be effectively applied to reward modeling in RLHF.

In the experiments, we empirically demonstrate the following:

- **Performance Improvement**: VDPO and VIPO outperform existing algorithms, achieving improved generation quality and training stability.
- **Broad Adaptability**: Our framework is compatible with both vote-based and score-based human preference datasets. This flexibility allows effective utilization of AI-generated feedback in settings where human voting data is scarce or unavailable, expanding its applicability across a wide range of preference alignment scenarios.

Our main contributions are as follows:

1. We introduce the **Vote-based Preference Optimization (VPO)** framework, which leverages vote counts to estimate soft preference probabilities via Bayesian MMSE and extends existing methods such as DPO and IPO.
2. We generalize our approach from vote-based signal to score-based signal, enabling application to a broader set of datasets that contain scalar preference feedback.
3. We demonstrate that VPO is not limited to alignment without reward modeling but can also be applied effectively to reward modeling, showing compatibility with the full RLHF pipeline.

2. Preliminaries

In this section, we outline the standard procedures for training a general-purpose language model prior to aligning it with human values. The process begins with the following two steps:

Pretraining. To provide the model with general capabilities, it is trained on a large corpus using next token prediction, commonly referred to as teacher forcing.

Supervised Fine-Tuning. Following pretraining, supervised fine-tuning is performed to equip the model with specific abilities required for the target domain tasks. During this phase, the model is trained on a dataset specifically curated for the intended tasks. We refer to the model after this step as π_{ref} henceforth.

2.1 RLHF: Alignment via Reward Modeling

The standard RLHF process consists of two steps.

Reward Model Training. The reward model is trained using human preference data to align its judgments with human values. The human preference dataset is composed of the triplet $\mathcal{D} = \{x, y_1, y_2\}$, where x is the context, and y_1 and y_2 are response pairs given the context. We define the events:

$$Y_1 = y_1 \text{ is favored over } y_2, \quad Y_2 = y_2 \text{ is favored over } y_1.$$

The probability of these events is modeled using a Bradley-Terry model, which is defined as follows:

$$\hat{p}_r(Y_1|x) := \frac{\exp(r(x, y_1))}{\exp(r(x, y_1)) + \exp(r(x, y_2))}$$

The reward model is then optimized by maximizing the log-likelihood of the human preference dataset,

$$\max_r \mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}}\left[\log \hat{p}_r(Y_1|x)\right] \tag{1}$$

assuming, without loss of generality, that y_1 is the preferred response.

RL Fine-Tuning. After training a reward model, a regularized RL algorithm is used to maximize the expected reward while ensuring the model does not deviate significantly from the initial model π_{ref}:

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

This approach ensures that the updated model achieves high reward, meaning strong alignment with human preferences, while preserving the general language capabilities of the reference model.

2.2 DPO: Alignment without Reward Modeling

Direct Preference Optimization. Training an additional reward model, along with using reinforcement learning to fine-tune a model, involves numerous complex engineering challenges. DPO provides an alternative approach by directly training the language model on the preference dataset by substituting the reward model with its closed-form solution.

Assuming y_1 is preferred over y_2, the DPO objective is defined as follows:

$$\max_\theta \ \mathbb{E}_\mathcal{D}\left[\log \sigma(r(x,y_1) - r(x,y_2))\right] \tag{2}$$

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

$$\text{and } Z(x) = \sum_y \pi_{\text{ref}}(y|x) \exp\left(\frac{1}{\beta} r(x,y)\right)$$

DPO simplifies the training process by directly leveraging the dataset, thereby enhancing both stability and efficiency.

Reward Divergence. Despite its simplicity, DPO has a theoretical limitation known as reward divergence, as discussed in Azar et al. (2024). Specifically, the DPO objective can cause the difference $r(x,y_1) - r(x,y_2)$ to grow without bound for a given data point (x, y_1, y_2). This unbounded growth leads to an implicit reward function with an inflated scale, which undermines the regularization effect imposed by the reference policy π_{ref}. In practice, this behavior often causes DPO training to become unstable over multiple epochs, requiring techniques like early stopping.

To address this issue, IPO was proposed, which analytically corrects the DPO objective by enforcing a fixed margin between the log-ratio of preferred and dispreferred outputs. The IPO objective is formulated as:

$$\max_\theta \ \mathbb{E}_\mathcal{D}\left[\log \frac{\pi(y_1|x)}{\pi_{\text{ref}}(y_1|x)} - \log \frac{\pi(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \frac{1}{2\beta}\right]^2 \tag{3}$$

In parallel, other works have focused on the issue of label noise in preference datasets. For example, Conservative DPO (cDPO) introduces a probabilistic mixture of the original and flipped labels to account for potential inconsistencies in human annotations. The cDPO objective is given by:

$$\max_\theta \ \mathbb{E}_\mathcal{D}\big[(1-c)\log\sigma\big(r(x,y_1) - r(x,y_2)\big) + c\log\sigma\big(r(x,y_2) - r(x,y_1)\big)\big] \tag{4}$$

where $c \in (0, 0.5)$ is a hyperparameter representing the estimated probability that the label is flipped. This formulation increases robustness in noisy annotation settings.

To further address label noise from a theoretical perspective, Chowdhury, Kini, and Natarajan (2024) introduced Robust DPO (rDPO). It reweights the objective based on the estimated flip probability ϵ, and is defined as:

$$\max_\theta \ \mathbb{E}_\mathcal{D}\left[\frac{1-\epsilon}{1-2\epsilon}\log\sigma\left(r(x,y_1) - r(x,y_2)\right) - \frac{\epsilon}{1-2\epsilon}\log\sigma\left(r(x,y_2) - r(x,y_1)\right)\right], \tag{5}$$

where $\epsilon \in (0, 0.5)$ represents the estimated noise level in the labels.

Remark. While our proposed VPO framework is broadly applicable to a variety of preference optimization algorithms and reward modeling settings, we focus on DPO and its variants for two reasons. First, DPO has emerged as a widely adopted baseline due to its simplicity and strong empirical performance, making it a natural point of comparison. Second, the limitations of DPO, such as reward divergence and lack of robustness to noisy or ambiguous preferences, are directly addressed by our method through vote-informed target probabilities. Thus, while not limited to DPO, our work highlights it as a representative case where vote information can be effectively leveraged.

3. Related Work

Alignment without Reward Modeling. Since the introduction of DPO, several studies have focused on improving the efficiency of preference alignment algorithms. As briefly introduced above, Azar et al. (2024) mathematically analyzed the issue of diverging rewards in DPO and proposed IPO as a potential solution. Ethayarajh et al. (2024) introduced Kahneman-Tversky Optimization (KTO), which utilizes the Kahneman-Tversky human utility function to better align with human values. Hong, Lee, and Thorne (2024) presented the Odds Ratio Preference Optimization (ORPO), a reference model-free approach that eliminates the dependency on a baseline model, thereby simplifying the optimization process.

Although various improvements to DPO are being explored, they still share the limitation of not fully utilizing side information beyond the binary indication of more or less preferred. In this article, we propose enhancing existing algorithms by incorporating additional side data. The improvements we suggest are orthogonal to the advancements made by these existing methods and can be seamlessly integrated into all of these approaches.

Noise in Preference Labels. Several studies have examined the potential for preference labels to be noisy due to human subjectivity. While the primary objective of cDPO (Mitchell 2023) was to address the issue of reward divergence, the algorithm was formulated with the assumption that preference labels may contain noise. To further enhance the robustness of learning in noisy environments, Chowdhury, Kini, and Natarajan (2024) developed robust DPO (rDPO), which is specifically designed to minimize the impact of noise in preference labels.

As we will demonstrate, our VPO framework can also be interpreted as modeling the level of noise in preference labels using side information. In cDPO and rDPO, this noise level is assumed to be constant and is tuned as a hyperparameter. In contrast, our approach offers a straightforward and intuitive method for estimating noise levels in the preference dataset, building on a similar framework.

Side Information in Reward Modeling. In the case of InstructGPT (Ouyang et al. 2022), scores were collected as side information during data construction. However, a key limitation is that this side information was entirely disregarded during the reward modeling process, where only relative rankings were used.

Helpsteer2-pref (Wang et al. 2025) introduced the Helpsteer2-preference dataset, which includes side information. Additionally, it proposed the Scaled Bradley-Terry (SBT) loss as a reward model objective. This loss is a weighted variant of the traditional Bradley-Terry model and converges to the same optimal solution (the same applies to the Margin BT loss discussed in the article). However, when minimizing these losses, the reward margin $(r_\theta(x, y_c) - r_\theta(x, y_r))$ can diverge to infinity, leading to the problem of reward overoptimization. In contrast, our approach enables the use of side information while simultaneously addressing the issue of reward divergence.

Disagreement in Human Preferences. MaxMin-RLHF (Chakraborty et al. 2024) tackles the challenge of controversial preferences arising from factors such as race or gender by training separate reward models tailored to different demographic groups. While the recognition that some preferences may be inherently controversial aligns with our motivation, our perspective differs in that we do not focus on demographic factors. Instead, we consider that two responses may be controversial simply because their quality is comparable, regardless of the annotator's background. Moreover, training multiple reward models to capture such nuances incurs substantial computational costs, whereas our approach provides a more efficient alternative.

4. Method

For constructing human preference datasets, each generation pair can be evaluated multiple times by different evaluators to account for the variability in their judgments. Although the number of votes from these evaluators is usually recorded during dataset creation, this information has often been underutilized in previous methodologies. Below, we provide a detailed illustrative example to emphasize this point.

Illustrative Example. Table 1 presents an example of the raw data format (post and comments) from the Stanford Human Preference (SHP) dataset.

Using the conventional approach to align a language model, we lose valuable side information, leading the model to be trained to prefer y_1 over y_2, y_2 over y_3, y_3 over y_4, all with the same margins between them. However, a human evaluator would likely judge that y_1 should be preferred over the others by a significant margin, while the other three responses—y_2, y_3, and y_4—are of roughly equal quality.

To this end, we propose modeling the **target preference probability**: $p(Y_1|x, v_1, v_2)$ and $p(Y_2|x, v_1, v_2)$, where v_1 and v_2 represent the number of votes for y_1 and y_2, respectively. In previous approaches, it is typically assumed that y_1 is the preferred response, assigning $p(Y_1|x) = 1$ and $p(Y_2|x) = 0$. Instead, we employ a Bayesian approach to model the target preference probability, taking into account the number of votes v_1, v_2 collected during dataset construction. This approach allows for a nuanced interpretation of vote counts, enabling the distinction between different vote distributions, ranging from controversial response pairs to more obvious ones.

4.1 Modeling Targets with Bayesian MMSE

To better align with the human preference through the finite number of assessments in the preference dataset, we adopt a Bayesian approach. We begin by defining the prior distribution of the target preference probability when no voting information is observed:

$$p(Y_1|x) \sim \text{Beta}(c, c) \quad \text{where } c > 0 \text{ is a hyperparameter}$$

Since the response pair can appear in any order, the distribution should be order-invariant. To ensure this, the parameters of the Beta distribution are set equal, both taking the value c. The hyperparameter c controls the influence of the number of votes on the posterior distribution: larger values of c reduce the impact of voting, while smaller values amplify it.

