HAF-RM: A Hybrid Alignment Framework for Reward Model Training

Anonymous ACL submission

Abstract

The reward model has become increasingly important in alignment, assessment, and data construction for large language models (LLMs). Most existing researchers focus on enhancing reward models through data improvements, following the conventional training framework for reward models that directly optimizes the predicted rewards. In this paper, we propose a hybrid alignment framework HAF-RM for reward model training by introducing an additional constraint on token-level policy probabilities in addition to the reward score. It can simultaneously supervise the internal preference model at the token level and optimize the mapping layer of the reward model at the sequence level. Experiment results on five datasets sufficiently show the validity and effectiveness of our proposed hybrid framework for training a high-quality reward model. By decoupling the reward modeling procedure and incorporating hybrid supervision, our HAF-RM framework offers a principled and effective approach to enhancing the performance and alignment of reward models, a critical component in the responsible development of powerful language models. We release our code at https://haf-rm-anonymized.github.io.

1 Introduction

Recent periods have witnessed a continuous evolution of Large Language Model (LLM) techniques, especially in pre-training (Devlin et al., 2019; Radford et al., 2019; Brown et al., 2020) and instruction tuning (Wei et al., 2021; Wang et al., 2022; Yue et al., 2023). As these models advance, researchers have shifted their focus from generating correct responses to aligning outputs more closely with human preferences (Russell, 2014) through Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022). As an efficient alternative to human feedback, reward models for generative language models emerge, facilitating scalable alignment in training (Christiano et al., 2017; Stiennon et al., 2020), response generation (Gao et al., 2023; Mudgal et al., 2024; Jinnai et al., 2024), and data construction(Yuan et al., 2023) etc.

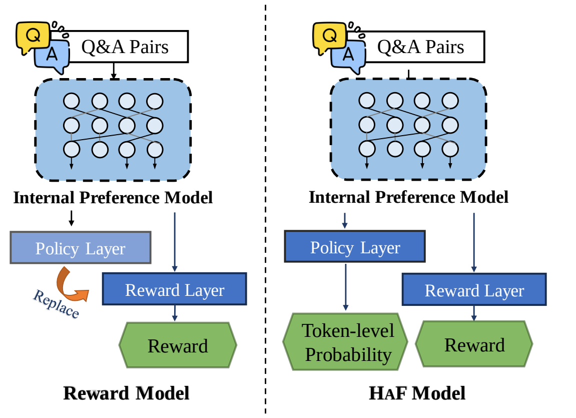

Figure 1: HAF model structure. It retains the policy layer which outputs the token-level probability.

Despite the availability of numerous sophisticated reward models (Kopf et al., 2023; Zhu et al., 2023), several key limitations remain. First, many reward models are proprietary and closed-source, originating from industry, which restricts their further training and transfer. Second, prior studies have highlighted incorrect and ambiguous preferences within the training data of these reward models (Bai et al., 2022; Pitis, 2023). These two issues both limit the quality and generalizability of existing reward models, necessitating further enhancement either from the data perspective or the training process. Recent efforts primarily focus on enriching data sources to improve reward models, including incorporating external tools or information sources to enhance generalization (Li et al., 2023a; Sun et al., 2023) or leveraging fine-grained signals (Wu et al., 2023; Cao et al., 2024) and their combinations (Go et al., 2023; Lai et al., 2024). In contrast, this work aims to improve the training framework of reward models.

A reward model is typically structured with two components: a transformer-based model (referred to as the "internal preference model"), and a projection module called "reward layer" (usually a linear layer). The former outputs preference vectors for each token, while the latter maps these vectors to sequence-level rewards. We argue that the standard practice for training the reward model may cause insufficient supervision for preference modeling, which can be improved by performing hybrid supervision of both token-level and sequence-level.