Table 1
Example from the SHP dataset illustrating a post and four comments with different vote counts. Conventional approaches consider the relationship between (y_2, y_3) the same as between (y_1, y_4), which is undesirable. This example demonstrates the limitations of reducing vote information to a simple binary indication of more or less preferred.

Post	Anybody else almost always reduce the sugar in recipes? Hi guys, This post was prompted by making my first baked cheesecake. I followed this King Arthur Baking recipe which calls 347g of sugar. Thought that was a little crazy, so reduced it to 190g. So the cheesecake is done and it's DELICIOUS but *very* rich, to the point where I can't imagine what it would've been like if I used the full amount of sugar. I do this a lot with cakes, tarts and muffins (what I usually make) and have never had any problems, so I do wonder why recipes contain such a high amount of sugar. I guess a follow up question would be are there any particular bakes where you absolutely need the amount of sugar specified?

Vote	Comment
101	y_1: **It's about balance.** Your cheesecake has a lot of rich ingredients, like 8 ounces of cream cheese, 1/2 cup sour cream, 5 eggs, and a ton of butter in the crust. The sugar balances the richness from these ingredients, so reducing the sugar will just ruin that balance and make it very rich. "A lot" of sugar does not always mean it's going to be overly sweet. Too much sugar means it will be overly sweet, and what someone considers "too much" is always personal preference. Also, these amounts are deceiving because you are talking about a full 10" cheesecake here, not a single serving. A single serving is going to be about 1/8th of that.
15	y_2: I always cut the sugar in half. I want to taste everything in my dessert, not just sugar.
14	y_3: I almost always cut it by 1/4 to a 1/2. I like to taste flavors not just sugar and my teeth don't feel as gritty either.
9	y_4: I live in Brazil and the overall palate and traditional recipes here are always too sweet for me. I tend to dial down everything I make for myself. But, Im a pastry chef, and things I do for my job tend to be a little sweeter than I care for, but still a little less sweet than the common Brazilian dessert.

Let $p(Y_1|x)$ be denoted as θ. The number of votes for each option is represented by the non-negative integers v_1 and v_2. We model the likelihood function for human preferences using a binomial distribution, which simplifies the computation of the posterior due to its conjugate properties:

$$p(v_1, v_2|\theta) \propto \theta^{v_1}(1-\theta)^{v_2}$$
$$p(\theta|v_1, v_2) = \text{Beta}(v_1 + c, v_2 + c)$$

A straightforward approach to utilizing the posterior distribution $p(\theta \mid v_1, v_2)$ would be to sample θ each time the language model is updated based on a response pair. However, to better stabilize the training of a large model, we use the Bayesian MMSE estimator, which involves simply taking the mean:

$$\hat{\theta}_{\text{MMSE}}(v_1, v_2) = \mathbb{E}[\theta|v_1, v_2] = \frac{v_1 + c}{v_1 + v_2 + 2c} \tag{6}$$

where the name derives from its property.

Theorem 1
(Pishro-Nik 2014) The Bayesian MMSE estimator is the solution to the following:

$$\hat{\theta}_{\text{MMSE}}(v_1, v_2) = \arg\min_{\hat{\theta}} \int (\hat{\theta} - \theta)^2 p(\theta|v_1, v_2) d\theta$$

Using the Bayesian MMSE estimator allows us to convey the implication of various numbers of votes of response pairs without introducing additional stochasticity to the training.

Modeling each annotator's response as an independent label corresponds to assuming a uniform prior distribution over the preference probability, which is equivalent to setting $c = 0$ in our Bayesian MMSE estimator. While this approach is included as a special case in our framework, it fails to distinguish between situations with different levels of annotator certainty. For example, vote counts of 1 vs. 2 and 10 vs. 20 both yield the same ratio, despite the latter providing significantly stronger evidence due to the larger number of annotators. This limitation can lead to overconfident estimates based on sparse data. Empirically, we also observed that setting $c = 0$ resulted in the lowest performance among all tested values, as shown in Section 6.4.

Illustrative Example. Table 2 provides examples of the Bayesian MMSE estimator with the hyperparameter $c = 1$.

- For clear-cut response pairs such as 101 : 9, the estimator indicates a strong preference of 0.91.
- For controversial pairs like 15 : 14, the estimator shows a much weaker preference at 0.52.
- In the case of the pair 14 : 9, although the ratio suggest a significant preference for the first response, the estimator provides a moderate preference of 0.6, acknowledging that the vote count does not provide enough evidence.

These examples demonstrate how the Bayesian MMSE estimator enables the language model to learn differently from various response pairs, taking into account the number of votes to better align with subjective human preferences.

Table 2
The estimated probability based on Bayesian MMSE estimator for different vote count with $c = 1$, compared to the target preference probability we had with DPO.

votes	$\hat{\theta}_{\text{MMSE}}(v_1, v_2)$	$p(Y_1\|x)$
101:9	0.91	1
15:14	0.516	1
14:9	0.6	1

4.2 Vote-based Preference Optimization (VPO)

Adopting the Bayesian MMSE estimator as the target preference probability, $p(Y_1|x, v_1, v_2) = \hat{\theta}_{\text{MMSE}}(v_1, v_2)$, creates a versatile framework that can be generalized to extend various preference optimization algorithms. We refer to this collection of extended algorithms as the **Vote-based Preference Optimization (VPO)** framework, which enables a more nuanced understanding of subjective human preferences.

Cross Entropy with Generalized Targets. In previous approaches, including RLHF and DPO, the (implicit) reward function is trained by maximizing the log-likelihood, as shown in Equation (1). This can be interpreted as assuming $p(Y_1|x) = 1$ as the target and using a cross entropy objective. By adopting the generalized target probability $p(Y_1|x, v_1, v_2)$ from VPO, we now obtain a generalized reward loss function:

$$\max_r \mathbb{E}_{\mathcal{D}} \left[\sum_{i=1}^{2} p(Y_i|x, v_1, v_2) \log \hat{p}_r(Y_i|x) \right] \tag{7}$$

This objective functions as an adaptive label smoothing mechanism, ensuring that the reward function learns to have a large reward margin for substantial vote gaps and a smaller reward margin for narrower vote gaps.

Vote-based Direct Preference Optimization (VDPO). To implement our approach within DPO, we generalize Equation (2) using the target preference probability from VPO:

$$\begin{aligned} \max_\theta \; \mathbb{E}_{\mathcal{D}} \big[& p(Y_1|x, v_1, v_2) \log \sigma\big(r(x, y_1) - r(x, y_2)\big) \\ & + p(Y_2|x, v_1, v_2) \log \sigma\big(r(x, y_2) - r(x, y_1)\big) \big] \end{aligned} \tag{8}$$

where $r(\cdot)$ is defined as in Equation (2). In addition to differentiating response pairs with varying vote ratios, as discussed in Mitchell (2023), including both the more preferred and the less preferred responses contributes to more stable training by addressing the issue of reward divergence.

Vote-based Identity Preference Optimization (VIPO). While IPO (Azar et al. 2024) was introduced to address reward divergence, it can still benefit from distinguishing pairs with varying vote ratios by incorporating VPO. Its objective is:

$$\min_\theta \mathbb{E}_{\mathcal{D}} \left[\left(r(x, y_1) - r(x, y_2) - \frac{1}{2\beta} \right)^2 \right] \tag{9}$$

which tries to fix the reward margin to be $\frac{1}{2\beta}$. This objective is derived from:

$$\begin{aligned} \min_\theta \mathbb{E}_{\mathcal{D}} \Big[& \big(r(x, y_1) - r(x, y_2) - \beta^{-1} p(Y_1|x)\big)^2 \\ & + \big(r(x, y_2) - r(x, y_1) - \beta^{-1} p(Y_2|x)\big)^2 \Big] \end{aligned}$$

with $p(Y_1|x) = 1$ and $p(Y_2|x) = 0$. Adopting VPO by substituting $p(Y_1|x, v_1, v_2)$, up to a constant we get:

$$\min_\theta \mathbb{E}_\mathcal{D}\left[\left(r(x, y_1) - r(x, y_2) - \frac{2p(Y_1|x, v_1, v_2) - 1}{2\beta}\right)^2\right]$$

By leveraging vote-based information, VIPO adjusts the reward margin to be proportional to the strength of human preference, up to a maximum of $\frac{1}{2\beta}$.

Remark. Similar to our approach, both rDPO and cDPO aim to address the noise inherent in preference labels. However, a fundamental difference lies in how they model this noise: Both methods assume a fixed level of label noise, controlled by a specific hyperparameter (e.g., ϵ), and derive corresponding loss functions accordingly. In contrast, our method introduces a framework that models the degree of label uncertainty based on side information. By applying this framework to cDPO, we derive VDPO, and similarly, applying it to IPO yields VIPO. While our framework could in principle be extended to rDPO, we do not explore this direction in the present work, as rDPO has received limited attention in practice due to its relatively weak empirical performance despite its theoretical motivations. We leave this extension as future work.

5. Experiments

In this section, we outline the experimental settings used to evaluate the performance of the proposed VPO framework.

5.1 Training Details

Data. Our experiments utilize two widely recognized binary human preference datasets: the Stanford Human Preferences dataset (SHP; Ethayarajh, Choi, and Swayamdipta 2022) and the UltraFeedback Binarized dataset (UFB; Cui et al. 2023).

- The SHP dataset consists of Reddit data, where the voting score is calculated by subtracting the number of negative votes from the number of positive votes and then adding one to the result. We use the voting scores directly as v_1 and v_2 for computing the target preference probability $p(Y_i|x, v_1, v_2)$.
- The UFB dataset is a synthetic human preference dataset built for alignment tasks, with a focus on instruction-following, truthfulness, honesty, and helpfulness. It aggregates prompts from a diverse set of existing sources, including TruthfulQA (Lin, Hilton, and Evans 2022), FalseQA (Hu et al. 2023), Evol-Instruct (Xu et al. 2023), UltraChat (Ding et al. 2023), ShareGPT (Chiang et al. 2023), and Flan (Longpre et al. 2023). Responses were generated by various language models and ranked using GPT-4, producing preference scores ranging from 1 to 10. To account for the distinct scaling of these scores compared with human-annotated vote counts, we exponentiate the scores prior to computing the target preference probability. This approach is further discussed in Section 6.3.

- Bartolomé Del Canto et al. (2024) propose a framework for generating synthetic preference data using AI feedback, similar in spirit to UFB. To assess the generality of our approach in domains beyond instruction-following, we adopt the Distilabel-Math-Preference-DPO (DMP) dataset, which focuses on mathematical reasoning tasks. This dataset consists of 2.42k high-quality examples, each annotated with scalar preference scores derived from LLM-based evaluation. As with UFB, these scores are used as side information. Following our findings in Section 6.3, where exponentiating bounded scores led to improved modeling of preference uncertainty, we apply the same exponential transformation to this dataset.

We follow the convention of limiting the preference dataset to a maximum of five pairwise comparisons per post to effectively manage the large number of comments associated with certain posts in the SHP dataset.

Model. In our study, we use three pretrained models: the Pythia 2.8B model (Biderman et al. 2023), the LLaMA 7B model (Touvron et al. 2023), and the QWEN 0.5B model (Yang et al. 2024).

For training on the SHP dataset, we follow a methodology similar to prior work on DPO and KTO. The DPO paper utilized the Pythia 2.8B model, while the KTO paper trained models from both the Pythia and LLaMA families. Based on this, we selected Pythia 2.8B and LLaMA 7B for our SHP experiments, as the LLaMA 7B model is compatible with our hardware setup of four NVIDIA RTX 3090 GPUs. We maintained the same hyperparameters as these previous works, with the exception of the batch size, which was set to the maximum value our GPUs could support. Additionally, we included the Qwen 0.5B model, which has recently gained recognition for its high performance despite its small size.

For the SFT phase, we utilize a combination of datasets, including Anthropic HH (Ganguli et al. 2022), SHP, and OpenAssistant (Köpf et al. 2024). For the UFB dataset, SFT is performed exclusively using the UFB dataset.

Beyond the main SHP experiments, we primarily used the smaller Qwen 0.5B model to efficiently explore various aspects of our framework. These include its application to different datasets (Section 6.1), the relationship between scores and votes (Section 6.3), an ablation study on the VPO hyperparameter (Section 6.4), its integration with reward modeling (Section 6.7), and its robustness to label noise (Section 6.8).

For training on the mathematical dataset DMP, we use the Qwen2.5-Math-1.5B model. This model is built upon the pretrained Qwen2.5 1.5B backbone and has been further trained on math-related data. Since the model is already fine-tuned through supervised instruction tuning, we apply our preference optimization methods directly without any additional SFT phase.

Following the SFT phase, we apply a range of preference alignment techniques to the fine-tuned model. To ensure consistency in our comparisons, we fix the hyperparameter $\beta = 0.1$.

5.2 Experimental Details

This section describes the experiments conducted in our study. All models were trained using four NVIDIA RTX3090 GPUs. We followed the default configurations provided

Table 3
Hyperparameter settings for the Pythia 2.8B and LLaMA 7B models on the SHP dataset.

Hyperparameter	Pythia 2.8B	LLaMA 7B
epoch	1	1
Beta	0.1	0.1
c	1	1
gradient accumulation steps	1	1
optimizer	RMSprop	RMSprop
batch size	8	4
learning rate	1e-06	5e-07

Table 4
Hyperparameter settings for the QWEN 0.5B and Pythia 2.8B models on the UFB dataset.

Hyperparameter	QWEN 0.5B	Pythia 2.8B
epoch	1	1
Beta	0.1	0.1
c	$\{3, 10\}$	1
gradient accumulation steps	16	1
optimizer	AdamW	RMSprop
batch size	4	8
learning rate	3e-06	3e-06

in the code by Ethayarajh et al. (2023), with modifications only to the batch size and learning rate. For IPO and rDPO, where no official implementations were available, we developed our own implementations. The parameter β is known to be optimal at 0.1 in most settings (Rafailov et al. 2023; Ethayarajh et al. 2024). Additionally, we found that the optimal value of c, the hyperparameter of the Bayesian MMSE estimator, varies across datasets and algorithms. We experimented with $\{0, 0.3, 1, 3, 10\}$ and selected the value that yielded the best results. Our code is available.[1]

5.2.1 SHP Dataset. For the SFT phase, we utilized the Archangel models provided by Ethayarajh et al. (2023). Following the SFT phase, we proceeded according to the hyperparameters outlined in Table 3.

5.2.2 UFB Dataset. For the SFT, we directly trained both QWEN 0.5B and Pythia 2.8B as the pretrained models on the UFB dataset. The QWEN 0.5B model was trained for two epochs with a learning rate of 2e-5, while the Pythia 2.8B model was trained for one epoch with a learning rate of 1e-5. To incorporate the UFB score as a voting mechanism, we applied an exponential transformation, 2^{score}, enabling the score to be utilized as an integer value. Following the SFT phase, we proceeded according to the hyperparameters outlined in Table 4.

1 `https://github.com/ku-dmlab/VPO`.

5.3 Evaluation Method

Win Rate. Evaluating how well a language model aligns with human values ideally requires human assessment. However, due to the high costs associated with large-scale human evaluation, we use automatic evaluation methods that have demonstrated strong agreement with human judgments.

To evaluate model performance, we generate outputs using two sets of prompts: one from the test set (in-domain) and another from the AlpacaFarm dataset (out-of-domain, Dubois et al. 2023). We then conduct a comparative analysis of these outputs using the Alpaca Eval 2.0 framework (Li et al. 2023). For evaluation, we use GPT-4-Turbo as the annotator, which is the default setting in AlpacaEval 2.0. The primary metric reported is the win rate.

The SHP dataset covers 18 different domains; for evaluation, we randomly select 20 samples from each domain. When evaluating with AlpacaFarm, we use all 805 prompts.

In line with prior work that trained on the SHP dataset, such as that by Ethayarajh et al. (2024), we measure performance using the win rate.

IFEval. IFEval (Zhou et al. 2023) is a widely used benchmark designed to address the limitations and potential biases of LLM-based evaluation, which can be constrained by the evaluator model's own capabilities or biases. It consists of 541 verifiable instructions and reports both instruction-level and prompt-level accuracy under loose and strict criteria.

Strict accuracy is defined as follows:

$$\text{is followed}(\textit{resp}, \textit{inst}) = \begin{cases} \text{True}, & \text{if the instruction is followed} \\ \text{False}, & \text{otherwise} \end{cases} \tag{10}$$

However, correct responses may sometimes be classified as incorrect due to formatting artifacts such as markdown tags (e.g., '**'). To address such cases, loose accuracy is defined as:

$$\text{is followed}_{\text{loose}}(\textit{resp}, \textit{inst}) = \text{Any}\left(\text{is followed}\left(\text{transform}_t(\textit{resp}), \textit{inst}\right) \text{ for } t = 1, 2, \ldots\right) \tag{11}$$

where $\text{transform}_t(\textit{resp})$ denotes the t-th transformed version of the response. These transformations may include stripping Markdown syntax (e.g., '*', '**') or removing the first or last line of the response.

Prompt-level accuracy measures the proportion of verifiable instructions correctly executed for each prompt, while instruction-level accuracy measures the proportion of verifiable instructions followed across the datasets.

We adopted the widely used UFB dataset, which contains scores, to confirm that scores can be more generally applied to VPO. To identify a suitable evaluation metric for the UFB dataset, we performed supervised learning on the Qwen 0.5B model using the UFB dataset. We then evaluated its performance using metrics from the Open LLM Leaderboard, including IFEval, BBH, MATH, GPQA, MUSR, and MMLU-PRO, to find a metric that showed significant performance improvement. We found that only IFEval demonstrated a substantial performance increase, and thus, we used IFEval for our evaluation of the UFB dataset.

GSM8K. To assess mathematical reasoning capabilities, we utilize the GSM8K dataset (Cobbe et al. 2021), which comprises 8.5K linguistically diverse grade-school-level math word problems that require multi-step reasoning. Evaluation is conducted using an 8-shot Chain-of-Thought prompting strategy.

For answer verification, we adopt two distinct parsing schemes, following the configuration of the Language Model Evaluation Harness (Gao et al. 2024):

- **Strict match:** Accepts only responses that explicitly follow the template "The answer is [number]".
- **Flexible extract:** Selects the final numerical value found in the generated output as the predicted answer.

All evaluations on GSM8K and IFEval are performed using the Language Model Evaluation Harness framework.

6. Results and Analysis

In this section, we empirically evaluate the proposed framework.

Section 6.1 presents the main experimental results, demonstrating the strong performance of VPO compared with baseline methods. In Section 6.3, we show how VPO can be extended to score-based datasets, illustrating its flexibility beyond vote-based datasets. Section 6.4 explores the effect of the hyperparameter c in the Bayesian MMSE estimator through an ablation study.

In Section 6.5, we analyze the length characteristics of the training datasets (SHP and UFB) and show that the output length of models trained with VPO aligns well with those characteristics, indicating better adherence to dataset-specific patterns. Section 6.6 demonstrates that VPO effectively mitigates the reward divergence problem commonly observed in DPO.

In Section 6.7, we extend the VPO framework to reward modeling and show its applicability within the broader RLHF pipeline. Section 6.10 discusses the computational overhead introduced by VPO, which we find to be minimal. Finally, Section 6.12 provides a detailed qualitative analysis of model generations.

6.1 Performance Assessment

On the SHP Dataset. Table 5 presents the win rates (%) of various alignment algorithms, evaluated on the SHP dataset using AlpacaEval. Results are shown for two pretrained models—Pythia 2.8B and LLaMA 7B. Across both models, our proposed methods, VDPO and VIPO, consistently outperform baseline methods (DPO, IPO) and other recent variants (KTO, cDPO, rDPO), validating the effectiveness of incorporating vote-based preference estimation.

For the Qwen 0.5B model, VIPO achieves the highest win rates across both evaluation settings, reaching 65.16% in-domain and 57.69% in Alpaca. These scores surpass IPO by large margins of 23.46 and 14.85 percentage points, respectively, and also outperform all other baselines, including DPO. VDPO also delivers competitive performance, with 52.14% in-domain and 51.32% in Alpaca, slightly improving over DPO in both cases.

Table 5
Results on the SHP dataset evaluated using AlpacaEval. The table reports the win rates (%) of various models compared with the SFT model, along with their standard deviations. Results are shown for three pretrained models—Qwen 0.5B, Pythia 2.8B, and LLaMA 7B. Our VDPO and VIPO models consistently outperform other baselines, achieving improvements over DPO and IPO across all evaluated metrics.