Given that a policy model also relies on an internal preference model to predict expected rewards for each action or token, essentially acting as a *Q*-function under token-level supervision (Rafailov et al., 2024), we propose a Hybrid Alignment Framework (HAF). This framework jointly optimizes the reward model and the policy model by sharing the internal preference model. With an additional policy loss, we can directly supervise the internal preference model at the token level, while simultaneously optimizing the mapping layer of the reward model using the reward loss, enabling more effective alignment of the reward model.

We provide massive empirical experiments with an intuitional justification to demonstrate the effectiveness of our HAF. In the experiment section, we compare the performance of reward models trained using our framework against those resulting from traditional baseline and DPO approaches across five public datasets. The results highlight the advantage of HAF with different policy losses integrated. Further analysis reveals that using additional policy loss can improve the performance of policy model calibration, which opens a new horizon for training high-quality reward models.

2 Hybrid Alignment Framework

In this section, we first introduce the necessary notations (Section 2.1). Then we derive the formation of reward loss and policy loss as well as their practical calculation methods (Section 2.2), and propose HAF to effectively utilize the similarity between the reward model and the policy model (Section 2.3). Finally, we provide an intuition-based explanation for why HAF works (Section2.4).

2.1 Notation

The objective of our framework is to train the reward model $\boldsymbol{r}$ based on a pairwise comparison dataset (also known as "preference dataset") $\mathcal{D}$, following typical reward model training settings.

- $\mathcal{D} = \{(x_i, y_i, y_i')\}_{i=1}^{n}$ represents the dataset used to train the reward model, where x_i, y_i and y_i' are the query, preferred and non-preferred responses respectively.
- $\mathcal{P} = \{(x, y) \mid (x, y, y') \in \mathcal{D}\} \cup \{(x, y') \mid (x, y, y') \in \mathcal{D}\}$ is the set of query-response pairs from the dataset $\mathcal{D}$.
- $\boldsymbol{r}$ is the **reward model** which can be split into two parts as $\boldsymbol{r}(\mathrm{x}, \mathrm{y}) = \mathrm{F} \circ \phi\,(x, y)$, to output the reward of a response y given a query x. Here, $\phi\,(\cdot, \cdot)$ denotes the model's internal preference model, while F serves as the reward prediction layer mapping the model's internal preference to the final reward. We use the symbol $\circ$ to signify function nesting, i.e., $\mathrm{F} \circ \phi\,(x, y) = \mathrm{F}\,(\phi\,(x, y))$.
- $\boldsymbol{\pi}$ is the **policy model**, and $\boldsymbol{\pi}\,(x, y)$ is the generation probability of y given x. It can also be divided into two parts as $\boldsymbol{\pi}\,(x, y) = \mathrm{K} \circ \phi\,(x, y)$ where the policy prediction layer K maps the model's internal preference to the generation probability.
- The **Oracle value** is denoted as the corresponding letter with an asterisk such as $\boldsymbol{r}^*$(Oracle reward model), ϕ^*(Oracle model preference), F^*(Oracle reward prediction layer) and K^*(Oracle policy prediction layer).

2.2 Basic Loss Functions

We use D_1 to represent the distribution discrepancy between the reward model's output and the oracle reward model's output, and D_2 for the outputs of the policy model and the oracle policy model.

Reward Loss The standard reward loss $\mathcal{L}_s$ considers the precision of rewards alone, being a simple and direct metric to quantify the quality of a reward model.

$$\mathcal{L}_s \coloneqq \mathop{\mathbb{E}}_{d}\left[\mathrm{D}_1\left(\boldsymbol{r}\,(d)\,, \boldsymbol{r}^*\,(d)\right)\right] \tag{1}$$

We use d to denote (x, y) for notational simplicity.