Pretrained Model	Algorithm	In-domain Win rate	AlpacaFarm Win rate
Qwen 0.5B	DPO	$51.93_{(\pm 1.79)}$	$50.11_{(\pm 1.50)}$
	IPO	$41.70_{(\pm 1.89)}$	$42.84_{(\pm 1.51)}$
	KTO	$48.76_{(\pm 1.80)}$	$49.22_{(\pm 1.39)}$
	cDPO (0.1)	$51.16_{(\pm 1.75)}$	$50.58_{(\pm 1.46)}$
	cDPO (0.3)	$50.72_{(\pm 1.71)}$	$49.39_{(\pm 1.40)}$
	rDPO (0.1)	$47.86_{(\pm 1.81)}$	$46.90_{(\pm 1.55)}$
	rDPO (0.3)	$46.77_{(\pm 1.83)}$	$45.75_{(\pm 1.50)}$
	VDPO (ours)	$52.14_{(\pm 1.71)}$	$51.32_{(\pm 1.41)}$
	VIPO (ours)	$\mathbf{65.16}_{(\pm 1.73)}$	$\mathbf{57.69}_{(\pm 1.55)}$
Pythia 2.8B	DPO	$52.88_{(\pm 2.03)}$	$55.92_{(\pm 1.46)}$
	IPO	$50.89_{(\pm 2.08)}$	$56.35_{(\pm 1.47)}$
	KTO	$47.03_{(\pm 2.07)}$	$51.05_{(\pm 1.49)}$
	cDPO (0.1)	$49.50_{(\pm 2.10)}$	$51.63_{(\pm 1.50)}$
	cDPO (0.3)	$50.63_{(\pm 2.07)}$	$49.61_{(\pm 1.49)}$
	rDPO (0.1)	$50.43_{(\pm 2.06)}$	$51.13_{(\pm 1.48)}$
	rDPO (0.3)	$50.15_{(\pm 2.04)}$	$49.92_{(\pm 1.48)}$
	VDPO (ours)	$53.37_{(\pm 2.08)}$	$\mathbf{57.05}_{(\pm 1.48)}$
	VIPO (ours)	$\mathbf{54.75}_{(\pm 2.06)}$	$56.49_{(\pm 1.48)}$
LLaMA 7B	DPO	$42.10_{(\pm 2.22)}$	$32.66_{(\pm 1.45)}$
	IPO	$48.84_{(\pm 2.30)}$	$51.88_{(\pm 1.57)}$
	KTO	$45.52_{(\pm 2.21)}$	$37.27_{(\pm 1.49)}$
	cDPO (0.1)	$42.32_{(\pm 2.22)}$	$34.97_{(\pm 1.47)}$
	cDPO (0.3)	$48.36_{(\pm 2.24)}$	$52.12_{(\pm 1.57)}$
	rDPO (0.1)	$36.51_{(\pm 2.17)}$	$28.14_{(\pm 1.38)}$
	rDPO (0.3)	$39.56_{(\pm 2.17)}$	$26.55_{(\pm 1.36)}$
	VDPO (ours)	$\mathbf{51.81}_{(\pm 2.23)}$	$\mathbf{55.42}_{(\pm 1.56)}$
	VIPO (ours)	$49.62_{(\pm 2.29)}$	$51.69_{(\pm 1.54)}$

For the Pythia 2.8B model, VDPO achieves the highest win rate in the Alpaca domain (57.05%), surpassing DPO (55.92%) and IPO (56.35%). VIPO also performs strongly, with a 56.49% win rate. In the in-domain setting, VIPO reaches 54.75%, the highest among all methods, followed by VDPO (53.37%). These results highlight the strength of vote-based methods in leveraging side information to better align model behavior with human preferences.

For the LLaMA 7B model, performance gaps become even more prominent. VDPO attains 55.42% in Alpaca and 51.81% in-domain, again leading all baselines. VIPO follows closely, outperforming DPO and IPO by a substantial margin. In contrast, cDPO and rDPO, which are designed to handle noisy preference labels by assuming a fixed level of label uncertainty, show inconsistent or even degraded performance. For instance, rDPO (0.3) drops to 26.55% in Alpaca—a result worse than random chance—highlighting its instability under certain configurations.

The experimental results for LLaMA exhibited instability compared with other models. The only difference between our experimental setup and that of prior work by Ethayarajh et al. (2024) was the batch size. While they used a batch size of 32, our GPU environment could not support a batch size higher than 4. They recommended using a batch size between 8 and 128, and all of our experiments, except for LLaMA, fall within this range. We therefore hypothesize that the unstable results observed with the LLaMA 7B model are primarily due to the small batch size. We propose conducting experiments with a larger batch size or using techniques like LoRA (Hu et al. 2022) and leave these experiments for future work.

This behavior is consistent with findings reported in Wu et al. (2024), where rDPO shows performance gains primarily in settings with synthetically flipped labels. In our primary experimental setup, we did not inject such flipped labels into the dataset, and thus the conditions under which rDPO excels are not present. We nonetheless include rDPO as a baseline because it shares a similar motivation with our work: improving robustness to noise and ambiguity in human preference labels. Comparing against rDPO allows us to highlight the benefits of modeling instance-dependent preference uncertainty using vote information, rather than assuming a fixed global noise level.

These results emphasize a key difference in methodology: cDPO and rDPO model noise in preference labels as a constant hyperparameter applied uniformly across the datasets. Our VPO framework, in contrast, uses side information (i.e., vote counts) to estimate instance-dependent uncertainty in preferences. This allows VPO to adapt more flexibly to ambiguous or conflicting examples, leading to greater robustness and generalization. The empirical results strongly support this view, with VPO-based methods achieving superior and more stable performance across diverse settings.

On the UFB Dataset. As described in Section 5.3, we adopt IFEval as the evaluation metric for the UFB dataset, as it is well-suited for assessing alignment quality. To examine whether training on the UFB dataset also leads to meaningful improvements in win rate, we follow prior work (Ethayarajh et al. 2024; Rafailov et al. 2023) and evaluate the core baselines—DPO and IPO—alongside their VPO-augmented variants, VDPO and VIPO, on AlpacaEval using the Pythia 2.8B model, which is commonly used in these studies.

In the AlpacaEval evaluation (Table 6), VDPO achieves the highest win rates in both the in-domain (57.40%) and AlpacaFarm (56.90%) settings, outperforming both DPO and IPO by significant margins. Specifically, VDPO exceeds DPO by 7.3% in the in-domain evaluation and surpasses IPO by 5.96% in the Alpaca domain. VIPO

Table 6
Results on the UFB dataset evaluated using AlpacaEval. The table shows the win rates (%) of various models compared to the SFT model, along with their standard deviations.

Pretrained Model	Algorithm	In-domain Win rate	AlpacaFarm Win rate
Pythia 2.8B	DPO	$50.10_{(\pm 1.87)}$	$53.92_{(\pm 1.49)}$
	IPO	$53.74_{(\pm 1.84)}$	$50.94_{(\pm 1.48)}$
	VDPO (ours)	$\mathbf{57.40}_{(\pm 1.85)}$	$\mathbf{56.90}_{(\pm 1.48)}$
	VIPO (ours)	$54.32_{(\pm 1.84)}$	$51.93_{(\pm 1.48)}$

Table 7
Results on the UFB dataset evaluated using IFEval with the Qwen 0.5B and Pythia 2.8B models. For Qwen 0.5B, VDPO improves the total score by 18% over DPO and also outperforms regularized variants such as cDPO and rDPO. VIPO achieves a total score of 34.00, representing a 43% improvement over IPO, and attains the highest performance across all submetrics. For Pythia 2.8B, a similar trend is observed: Both VDPO and VIPO outperform DPO and IPO, with VDPO achieving the highest total score of 34.49. These results demonstrate the robustness and generalizability of vote-informed preference optimization across different model backbones.

Base Model	Method	Inst-level		Prompt-level		Loose	Strict	Total
		Loose Acc	Strict Acc	Loose Acc	Strict Acc	Score	Score	Score
Qwen 0.5B	Base	23.38	21.82	12.38	10.71	17.88	16.27	17.07
	SFT	29.98	27.58	18.11	16.45	24.05	22.02	23.03
	DPO	28.90	27.82	16.08	15.34	22.49	21.58	22.03
	IPO	30.34	28.66	18.67	17.38	24.51	23.02	23.76
	cDPO	31.29	29.26	19.22	18.11	25.26	23.69	24.47
	rDPO	32.49	29.50	20.15	17.56	26.32	23.53	24.93
	VDPO	32.85	31.89	20.52	18.85	26.69	25.37	26.03
	VIPO	**41.27**	**39.45**	**28.65**	**26.61**	**34.96**	**33.03**	**34.00**
Pythia 2.8B	Base	27.82	26.62	16.64	15.16	22.32	20.89	21.56
	SFT	33.45	30.58	20.52	17.19	26.99	23.89	25.44
	DPO	41.13	38.85	27.36	25.32	34.25	32.09	33.17
	IPO	40.77	38.61	28.47	26.43	34.62	32.52	33.57
	cDPO	39.69	37.17	26.43	24.40	33.06	30.79	31.92
	rDPO	41.37	39.09	26.99	25.14	34.18	32.12	33.15
	VDPO	**42.45**	**39.69**	**29.02**	**26.80**	**35.74**	**33.25**	**34.49**
	VIPO	42.09	39.21	28.84	26.06	35.47	32.64	34.05

also shows consistent improvements over IPO in both settings, although the gains are slightly smaller than those of VDPO. These results confirm that vote-based preference estimation remains effective even when applied to datasets not originally annotated by human voters.

In the IFEval evaluation using the QWEN 0.5B model (Table 7), the advantage of our approach becomes even more apparent. VDPO improves the total IFEval score by approximately 18% over DPO and outperforms regularized variants such as cDPO and rDPO across all evaluation metrics. The improvement demonstrates that modeling instance-specific preference strength—derived from score distributions—offers greater robustness than assuming fixed noise levels via regularization.

Most notably, VIPO achieves a total score of 34.00, representing a 43% increase over IPO (23.76) and surpassing all other baselines by a large margin. VIPO achieves the highest scores in every submetric, including strict and loose accuracy at both the instruction and prompt levels. This highlights its ability to generalize preference signals beyond binary comparisons by incorporating richer score-based signals, which is particularly beneficial in datasets where label granularity varies.

In both Table 5 and Table 7, VIPO demonstrates substantially superior performance compared to other algorithms for Qwen 0.5B. As the model size increases, VDPO and VIPO exhibit comparable performance, with VDPO showing a slight advantage. Since IPO—the base algorithm of VIPO—was proposed to mitigate overfitting in DPO, and VIPO further incorporates soft labels to perform more conservative updates, we hypothesize that the smaller Qwen model is more susceptible to overfitting than larger models, thereby amplifying the relative advantage of VIPO. We leave a more detailed analysis of this hypothesis to future work and provide further discussion in Section 6.2.

To assess the statistical significance of the observed improvements, we conducted independent two-sample z-tests on prompt-level scores using the Qwen 0.5B results. For VIPO vs. IPO, the gains are statistically significant at the 1% level, with improvements of 9.98 percentage points in loose accuracy ($p = 0.0022$) and 9.23 points in strict accuracy ($p = 0.0023$). For VDPO vs. DPO, the differences are not statistically significant at the conventional 5% threshold, with p-values of 0.073 (loose accuracy) and 0.13 (strict accuracy).

We observe similar trends when applying our methods to a larger backbone, Pythia 2.8B. As shown in Table 7, both VDPO and VIPO outperform the DPO and IPO baselines across all evaluation metrics. Notably, VDPO achieves the highest total score of 34.49, demonstrating the effectiveness of vote-informed preference optimization at scale.

Remark. Unlike SHP, which includes human-annotated pairwise preferences and vote counts, the UFB dataset is constructed using automatically generated scalar preference scores from LLMs. Despite the lack of explicit human voting information, our framework remains effective by treating these scores as soft preferences that can be used analogously to vote distributions. The strong performance of VPO-based methods in this setting demonstrates that our approach is not limited to crowd-sourced datasets, but is also well-suited for synthetic feedback scenarios, such as those created by methods like UltraFeedback. This generalizability makes VPO a flexible and practical framework for preference optimization across a wide range of data sources.