In avoiding the issue of uncertain reward values, there is consensus on the use of the Bradley-Terry model (Bradley and Terry, 1952) to transform the reward modeling problem into a probability optimization problem (Stiennon et al., 2020; Rafailov et al., 2023; Meng et al., 2024), which yields the popular form of a binary classification

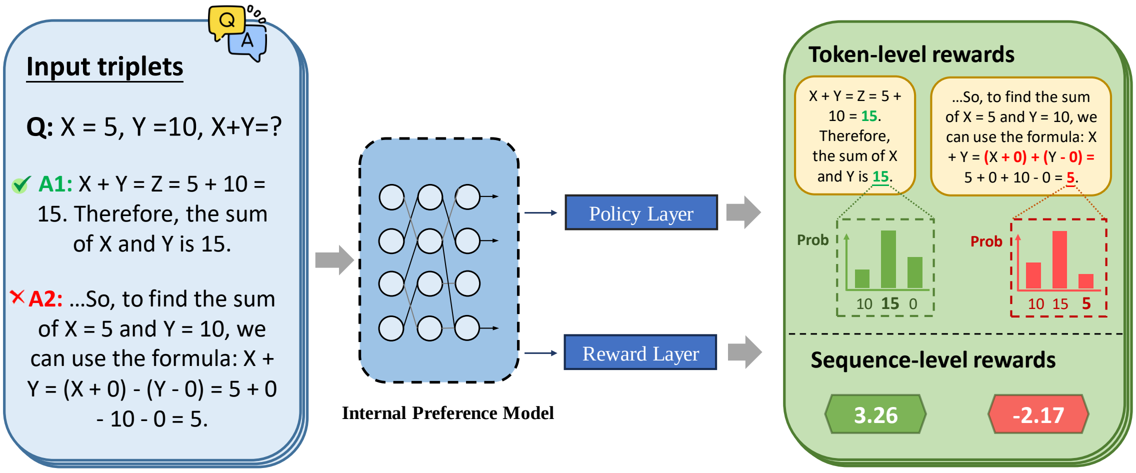

Figure 2: HAF training framework. We add the reward layer to the language model while retaining its policy layer. During training, we optimize both the token-level rewards and sequence-level rewards for the input triplets by maximizing the reward differences between better responses and worse responses.

cross-entropy loss:

$$\mathcal{L}_s \leftarrow \underset{(x,y,y')\sim\mathcal{D}}{\mathbb{E}} \left[-\log\sigma\left(\boldsymbol{r}\left(x,y\right)-\boldsymbol{r}\left(x,y'\right)\right)\right] \quad (2)$$

where $\sigma(\cdot)$ is the sigmoid function (derivation can be found in Appendix C.1).

Policy Loss Similar to the reward loss, the standard policy loss aims to measure the error of the policy model.

$$\mathcal{L}_P \coloneqq \underset{d}{\mathbb{E}}\left[\mathrm{D}_2\left(\boldsymbol{\pi}\left(d\right),\boldsymbol{\pi}^*\left(d\right)\right)\right] \quad (3)$$

Here, we use DPO (Rafailov et al., 2023) for calculating policy loss since its derivation is similar to that made for the reward loss (as detailed in Appendix C.2).

$$\mathcal{L}_P \leftarrow \underset{(x,y,y')\sim\mathcal{D}}{\mathbb{E}}\left[-\log\sigma\left(\tau\left(pd_{win}-pd_{lose}\right)\right)\right] \quad (4)$$

$pd_{win} = \log\frac{\boldsymbol{\pi}(x,y)}{\boldsymbol{\pi}_{ref}(x,y)}$, $pd_{lose} = \log\frac{\boldsymbol{\pi}(x,y')}{\boldsymbol{\pi}_{ref}(x,y')}$. $\boldsymbol{\pi}_{ref}$ is the reference policy model and τ is the hyperparameter set to 0.1.

2.3 HAF Implementation

Hybrid Alignment Loss To fully leverage the similarity between the reward model and the policy model, we incorporate an additional supervising term D_2 on the policy model into the loss function. By calibrating the shared preference space,