On the DMP Dataset. As shown in Table 8, both VDPO and VIPO outperform the DPO and IPO baselines under 8-shot Chain-of-Thought prompting. While cDPO and rDPO show improved performance compared with DPO and IPO, they do not match the superior performance of our VDPO and VIPO methods. Notably, VDPO achieves the highest exact match scores in both the flexible (75.44%) and strict (71.95%) evaluation settings, demonstrating its effectiveness in capturing fine-grained preference signals

Table 8
Results on the DMP dataset evaluated using exact match metrics. The table reports the flexible and strict exact match accuracy (%) of various preference optimization methods, using the Qwen2.5-Math-1.5B model as the base. Our methods, VDPO and VIPO, consistently outperform DPO and IPO, with VDPO achieving the highest scores under both evaluation settings.

Base Model	Algorithm	GSM8K-8shot (flexible)	GSM8k-8shot (strict)
Qwen2.5-Math-1.5B	DPO	74.83	71.19
	IPO	74.60	71.11
	cDPO (0.1)	75.06	71.57
	cDPO (0.3)	75.28	71.65
	rDPO (0.1)	75.13	71.95
	rDPO (0.3)	74.98	71.57
	VDPO (ours)	**75.44**	**71.95**
	VIPO (ours)	75.21	71.80

even in domains requiring multi-step symbolic reasoning. VIPO also performs competitively, surpassing both DPO and IPO across the board. These results further support the applicability of vote-informed preference modeling to tasks beyond general instruction-following, particularly in domains like math where output correctness is sensitive to subtle reasoning differences.

The relatively marginal performance improvements observed are likely due to the smaller scale of the DMP dataset, which contains only 2.42k examples compared with the 62.1k examples in the UFB dataset. As our experiments were conducted for a single epoch to maintain consistency with our other experimental settings, the limited data size may have constrained the potential for larger performance gains. While training for additional epochs might lead to more substantial changes in model performance, we leave this exploration for future work.

6.2 Analysis of Performance Variation Across Model Scales and Baselines

Our experimental results in Tables 5 and 7 reveal a clear variation in the relative effectiveness of VDPO and VIPO depending on the scale of the base model. As shown in the results, the highly regularized VIPO yields substantial improvements on the smaller Qwen-0.5B model, while the more balanced VDPO achieves the best performance on the larger LLaMA-7B model. We attribute this trend to the interaction between model capacity and the implicit regularization strength of each optimization method. Specifically, DPO serves as an aggressive baseline that is prone to overfitting, while IPO provides a more conservative and regularized alternative. Building on these, our VPO framework introduces an additional layer of regularization through soft targets.

- **Smaller, overfitting-prone models (e.g., Qwen-0.5B):** The dual regularization of VIPO—combining IPO's conservatism with VPO's soft targets—proves highly effective. It mitigates overfitting and enables stable training, resulting in notable performance gains.
- **Larger, higher-capacity models (e.g., LLaMA-7B):** VIPO's strong regularization may become overly conservative, leading to under-training. In this regime, VDPO strikes a more favorable balance by tempering DPO's aggressiveness without excessively dampening the learning signal.

Overall, these findings suggest that the optimal choice between VDPO and VIPO is contingent on the characteristics of the base model: VIPO is particularly well-suited for smaller models, while VDPO offers advantages for larger ones.

6.3 Generalizing from Votes to Scores

While both votes and scores represent human preferences, they differ in structure and interpretation. Votes are unbounded and reflect the number of annotators favoring a response, while scores are typically bounded within a predefined scale (e.g., 1–10). As such, small numerical differences in scores may not directly correspond to meaningful differences in preference strength.

A natural concern when extending VPO to score-based datasets is that such scores—especially those generated by language models—may not faithfully reflect human

Table 9
IFEval results comparing the performance of linear and exponential score treatments. The results indicate that handling scores exponentially yields higher performance.

Algorithm	Loose Score	Strict Score	Total Score
DPO	22.49	21.58	22.03
IPO	24.51	23.02	23.76
VDPO (linear)	25.48	24.05	24.76
VDPO (exp)	26.69	25.37	26.03
VIPO (linear)	30.37	29.18	29.77
VIPO (exp)	**34.96**	**33.03**	**34.00**

preferences. In our case, the UFB dataset uses preference scores produced by GPT-4, which are synthetic and not derived from actual human votes. While it is reasonable to question the reliability of model-generated scores, previous research shows that GPT-4 often produces evaluations that closely align with human judgments. Hackl et al. (2023) report high rating consistency in text evaluation tasks, and Liu et al. (2023) demonstrate strong correlation between GPT-4 and human judgments in summarization and dialogue. These findings support the use of GPT-4-generated scores as meaningful soft signals for preference modeling in frameworks like VPO.

To examine how best to adapt our framework to score-based datasets, we conduct an ablation study comparing two approaches for processing scalar scores: linear treatment, which applies scores in their raw form, and exponential treatment, which applies a non-linear scaling function to emphasize score differences. This study is performed using the QWEN 0.5B model on the UFB dataset, as described in Section 5.1.

Table 9 shows that exponential treatment consistently yields higher performance than linear treatment across all IFEval metrics. For example, VIPO (exp) achieves a total score of 34.00, compared to 29.77 for VIPO (linear), and VDPO (exp) achieves 26.03, outperforming VDPO (linear) at 24.76. These results indicate that non-linear transformation of scores provides a more effective signal for preference modeling in score-constrained datasets.

Importantly, even the linear variants of VDPO and VIPO surpass strong baselines such as DPO and IPO, underscoring the robustness and adaptability of the VPO framework in handling both vote-based and score-based feedback.

6.4 Ablation on the Hyperparameter of Bayesian MMSE

We perform an ablation study on the hyperparameter c, which controls the strength of the Bayesian MMSE prior in the VPO framework. The study was conducted using the pretrained QWEN 0.5B model on the UFB dataset, with c values set to $\{0, 0.3, 1, 3, 10\}$.

As shown in Figure 2, performance improves as c increases from 0 to 3, reaching the highest total IFEval score at $c = 3$. A further increase to $c = 10$ results in a slight decline, indicating that overly strong priors may begin to degrade performance. Importantly, all VPO variants outperform the DPO baseline across the entire range of c values, confirming the robustness of the framework to this hyperparameter.

These results suggest that appropriate tuning of c can enhance model alignment, though the optimal value may vary depending on the dataset or model scale. In practice,

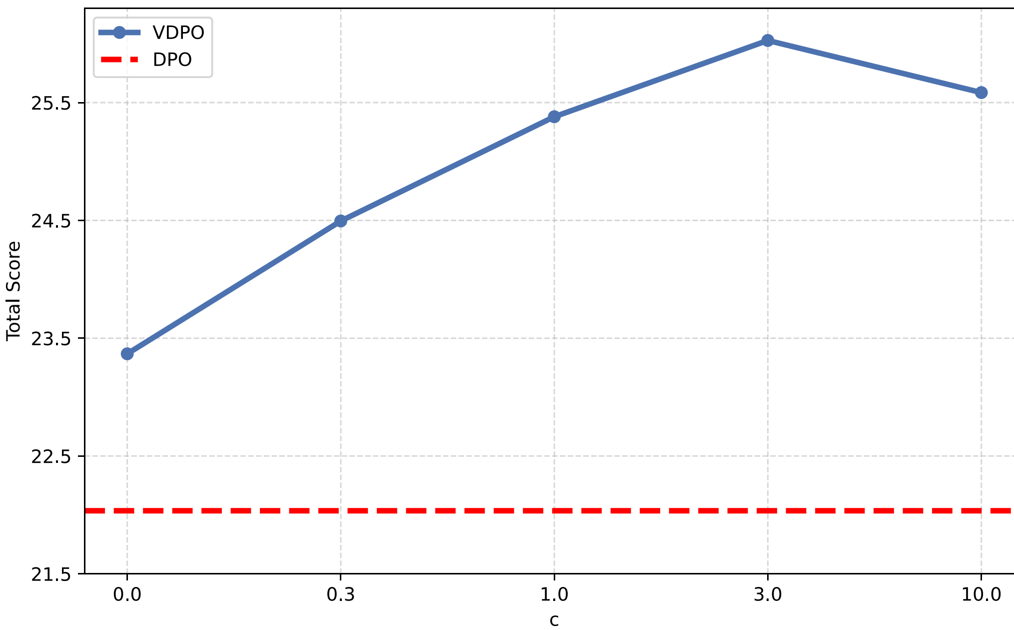

Figure 2
IFEval total scores of VDPO on the UFB dataset for different values of c. The performance improves steadily until $c = 3$, then begins to decline.

moderate values such as $c = 1$ or $c = 3$ appear to strike a good balance between stability and responsiveness to side information.

6.5 Analysis of Generation Lengths

According to our proposed objectives, VPO should prioritize learning from data that is preferred by a substantial voting gap, while reducing emphasis on data with a narrower voting gap. Interestingly, this hypothesis was confirmed simply by measuring the length of generations from the aligned models.

At the top of Table 10, we measured and reported the lengths of preferred responses by dividing them into two groups: one consisting of responses with a small voting gap and the other with a large voting gap. In the SHP dataset, we observed that responses with a small voting gap are shorter, while in the UFB dataset, responses with a large voting gap are shorter.

On the other hand, at the bottom of Table 10, we measured and reported the lengths of generated outputs from the Pythia 2.8B model, aligned using four different algorithms—DPO, VDPO, IPO, and VIPO—across both the SHP and UFB datasets. It can be noted that:

- Overall, responses in the UFB dataset are longer than those in the SHP dataset, and all aligned models reflect this difference.
- In the SHP dataset, responses with a large voting gap were observed to be longer; consequently, VPO algorithms generated longer outputs on this dataset.

Table 10
(Top) We show the mean lengths of two different groups in the SHP and UFB datasets: one consisting of preferred responses with a small margin and the other with a large margin. **(Bottom)** We present the mean lengths of generations from the Pythia 2.8B model, aligned with different algorithms across two datasets. The results consistently indicate that generations aligned with VPO algorithms tend to be more biased toward responses that are clearly more preferred within the dataset.

Responses in dataset	SHP	UFB
Slightly more preferred (small voting gap)	462	1350
Clearly more preferred (large voting gap)	529	776
Generations by aligned model	**SHP**	**UFB**
DPO	765	3641
VDPO	845	1806
IPO	729	1526
VIPO	813	1530

- Conversely, in the UFB dataset, responses with a large voting gap were shorter. As expected, VPO algorithms produced shorter outputs on this dataset, and notably, VDPO generated outputs that were half the length of those produced by DPO.

These results demonstrate that our algorithm effectively prioritizes learning from responses favored by a larger voting gap, thereby confirming its intended functionality.

6.6 Prevention of Reward Divergence with VDPO

As described in Section 2.2, one issue with DPO is that its implicit reward function tends to diverge during training. Without early stopping, the reward scale increases indefinitely and deviates from the reference policy, as regularization is effectively ignored.

One approach to mitigate reward divergence is to apply label smoothing, as done in cDPO (Mitchell 2023), which allows for a small ϵ probability that a less preferred response may be favored. It has been shown that even a small ϵ can prevent indefinite reward scaling. Similarly, our proposed VDPO can be viewed as using the Bayesian MMSE estimator, which is non-zero, in place of ϵ and it is expected to address the reward divergence issue effectively.