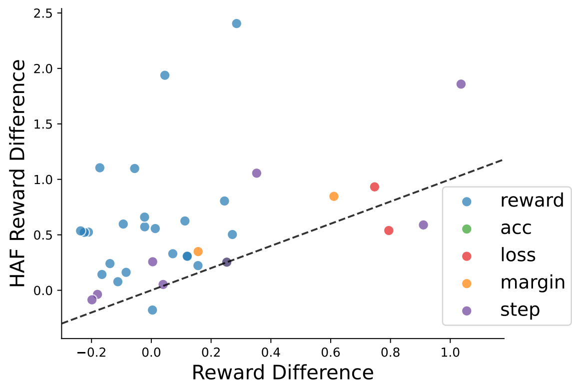

Figure 3: HAF tends to assign higher scores to the responses it generates. The x-axis represents the score difference between the ideal reward model's evaluation of the content generated by HAF's policy head and the content generated by the model trained with DPO. The y-axis indicates the score difference when HAF evaluates these two outputs. Different colors represent different model checkpoint selection strategies.

we effectively align the model in a hybrid manner:

$$\begin{aligned}\mathcal{L}_H &\coloneqq \underset{d}{\mathbb{E}}\left[\mathrm{D}_1\left(\boldsymbol{r}\left(d\right),\boldsymbol{r}^*\left(d\right)\right)\right.\\ &\quad \left.+\alpha\cdot\mathrm{D}_2\left(\boldsymbol{\pi}\left(d\right),\boldsymbol{\pi}^*\left(d\right)\right)\right]\\ &= \underset{d}{\mathbb{E}}\left[\mathrm{D}_1\left(\mathrm{F}\circ\phi\left(d\right),\mathrm{F}^*\circ\phi^*\left(d\right)\right)\right.\\ &\quad \left.+\alpha\cdot\mathrm{D}_2\left(\mathrm{K}\circ\phi\left(d\right),\mathrm{K}^*\circ\phi^*\left(d\right)\right)\right]\end{aligned} \quad (5)$$

where α is a hyperparameter to balance losses from the reward and policy model, ϕ is the shared internal preference model which receives gradients from both loss terms.

Model structure The most commonly used decoder-only LLM consists of stacked transformer blocks (Vaswani et al., 2017) or similar structures, and a linear layer for policy projection. In the reward model, only the shape of the final linear layer is adjusted to match the format of the reward value output compared to the policy model (Stiennon et al., 2020). We retain two linear layers for our model, enabling it to output rewards and probabilities simultaneously, as shown in Figure 1.

2.4 Why HAF is Better?

Figure 3 shows the consistency between the reward model and the policy model in preference learning. Despite possessing similar generation quality, the policy model which shares parameters with the reward model is rated higher, indicating that the two models do have resembling preferences when they have the same internal preference model. We will elaborate on this finding in Appendix E.1.

Besides, we provide an intuitive explanation of why the hybrid alignment loss can yield a better solution than simply using the standard reward loss.

Claim 1. *The model learned from the joint calibrated loss outperforms the one learned solely from the preference space using the standard reward loss.* Details can be found in Appendix D.

Claim 2. *Policy loss can act as a regularization term preventing the inner representation from degrading, so HAF tends to outperform the traditional training framework.*

3 Experimental Setup

3.1 Datasets

We comprehensively evaluate the performance of our framework using five public datasets: Anthropic-HH-Harmless (HH-harmless) (Bai et al., 2022), Anthropic-HH-Helpful (HH-Helpful) (Bai et al., 2022), Beaver Safe (BS) (Ji et al., 2023), Alpaca Human Pref (AHP) (Dubois et al., 2023), and Chatbot Arena (CA) (Zheng et al., 2023). Since AHP and CA do not provide original data split for evaluation, we randomly extract 10% from the original data as the test set. Detailed statistics of our used datasets for training are shown in Table 1.

Dataset	Size	#Word/QA	#Token/QA
Harmless	12,915	42.9	61.5
Helpful	13,543	54.3	77.2
BS	47,625	69.3	88.5
AHP	8,722	59.6	81.9
CA	19,466	165.5	257.6

Table 1: Statistics of the Training Subsets.

3.2 Compared Models

Baseline We compare our framework with the standard training approach, wherein the reward model only has a reward layer dedicated to reward prediction and is optimized only with reward loss, as delineated in Eq. 2.

DPO DPO can implicitly convert model's outputs into reward values (Rafailov et al., 2023), so the model can also function as a reward model (Rafailov et al., 2024). Following the work of Lambert et al. (2024), we evaluate the model trained with DPO loss.

HAF Under our framework, the reward model has both a reward layer and a policy layer for predicting sequence-level rewards and providing token-level probabilities.

Our framework is implemented based on three different backbone LLMs including both pre-trained and fine-tuned models: `Phi-2-2.7B` (Javaheripi et al., 2023), `Mistral-7B-base-v0.3` and `Mistral-7B-Instruct-v0.2` (Jiang et al., 2023). We train Phi-2 and Mistrals using full-parameter and Low-rank Adaptation (LoRA) (Hu et al., 2022) strategies, respectively. More implementation details can be found in Appendix A.

4 Experiment Results

4.1 Intrinsic Performance of Reward Models

The primary function of a reward model is to evaluate the quality of responses to a given question, which involves accurately comparing pairs of answers to the same question. To demonstrate the effectiveness of our HAF in training reward models, we first conduct several experiments evaluating the intrinsic performance of our trained reward model, specifically by taking judgment accuracy as the evaluation metric.

4.1.1 Overall Performance

Table 2 presents the overall results of our HAF compared to two basic approaches across five datasets. We observe that **DPO and the baseline method show similar performance on average but there is significant variability in individual comparisons**. This suggests that the two methods focus on different features when learning preferences. In contrast, HAF consistently outperforms both, in-

Method	Helpful	Harmless	CA	BS	AHP	**Avg**
DPO(Phi-2)	69.70	66.30	66.80	**87.80**	52.60	68.64
Baseline(Phi-2)	64.30	69.50	**79.30**	76.00	58.40	69.50
HAF (Phi-2)	**76.40**	**70.40**	79.00	84.00	**60.80**	**74.12**
DPO(Mistral-base)	64.60	69.90	68.80	**91.70**	53.80	69.76
Baseline(Mistral-base)	72.60	69.80	64.20	78.30	50.40	67.06
HAF (Mistral-base)	**73.00**	**70.00**	**74.40**	85.40	**56.30**	**71.82**
DPO(Mistral)	74.29	70.30	**81.90**	**92.70**	60.30	75.90
Baseline(Mistral)	**76.20**	72.70	79.80	80.80	56.30	73.16
HAF (Mistral)	75.80	**73.10**	**81.90**	88.70	**63.10**	**76.52**

Table 2: Overall results (accuracy) for each dataset, by calculating the proportion that the better response is scored higher. The best performance is highlighted in boldface and the suboptimal result is underlined.

dicating its ability to effectively integrate features from both approaches to better learn preferences.

Specifically, Mistral-base performs poorly on the Helpful, CA, and AHP datasets because these datasets require preferences related to the quality of responses. **Since the base model has not undergone instruction tuning, it lacks the representation of relevant features, making it difficult to accurately judge response quality**. In contrast, the extensively trained base model is capable of distinguishing between benign and harmful content, allowing it to perform comparably to Mistral-Instruct on the safety-related BS and Harmless datasets. Nevertheless, HAF demonstrates promising results even for these challenging preferences.

Notably, DPO achieves the highest performance on BS across all three models, which is probably caused by DPO's "concentrated" data-fitting manner (Azar et al., 2023). This is evident from the much lower variance in token-level perplexity for good and bad responses in the BS dataset compared to other datasets, indicating a more concentrated distribution respectively of these two subsets (refer to Appendix E.2 for detailed illustration). By integrating DPO loss, our HAF partially captures this "concentrated" data-fitting characteristics, leading to a more nuanced improvement on BS compared to the baseline methods. However, DPO's concentrated data-fitting may potential lead to over-fitting issues, whereas HAF and the baseline demonstrate better generalization ability, which we will elaborate on in the following experiments.