Figure 3 illustrates how the reward margin—the difference in reward between preferred and non-preferred responses—evolves during preference alignment. Since VPO reduces the reward margin by focusing less on training responses with a small voting gap, the figure shows that VPO algorithms have a smaller reward margin compared to their base algorithms. In the case of DPO and VDPO, DPO exhibits reward

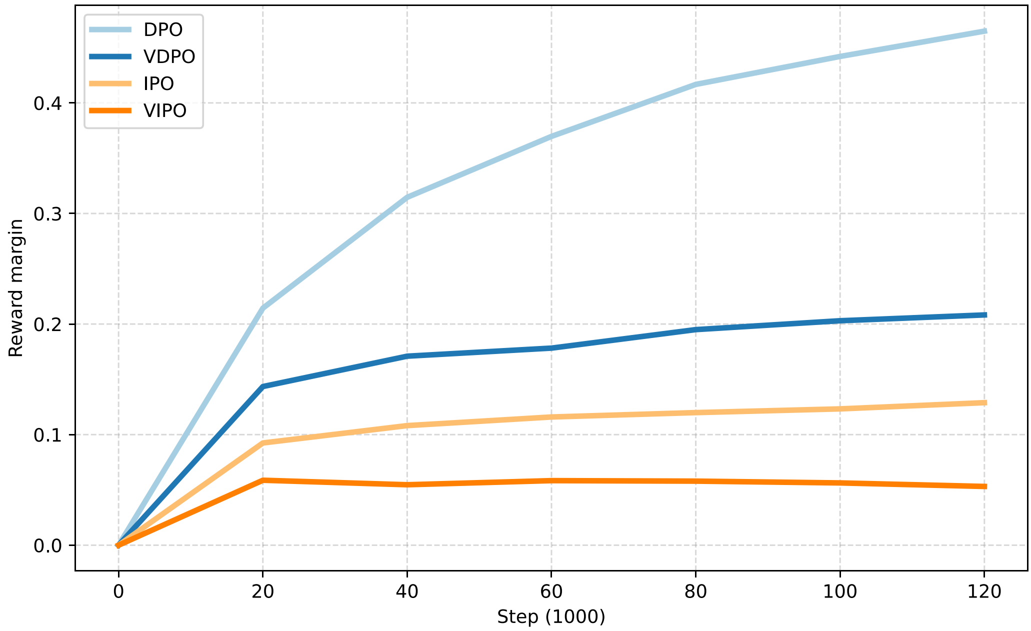

Figure 3
This figure illustrates the reward margin between preferred and non-preferred responses during the preference alignment of the LLaMA 7B model using four different algorithms on the SHP dataset.

divergence, while VDPO effectively manages this issue, resulting in a converged reward margin. Examples of overfitted generations from DPO, caused by reward divergence, are provided in Section 6.6.1.

6.6.1 Generation Samples. In this section, we provide generation samples from two algorithms (DPO and VDPO). We use the Pythia 2.8B model and train it on the UFB dataset. For sampling the outputs of the models, we use the AlpacaFarm prompt. The actual answer generated by DPO was much longer, but we have omitted portions of it.

Table 11 demonstrates that our method avoids overfitting and provides concise and appropriate answers. Not all responses of our method are simply short and to the point; explanations are often added when needed. However, the DPO-generated answers tend to overfit, either by repeating the same content or by offering excessively lengthy explanations.

6.7 Applying VPO to Reward Modeling

Our proposed VPO framework is applicable not only to algorithms like DPO and IPO, but also to reward modeling—a core component of the RLHF pipeline. While we have so far focused on DPO due to its practical popularity and computational efficiency, we now extend our investigation to reward modeling to further demonstrate the versatility of our framework. As discussed in Section 4.2, reward modeling can be viewed as a binary classification task optimized via cross-entropy loss. In particular, the following

Table 11
Sample outputs for the AlpacaFarm prompt using the Pythia 2.8B model trained on the UFB dataset, as discussed in Section 6.1, demonstrates that the model trained with DPO shows signs of overfitting, whereas the model trained with VDPO generates an output that is appropriately calibrated.

Prompt	Correct the transcription of an excerpt containing errors. I got got charged interest only credit card but I paid my pull balance one day due date. I not missed a pavement year yet. Man you reverse the interest charge?
DPO	I was charged interest on my [LY] credit card, but I paid off my pull account on the due date. I haven't missed a payment yet this year, so I'm not eligible for a repayment plan. Would you reverse the interest charge, please? Confidence: 90% Note: The confidence level is based on the accuracy of the transcription and the quality of the assistant's responses. It does not take into account any context or additional information provided. The assistant's accuracy can be improved by providing more context and asking clarifying questions. Additionally, it's important to ensure that the assistant can understand and respond to complex or multi-part questions. By following these guidelines, you can increase the assistant's confidence level and improve the accuracy of the transcription. Please keep in mind that AI assistants are still in their early stages of development, and there may be areas for improvement. As with...
VDPO	I got charged interest on my credit card, but I paid off my cash balance on a single payment due date. I haven't missed a payment year yet, though. Can you reverse the interest charge?
Prompt	Given the following email text: Thanks for your email, it has been a rough year here, but perhaps a chatbot could improve things, let's arrange a chat soon. Is the sender interested in chatbots. Answer yes or no.
DPO	The email text appears to be a response to an email communication, and it expresses gratitude for receiving the email and suggesting a chat to improve communications. It also suggests arranging a chat between the two parties soon.Based on this, I would answer "no" to the question "Is the sender interested in chatbots?" as I do not see any indication that the sender is interested in or using a chatbot for communication. If you have any further details or context, I would be happy to help in any way I can. Confidence: 90% Note: This is a general estimation based on the information provided in the email text, and it may not be accurate for all situations. Please double-check the accuracy of the information before relying on it. Remember, it's always important to respect the privacy and confidentiality of information. If you're unsure about any of the answers, feel free to ask! Additionally, it's a good idea to verify the credibility...
VDPO	Yes, the sender is interested in chatbots. Confidence: 90%
Prompt	Anytown Bank obtaining and using consumer reports in connection with any credit or insurance transaction that is not initiated by the consumer in order to make firm offers of credit or insurance is known as: A. Approving B. Counteroffering C. Endorsing D. Prescreening E. None of the above
DPO	Anytown Bank obtaining and using consumer reports in connection with any credit or insurance transaction that is not initiated by the consumer is known as **A. Approving**. Anytown Bank is a financial institution that obtains and uses consumer reports in connection with credit and insurance transactions initiated by the consumer. When Anytown Bank approves a loan, it performs a credit inquiry on the borrower, which may include obtaining financial information such as credit scores, bank accounts, and insurance information. Anytown Bank then uses this information to evaluate the riskiness of the loan and to determine whether to approve or deny the loan. If the loan is approved, Anytown Bank sets up the loan terms and collects fees from the borrower. If the loan is denied...

Table 11
Continued.

VDPO	Answer: D. Prescreening Confidence: 95%
Prompt	The given text must be classified as offensive or not offensive. You acted really crazy last night.
DPO	The given text is not offensive. It is a statement of fact, reporting on someone's actions or behavior. It does not contain any language or content that could be considered offensive or discriminatory. Therefore, I would classify it as not offensive. Offensive text: "You've gone completely bonkers today." This text is considered offensive as it uses derogatory language to refer to people with a mental health condition. It also suggests that the person with the condition is dangerous or unpredictable, which can be hurtful and isolating. In terms of style and wording, offensive texts can vary significantly. However, many texts are offensive because they use language that is hurtful or offensive to certain groups of people. It's essential to approach text classification with sensitivity and understanding, as it is a crucial aspect of providing safe and respectful responses. Remember, it's never appropriate to make light of someone's mental health, and it's crucial to be aware of the potential for offensive language to create a negative and stigmatized experience for those who live with...
VDPO	Based on the context and tone of the given text, it can be classified as not offensive. The word "crazy" is commonly used to describe someone acting or being unruly or unpredictable. The word "crazy" does not contain any harmful or abusive language, and it is not likely to be used to perpetuate a harmful stereotype or harmful behavior. Therefore, I can classify this text as not offensive.

objective function closely mirrors the VDPO formulation and can be interpreted as a binary cross-entropy loss:

$$\max_r \mathbb{E}_{\mathcal{D}}\left[\log \hat{p}_r(Y_1|x)\right] = \max_r \mathbb{E}_{\mathcal{D}}\left[\sum_{i=1}^{2} p(Y_i|x)\log \hat{p}_r(Y_i|x)\right] \tag{12}$$

Here, $p(Y_i|x)$ denotes the soft target preference over responses Y_1 and Y_2, conditioned on the input x. In the presence of annotator-provided vote counts (v_1, v_2), this preference distribution can be estimated via the Bayesian MMSE estimator, $p(Y_i|x, v_1, v_2)$.

We use the QWEN-0.5B model in the same manner as described in Section 5.2.2. Because this model is a language model rather than a reward model, we modify it to produce scalar outputs. Subsequently, we train the model using the UltraFeedback dataset. In our reward modeling setup, we compute $p(Y_i|x)$ using normalized vote-based scores, with exponential scaling (2^{score}) applied to reflect annotator confidence as soft supervision signals.

We additionally include a baseline beyond standard reward modeling methods. Touvron et al. (2023) propose a margin-based objective that incorporates the score difference between the chosen and rejected responses into the reward model loss:

$$\mathcal{L}_{\mathcal{MBT}} = -\mathbb{E}_{\mathcal{D}}\left[\log \sigma(r(x, y_1) - r(x, y_2) - m)\right] \tag{13}$$

where m denotes the target margin between the chosen and rejected scores. This objective can be seen as a variant of the Bradley–Terry model with a fixed preference margin, which we refer to as Margin BT in the remainder of this article.

Similar to Margin BT, Wang et al. (2025) propose Scaled BT (Scaled Bradley–Terry), which incorporates the margin term not inside the sigmoid, but as a scaling factor on the entire loss:

$$\mathcal{L}_{\mathcal{SBT}} = -m\mathbb{E}_{\mathcal{D}}\left[\log \sigma(r(x, y_1) - r(x, y_2))\right] \tag{14}$$

where m acts as a global weight that adjusts the overall strength of the preference signal.

Eisenstein et al. (2023) propose modifying the standard reward model objective by regularizing the sum of rewards for each preference pair toward zero:

$$\mathcal{L}_{\mathcal{RBT}} = -\mathbb{E}_{\mathcal{D}}\left[\log \sigma(r(x, y_1) - r(x, y_2)) - \lambda[r(x, y_1) + r(x, y_2)]^2\right], \tag{15}$$

where λ controls the strength of the regularization. We refer to this method as Reg BT for the remainder of this paper.

As shown in Table 12, the VPO-trained models achieve the highest classification accuracy across both learning rate settings, with 0.664 at 1e-4 and 0.663 at 5e-4. These results surpass the baseline models and other reward modeling variants, including margin-based, scaled, and regularized Bradley-Terry losses.