4.1.2 Evaluation on Mixed Data

To illustrate HAF 's effectiveness in training reward models on mixed data, we construct a dataset by evenly sampling and combining examples from all five datasets. As shown in Figure 4, our proposed hybrid alignment framework achieves the best overall performance across all reward models when evaluated on the mixed data distribution. This suggests that HAF is more effective at learning the diversity within the combined datasets.

Specifically, compared to the individual results on corresponding datasets in Table 2 (shown as lightly shaded bars in Figure 4), we observe that **both the baseline method and HAF replicate their performance in learning individual preferences better than DPO when applied to mixed preference learning**. Notably, DPO's performance drops significantly on the CA and Helpful datasets, suggesting that DPO tends to fit the most prominent features of the overall data distribution. This also aligns with the finding of Chen et al. (2024) that DPO would optimize the margins of correct data rather than the wrong ones.

4.1.3 Transferability to OOD Data

We further evaluate the generalizability of our framework to entirely held-out out-of-distribution (OOD) datasets to simulate distribution shifts in real-world applications. Specifically, the five datasets are grouped into two categories: "Safety" (BS, Harmless) and "Chat" (AHP, CA, Helpful). We train the model on one dataset and evaluate its performance within the same category. The evaluation data comes from two sources, including the "*internal*" source referring to different datasets within the same category, and an "*external*" source, consisting of test data on related topics from RewardBench.

As shown in Table 3, HAF achieves a higher *internal* accuracy compared to both Baseline and DPO, demonstrating HAF 's strong ability to learn preferences and effectively generalize to similar preference distributions, even with notable differ-

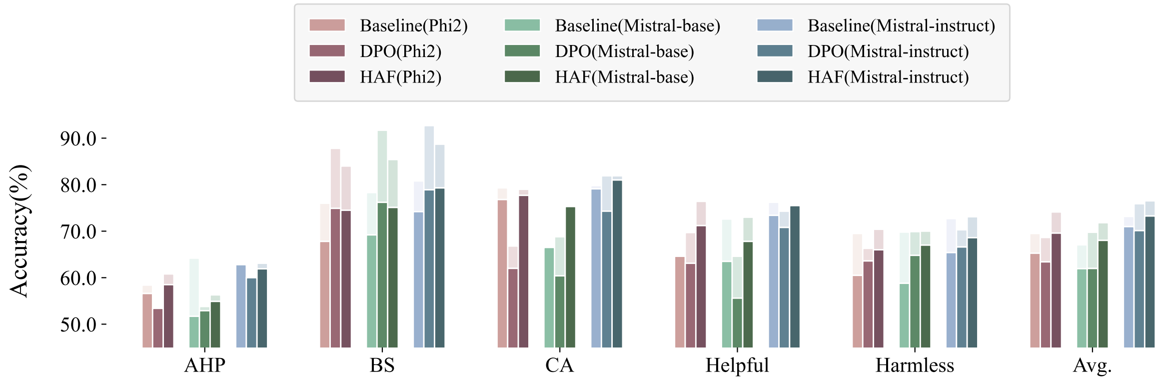

Figure 4: The performance differences of HAF / baseline / DPO under mixed preference training, with light shading indicating the upper bound performance of individually trained reward models on each dataset.

Acc(%)	AHP_C	CA_C	Helpful_C	BS_S	Harmless_S	Avg.
			internal			
Phi-2	$67.50^{(1.20\uparrow)}_{(23.70\uparrow)}$	$62.45^{(1.35\downarrow)}_{(11.60\uparrow)}$	$66.10^{(0.90\uparrow)}_{(19.80\uparrow)}$	$70.60^{(5.60\uparrow)}_{(4.60\uparrow)}$	$76.90^{(1.50\uparrow)}_{(8.60\uparrow)}$	$68.71^{(1.57\uparrow)}_{(13.66\uparrow)}$
Mistral-base	$59.65^{(4.90\uparrow)}_{(14.55\uparrow)}$	$56.35^{(2.15\downarrow)}_{(6.00\uparrow)}$	$62.40^{(0.85\downarrow)}_{(12.85\uparrow)}$	$69.60^{(0.50\uparrow)}_{(3.30\uparrow)}$	$75.30^{(1.90\uparrow)}_{(5.80\uparrow)}$	$64.66^{(0.86\uparrow)}_{(8.50\uparrow)}$
Mistral	$72.20^{(8.40\uparrow)}_{(12.75\uparrow)}$	$63.30^{(0.70\downarrow)}_{(9.65\uparrow)}$	$67.40^{(0.20\downarrow)}_{(14.25\uparrow)}$	$71.90^{(1.40\uparrow)}_{(3.00\uparrow)}$	$76.70^{(2.40\uparrow)}_{(5.70\uparrow)}$	$70.30^{(2.26\uparrow)}_{(9.07\uparrow)}$
			external			
Phi-2	$85.14^{(1.36\uparrow)}_{(65.88\uparrow)}$	$95.27^{(0.34\uparrow)}_{(19.59\uparrow)}$	$89.86^{(6.08\uparrow)}_{(74.66\uparrow)}$	$66.30^{(0.95\uparrow)}_{(2.04\uparrow)}$	$66.44^{(0.38\downarrow)}_{(4.62\uparrow)}$	$80.60^{(8.35\uparrow)}_{(33.36\uparrow)}$
Mistral-base	$79.66^{(20.34\uparrow)}_{(64.14\uparrow)}$	$93.79^{(21.03\uparrow)}_{(33.45\uparrow)}$	$81.38^{(6.90\downarrow)}_{(67.24\uparrow)}$	$70.40^{(3.27\uparrow)}_{(8.73\uparrow)}$	$63.30^{(3.82\downarrow)}_{(4.91\uparrow)}$	$77.70^{(6.79\uparrow)}_{(35.69\uparrow)}$
Mistral	$91.55^{(32.77\uparrow)}_{(53.37\uparrow)}$	$91.89^{(3.04\uparrow)}_{(16.21\uparrow)}$	$82.43^{(1.69\uparrow)}_{(63.51\uparrow)}$	$70.52^{(1.22\downarrow)}_{(4.08\uparrow)}$	$73.37^{(2.72\uparrow)}_{(5.17\uparrow)}$	$81.95^{(7.80\uparrow)}_{(28.47\uparrow)}$

Table 3: Results for out-of-distribution data. Subscripts C and S denote the subjects of training sets, where C represents Chat and S represents Safety. "*internal*" refers to testing results among datasets sharing the same subject category, while "*external*" refers to testing results on RewardBench. The displayed accuracies are for HAF , with superscripts and subscripts indicating the performance differences relative to the baseline and DPO, respectively. $\uparrow$ denotes an improvement with HAF , while $\downarrow$ signifies a decline.

ences in language style and topic. As Touvron et al. (2023) noted, RLHF causes distributional shifts in the policy model during training, often requiring iterative training of the reward model. HAF 's robustness against these distributional shifts could potentially be a key factor in mitigating this issue.

It is important to note that nearly all of DPO's test outcomes converge around 50%, indicating a complete loss of modeling capability for OOD data. This likely stems from DPO's inherent nature as a language model, where the generation process exhibits strong stylistic biases, favoring responses that align with its style (as reflected in generation probabilities and implicit reward values). When response distribution deviates from these stylistic norms (e.g., responses that are too short, too long, or use different vocabulary), DPO's output probabilities become highly inaccurate, rendering it unsuitable as a conventional reward model.

From these three experiments, we conclude that DPO learns features significantly different from those learned by the baseline method. In contrast, HAF inherits both the baseline method's generalization ability and DPO's stronger fitting capability.