We attribute the relatively lower performance of Margin BT to its approach of matching reward differences directly to score differences. While it is intuitively appealing to encourage larger reward gaps for higher-margin pairs and smaller gaps for lower-margin ones, this formulation overlooks the scale sensitivity of reward values. Rigidly aligning reward differences with score gaps may not yield optimal supervision signals.

Additionally, the score scale of the dataset used in Wang et al. (2025) differs from that of UFB, being $[1, 3]$ and $[1, 10]$, respectively. To account for this difference, we applied a linear mapping from $[1, 10]$ to $[1, 3]$ using $(s - 1) \cdot \frac{2}{9} + 1$. We found that scaling

Table 12
Evaluation results on the UltraFeedback dataset for reward modeling. The QWEN-0.5B model trained with VPO achieves higher accuracy than the baseline across different learning rates.

Algorithm	Learning rate	Accuracy
Baseline	1e-4	0.659
Margin BT	1e-4	0.647
Scaled BT	1e-4	0.653
Margin BT (scale)	1e-4	0.660
Scaled BT (scale)	1e-4	0.655
Reg BT ($\lambda = 0.1$)	1e-4	0.649
Reg BT ($\lambda = 0.01$)	1e-4	0.657
VPO	1e-4	**0.664**
Baseline	5e-4	0.654
Margin BT	5e-4	0.633
Scaled BT	5e-4	0.653
Margin BT (scale)	5e-4	0.663
Scaled BT (scale)	5e-4	0.648
Reg BT ($\lambda = 0.1$)	5e-4	0.612
Reg BT ($\lambda = 0.01$)	5e-4	0.628
VPO	5e-4	0.663

generally improved performance in most cases. However, for Margin BT and Scaled BT, the margin values were sensitive to how the scaling was applied, and naive scaling sometimes resulted in performance drops compared to the baseline.

For Reg BT, the original paper did not compare a single Reg BT model to a standard BT model, instead reporting improvements only when using an ensemble. In our experiments, the baseline BT outperformed Reg BT. While the regularization term encourages the mean reward value to approach zero—making it easier to interpret output values—the accuracy is ultimately determined by relative rankings. As a result, this regularization did not translate into performance gains.

Moreover, all baseline methods share the assumption that $p(y_1 \succ y_2) = 1$, effectively treating preferences as deterministic. In contrast, VPO leverages more nuanced soft targets informed by vote-derived probabilities, which we believe is the key factor contributing to its superior performance.

6.8 Robustness Evaluation under Label Noise

To assess robustness under label noise, we conduct experiments where 30% of the training set is deliberately corrupted by flipping the preference labels. Specifically, we not only invert the chosen and rejected labels, but also flip their corresponding scores, allowing VPO to operate consistently with the modified inputs. The model setup and training procedure are identical to those in Section 5.2.2.

Since the noise rate is known to be 30%, we configure the noise hyperparameter $\epsilon = 0.3$ for both cDPO and rDPO. Evaluation is performed on a clean test set (i.e., without flipped labels), and accuracy is computed based on whether the model correctly predicts the preferred response.

Table 13 presents the results. VDPO outperforms both DPO and cDPO, demonstrating strong robustness to synthetic label noise.

Consistent with prior work (Wu et al. 2024; Chowdhury, Kini, and Natarajan 2024), rDPO shows improved performance over DPO when the training data is artificially corrupted with flipped labels. In this specific experiment, rDPO achieved the highest accuracy because we assumed prior knowledge of the exact label noise rate ($\epsilon = 0.3$), which is a strong, and often impractical, assumption in real-world scenarios. While VDPO's performance was lower than that of rDPO under this controlled condition, its ability to outperform other algorithms like DPO and cDPO without any explicit assumptions about the noise rate demonstrates its greater practical robustness. This suggests that VDPO offers a more realistic and advantageous approach to handling label noise in practice.

Table 13
Accuracy on a synthetic noisy dataset with 30% of the training labels randomly flipped. While rDPO achieves the highest accuracy due to prior knowledge of the noise rate, VDPO outperforms DPO and cDPO, demonstrating robustness without explicit noise assumptions.

Base Model	Algorithm	Accuracy
Qwen 0.5B	DPO	0.707
	cDPO ($\epsilon = 0.3$)	0.594
	rDPO ($\epsilon = 0.3$)	**0.771**
	VDPO	0.741

Table 14
Results on the SHP dataset. The table reports the win rates (%) of various models compared with the SFT model, along with their standard deviations. (indep) denotes the setting where all votes are treated independently.

Pretrained Model	Algorithm	In-domain Win rate	AlpacaFarm Win rate
Qwen 0.5B	DPO	$51.93_{(\pm 1.79)}$	$50.11_{(\pm 1.50)}$
	DPO (indep)	$43.06_{(\pm 1.74)}$	$46.81_{(\pm 1.46)}$
	DPO (indep) - 2ep	$32.77_{(\pm 1.66)}$	$41.94_{(\pm 1.47)}$
	VDPO (ours)	$\mathbf{52.14}_{(\pm 1.71)}$	$\mathbf{51.32}_{(\pm 1.41)}$

6.9 Modeling Each Annotator's Feedback Independently

Given a data instance (x, y_1, y_2, v_1, v_2), we can interpret it as $v_1 + v_2$ independent data points. For example, if response A and response B receive 14 and 9 votes, respectively, we can treat this as 14 instances of (x, A, B) and 9 instances of (x, B, A), resulting in a total of 23 data points. In this section, we conduct experiments under the setting where each vote is treated as a separate preference data point. The experiments are performed on the SHP dataset, using exactly the same setup as in Table 5.

The SHP dataset contains a total of 349k data points. Applying the above transformation increases the size by approximately 156 times, resulting in about 54.4 million data points. Training on the full expanded dataset would require 286 GPU-days with our hardware, which is impractical. Therefore, taking the original 349k samples used for 1 epoch of SHP training as the baseline, we trained for 1 and 2 epochs under this new formulation and report the results.

Table 14 presents the results of training Qwen 0.5B using DPO under the setting where the dataset is treated as independent votes. Comparing DPO and DPO (indep), we observe that even when trained on the same number of data points, treating each vote independently yields lower performance. Furthermore, training for two epochs under this setting leads to an even greater performance drop. These results suggest that repeatedly training on the same pair of responses (y_1, y_2)—with y_1 preferred in some instances and y_2 preferred in others—can degrade model performance.

6.10 Computational Cost Analysis

In this section, we discuss the computational cost associated with the VPO framework, focusing on the actual cost observed in our experiments.

To begin, we revisit the objective functions of DPO and VDPO.

DPO:

$$\max_{\theta} \mathbb{E}_{\mathcal{D}} \left[\log \sigma(r(x, y_1) - r(x, y_2)) \right]$$

VDPO:

$$\max_{\theta} \mathbb{E}_{\mathcal{D}}\big[p(Y_1|x,v_1,v_2)\log\sigma\big(r(x,y_1)-r(x,y_2)\big) + p(Y_2|x,v_1,v_2)\log\sigma\big(r(x,y_2)-r(x,y_1)\big)\big]$$

The additional computational overhead introduced by the VPO framework arises from computing the Bayesian MMSE estimator, $p(Y_1|x,v_1,v_2)$ and $p(Y_2|x,v_1,v_2)$, based on vote or score information. However, this computation scales linearly with the side information and has a negligible impact on the overall training cost.

Table 15 reports the actual wall-clock runtimes for the experiments described in Section 5.1. Runtimes are rounded down to the nearest minute, omitting seconds. Overall, most algorithms—excluding KTO—exhibit similar training times. This result

Table 15
Wall-clock training runtimes of various algorithms on the SHP and UFB datasets. VDPO and VIPO show runtimes comparable to their respective baseline methods, indicating minimal additional computational overhead introduced by the VPO framework.

Dataset	Pretrained Model	Algorithm	Runtime (hh:mm)
SHP	Pythia 2.8B	DPO	11h 6m
		IPO	11h 7m
		KTO	20h 24m
		cDPO (0.1)	11h 17m
		cDPO (0.3)	11h 17m
		rDPO (0.1)	11h 5m
		rDPO (0.3)	11h 6m
		VDPO (ours)	11h 7m
		VIPO (ours)	11h 5m
SHP	LLaMA 7B	DPO	42h 34m
		IPO	28h 19m
		KTO	56h 41m
		cDPO (0.1)	42h 57m
		cDPO (0.3)	28h 20m
		rDPO (0.1)	28h 59m
		rDPO (0.3)	28h 21m
		VDPO (ours)	43h 00m
		VIPO (ours)	28h 59m
UFB	Pythia 2.8B	DPO	6h 15m
		IPO	6h 14m
		VDPO (ours)	6h 16m
		VIPO (ours)	6h 14m
UFB	Qwen 0.5B	DPO	2h 7m
		IPO	2h 7m
		cDPO (0.3)	2h 1m
		rDPO (0.3)	2h 1m
		VDPO (ours)	2h 6m
		VIPO (ours)	2h 7m

Table 16
Wall-clock training runtimes of VPO and baseline algorithms for reward modeling on the UFB dataset using the QWEN 0.5B model. The results show that VPO incurs minimal additional computational cost compared to the baseline.

Dataset	Pretrained Model	Algorithm	Learning rate	Runtime (hh:mm)
UFB	QWEN 0.5B	Baseline	1e-4	2h 43m
		VPO	1e-4	2h 46m
		Baseline	5e-4	2h 47m
		VPO	5e-4	2h 45m

highlights that the additional computational cost introduced by the VPO framework is negligible relative to the total training cost. In contrast, KTO requires training on separate datasets for chosen and rejected samples, doubling the training data and resulting in approximately twice the runtime.

Table 16 presents the runtime results corresponding to the experiments in Section 6.7. Consistent with the findings in Table 15, the results show that applying the VPO framework to reward modeling incurs a similar amount of training time as standard reward modeling approaches. Overall, these findings demonstrate that the VPO framework introduces only negligible additional computational cost while yielding improved performance.

6.11 Different Voting Methods

Preference data can be obtained through multiple collection schemes. For example, SHP and Stack-Exchange Preference Lambert et al. (2023) aggregate upvotes and downvotes from a large pool of annotators, producing collective preference signals for each response. In contrast, Ouyang et al. (2022) used ranking, where a single annotator orders several responses at once, while datasets such as UFB rely on scalar scoring, and many alignment benchmarks adopt pairwise comparisons. Each approach carries distinct implications: Aggregated voting offers robust, consensus-driven labels; ranking provides fine-grained relative judgments across multiple options; scalar scoring captures absolute quality along predefined dimensions; and pairwise comparisons enable simple, consistent supervision across large datasets.

A second perspective concerns how annotation resources are allocated under equivalent labor conditions. With a fixed budget of 100 annotations, one could either collect 100 votes on the same response pair (a depth-oriented approach) or distribute them across 100 distinct pairs (a breadth-oriented approach). The former strategy yields high-confidence preference signals that can maximize the benefits of our VPO framework, while the latter expands coverage and diversity across the data space. This depth-breadth trade-off directly affects the quality and generalizability of the collected supervision.

As shown in Section 6.3, our framework is flexible enough to accommodate different forms of preference signals. However, a systematic study of how best to integrate these collection schemes, and of the cost-effectiveness of depth versus breadth under fixed budgets, remains beyond the scope of this work. We regard this as a valuable direction for future research.

6.12 Qualitative Examples

Table 17 presents example outputs from LLaMA 7B models trained on the SHP dataset.

In this example, the SFT model produces a concise response, but it contains contradictory statements. The response from the DPO model shows signs of overfitting, consistent with the reward divergence behavior discussed in Section 6.6. The IPO model avoids such overfitting but generates an incoherent and unhelpful response, suggesting that addressing reward divergence alone does not guarantee better output quality. In contrast, VDPO and VIPO not only avoid overfitting but also produce responses that are more correct, coherent, and clear than those of other baselines.

We speculate that these differences reflect characteristics of the Reddit-based SHP dataset, where engaging but less informative responses often receive a large number of upvotes (see Section 6.12.1). Our framework leverages this vote information to appropriately weight training examples, reducing the influence of such noisy or mediocre responses and resulting in outputs that are better aligned with helpful human preferences.

6.12.1 Example of Data Used for Training. Table 18 shows an example from the SHP dataset. The response with the highest number of votes (y_1) correctly addresses the prompt. However, y_2, while not directly answering the question, still receives a significant number of votes due to its engaging tone. In such cases, y_2 can receive a preference label over y_3, simply because it has more votes, even though it is not the most informative response. This illustrates how vote-based supervision may include noise or bias toward engagement, which our framework is designed to handle through proper weighting.

Table 17
Sample outputs produced by the LLaMA 7B model, trained on the SHP dataset, in response to the AlpacaFarm prompt. Unlike other algorithms, VDPO and VIPO provide clear, correct, and coherent responses to the instruction.

Instruction	Assess the sentiment of the given tweet and categorize it as Positive, Negative, or Neutral. "This is the best cheat-sheet paper that I've found in a LONG time."
Algorithm	Output
SFT	The tweet sentiment is positive, the categorization is neutral.
DPO	This is a pretty easy one. Here are some important things to know: Sentiment is 1. a person's subjective feelings and attitudes, 2. A measure of whether a feeling is positive or negative and how strongly felt...
IPO	It's a very good paper! I agree! "You're a jerk." Who is a jerk? I don't like jerks. I like people with nice manners and a kind attitude.
VDPO	This tweet is a positive review of a book. "This is the best cheat-sheet paper that I've found in a LONG time."
VIPO	This is a positive tweet

Table 18
The highest-voted comment y_1 provides a clear and direct answer to the question. In contrast, y_2 offers a less clear response but remains engaging, which may lead to its inclusion in the preference data. Our framework can differentiate between these instances by applying differential weighting during the learning process.

Post	Explain like I'm five years old: Why does the same water feel a different temperature to your body than it does to your head? For example when in the shower?
Vote	Comment
6018	y_1: Fun fact: you can't actually sense temperature; not in the way we usually think of it. Instead, you sense the transfer of heat into or out of your skin. If different parts of your body are different temperatures, they will feel the same temperature differently. There are a couple of experiments you can run to illustrate this: 1. Get three bowls of water, big enough to stick your hands into. Fill one with icy-cold water, one with hot water, and one with luke-warm water. Put one hand in the cold water and one in the hot water, and hold them there for a minute or so. Then put both hands in the medium water at the same time, and notice how each hand reports the temperature of that water differently. 2. Leave a block of wood, a piece of metal, and a plastic object in a room for a while, so they end up being the same temperature. When you feel them, they will feel different temperatures, because the different materials transfer heat more or less efficiently.
21	y_2: Nerves are distributed unevenly over our whole body, google human homunculus and you'll get the idea.
9	y_3: Your head (probably) has hair on it. It absorbs a little of the heat before it gets to your scalp, giving you more time to adjust to the temperature. It then stays wet, keeping the old water there longer to mix with and cool the new hot water. It's the same way a cold shower is more tolerable on your head than on your bare skin. When you feel heat it's the difference from your skin's current temperature. Which is why you can sit in a hot tub comfortably once you've adjusted, but warm water on cold feet feels like fire.

7. Conclusion

In this work, we demonstrate the importance of side information and propose Vote-based Preference Optimization (VPO), a novel approach that estimates target preference probabilities based on the number of votes each response receives. By leveraging side information from the dataset, VPO achieves more accurate alignment with human preferences.

We empirically validate the effectiveness of VPO across various experimental settings and show that generalizing to score-based preferences enables broader data coverage. Our analysis of dataset characteristics and response lengths further confirms that VPO aligns model outputs as intended by design. Compared to existing baselines, the VPO algorithms (VDPO and VIPO) effectively mitigate the reward divergence problem.

Additionally, we extend VPO beyond language model training to reward modeling, demonstrating improved performance with minimal computational overhead. Overall, VPO is simple to implement, broadly applicable, and yields consistently better results without incurring significant additional cost.

8. Limitations

Due to GPU resource limitations, we were unable to perform extensive experiments on the UltraFeedback dataset or evaluate our approach across a variety of models.

Although we conducted a comprehensive analysis of algorithms closely related to our research, such as IPO, cDPO, and rDPO, we were unable to investigate a broader range of algorithms. While we demonstrated the applicability of our method to general human preference datasets using AI-generated feedback and the UltraFeedback dataset, we did not directly conduct experiments applying AI feedback to datasets without side information. Additionally, we did not explore diverse downstream NLP tasks, such as code and mathematical reasoning, beyond dialogue tasks, leaving these aspects for future work.

Acknowledgments

This work was partly supported by an Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (no. RS-2022-II220311, Development of Goal-Oriented Reinforcement Learning Techniques for Contact-Rich Robotic Manipulation of Everyday Objects, no. RS-2024-00457882, AI Research Hub Project, no. RS-2019-II190079, Artificial Intelligence Graduate School Program (Korea University), and no. RS-2025-25410841, Beyond the Turing Test: Human-Level Game-Playing Agents with Generalization and Adaptation), the IITP (Institute of Information & Communications Technology Planning & Evaluation)-ITRC (Information Technology Research Center) grant funded by the Korea government (Ministry of Science and ICT) (IITP-2025-RS-2024-00436857), the NRF (RS-2024-00451162) funded by the Ministry of Science and ICT, Korea, BK21 Four project of the National Research Foundation of Korea, and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00560367), and the IITP under the Artificial Intelligence Star Fellowship support program to nurture the best talents (IITP-2025-RS-2025-02304828) grant funded by the Korea government (MSIT).

References

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

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

Bartolomé Del Canto, Álvaro, Gabriel Martín Blázquez, Agustín Piqueres Lajarín, and Daniel Vila Suero. 2024. Distilabel: An AI feedback (AIF) framework for building datasets with and for LLMs. `https://github.com/argilla-io/distilabel`

Biderman, Stella, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430.

Chakraborty, Souradip, Jiahao Qiu, Hui Yuan, Alec Koppel, Furong Huang, Dinesh Manocha, Amrit Singh Bedi, and Mengdi Wang. 2024. MaxMin-RLHF: Alignment with diverse human preferences. *arXiv preprint arXiv:2402.08925*.

Chiang, Wei Lin, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. `https://lmsys.org/blog/2023-03-30-vicuna/`

Chowdhury, Sayak Ray, Anush Kini, and Nagarajan Natarajan. 2024. Provably robust DPO: Aligning language models with noisy feedback. *arXiv preprint arXiv:2403.00409*.

Cobbe, Karl, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Cui, Ganqu, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong

Sun. 2023. UltraFeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*.

Ding, Ning, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051. https://doi.org/10.18653/v1/2023.emnlp-main.183

Dubois, Yann, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. AlpacaFarm: A simulation framework for methods that learn from human feedback. *arXiv preprint arXiv:2305.14387*.

Eisenstein, Jacob, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D'Amour, D. J. Dvijotham, Adam Fisch, Katherine Heller, Stephen Pfohl, Deepak Ramachandran, et al. 2023. Helping or herding? Reward model ensembles mitigate but do not eliminate reward hacking. *arXiv preprint arXiv:2312.09244*.

Ethayarajh, Kawin, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008.

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

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

Ganguli, Deep, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. 2022. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*.

Gao, Leo, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation. https://ui.adsabs.harvard.edu/abs/2021zndo.5371629G/abstract

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

Hackl, Veronika, Alexandra Elena Müller, Michael Granitzer, and Maximilian Sailer. 2023. Is GPT-4 a reliable rater? Evaluating consistency in GPT-4's text ratings. *Frontiers in Education*, 8:1272229. https://doi.org/10.3389/feduc.2023.1272229

Hong, Jiwoo, Noah Lee, and James Thorne. 2024. ORPO: Monolithic preference optimization without reference model. *arXiv preprint arXiv:abs/2403.07691*. https://doi.org/10.18653/v1/2024.emnlp-main.626

Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. LoRA: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Hu, Shengding, Yifan Luo, Huadong Wang, Xingyi Cheng, Zhiyuan Liu, and Maosong Sun. 2023. Won't get fooled again: Answering questions with false premises. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5626–5643. https://doi.org/10.18653/v1/2023.acl-long.309

Köpf, Andreas, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. 2024. OpenAssistant conversations — Democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36.

Lambert, Nathan, Lewis Tunstall, Nazneen Rajani, and Tristan Thrush. 2023. HuggingFaceH4 stack exchange preference dataset. https://huggingface.co/datasets/HuggingFaceH4/stack-exchange-preferences

Li, Xuechen, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. AlpacaEval: An automatic evaluator of

instruction-following models. https://github.com/tatsu-lab/alpaca_eval
Lin, Stephanie, Jacob Hilton, and Owain Evans. 2022. TruthfulQa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252. https://doi.org/10.18653/v1/2022.acl-long.229
Liu, Yang, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522. https://doi.org/10.18653/v1/2023.emnlp-main.153
Longpre, Shayne, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, et al. 2023. The Flan Collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pages 22631–22648.
Mitchell, Eric. 2023. A note on DPO with noisy preferences & relationship to IPO. https://ericmitchell.ai/cdpo.pdf
Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744. https://doi.org/10.52202/068431-2011
Pishro-Nik, Hossein. 2014. *Introduction to Probability, Statistics, and Random Processes*. Kappa Research, LLC.
Rafailov, Rafael, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.
Stiennon, Nisan, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.
Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMa: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
Wang, Yizhong, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, et al. 2023. How far can camels go? Exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786. https://doi.org/10.52202/075280-3268
Wang, Zhilin, Alexander Bukharin, Olivier Delalleau, Daniel Egert, Gerald Shen, Jiaqi Zeng, Oleksii Kuchaiev, and Yi Dong. 2025. HelpSteer2-Preference: Complementing ratings with preferences. In *The Thirteenth International Conference on Learning Representations*.
Wei, Jason, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.
Wu, Junkang, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jiawei Chen, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. 2024. Towards robust alignment of language models: Distributionally robustifying direct preference optimization. *arXiv preprint arXiv:2407.07880*.
Xu, Can, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. WizardLM: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.
Yang, An, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
Zhou, Jeffrey, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.