4.2 Extrinsic Evaluation on Downstream Task

Intrinsic performance metrics offer only a partial view of a reward model's efficacy. To comprehen-

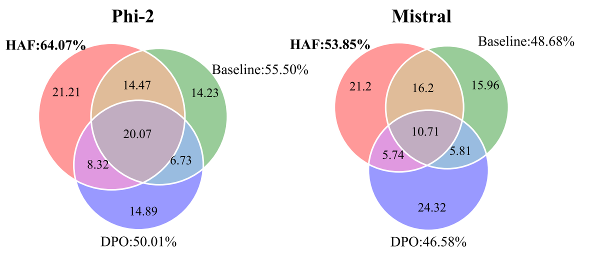

Figure 5: Average win rates of responses selected by the HAF reward model, baseline model and the DPO reward model. Circles may overlap as different models select the same response.

		Phi-2		Mistral	
		All	Chat	All	Chat
Top-1	*Random*	*25.00*	*25.00*	*12.50*	*12.50*
	Baseline	27.43	28.97	16.03	18.27
	DPO	22.94	26.39	12.81	13.85
	HAF	**33.77**	**37.19**	**18.19**	**21.12**
Top-2	*Random*	*50.00*	*50.00*	*25.00*	*25.00*
	Baseline	49.71	53.39	30.64	35.13
	DPO	46.22	51.59	29.05	31.56
	HAF	**58.28**	**64.23**	**34.89**	**39.96**

Table 4: Top-k recalls of different reward models. *Random* shows the recall when choosing responses randomly. The results are averaged over the recall values from all datasets. “Chat” indicates that the result in that column is averaged over the AHP, CA, and Harmless instead of all five datasets.

sively assess their practical applicability in real-world scenarios, it is crucial to evaluate how these models perform in downstream tasks that closely simulate practical applications.

In this section, we evaluate the robustness and effectiveness of HAF in such scenarios. Specifically, we explore its performance in two distinct downstream tasks: best-of-N sampling, a training-free response generation strategy (Stiennon et al., 2020; Gao et al., 2023; Jinnai et al., 2024), and RLHF, a training-dependent alignment method.

4.2.1 Best-of-N

We demonstrate the reliability of our trained reward model through Best-of-N selection, where the reward model should pick the best response (the one with the highest reward) from several responses sampled from the same generative model. The backbone for the reward model and the generation model is the same, with 8 and 4 responses are provided to the Mistral-Instruct reward model and the Phi-2 reward model, respectively. Because Phi-2 tends to generate more similar responses, reducing the need for 8 candidates. The prompts used for comparisons and ranking are listed in Appendix F, referencing AlpacaEval (Li et al., 2023b). We report two evaluation metrics. **Win rate**: We use GPT-4-turbo to rank the responses from HAF , DPO, and baseline reward model and report the win rate (Jang et al., 2023). **Consistency**: we use GPT-4-turbo to rank the sampled responses and calculate the recall of the top-1 and top-2 responses.

As shown in Figure 5 and Table 4, HAF demonstrates significant advantages over the baseline and DPO reward models in selecting responses in terms of both evaluation metrics, especially taking Phi-2 as the backbone. Notably, the recall scores of both DPO and baseline are close to those of random selection, indicating poor sensitivity and an inability to discern between responses with minimal quality differences. In contrast, the reward model trained by HAF exhibits good discriminative ability.

Considering that the model primarily learn to distinguish between harmful and non-harmful responses from the BS and Harmless datasets, and the responses generated by Phi-2 and Mistral are mostly benign, we also report average results on the remaining three datasets. When the safety-related datasets are excluded, all models show an improvement in average performance. The detailed results as well as the ArmoRM-judged results can be found in the appendix in Table 11, Figure 10.

Figure 5 presents the win rates of each method. We can observe that HAF consistently has the highest probability of selecting the best response (among the three methods), while DPO performs the worst. The frequency with which the baseline reward model and the HAF reward model select the same optimal response is considerably higher than their agreement with DPO. This difference is partly due to their modeling approaches: both HAF and the baseline reward model directly produce numerical rewards, whereas DPO derives rewards from token probabilities.

4.2.2 RLHF

We also test HAF in the standard RLHF process: we train two reward models respectively with HAF and the baseline method and then use them to train policy models through RLHF. After training, GPT-4 acts as the evaluator to compare the generations from the two policy models. We conduct two sets of experiments: one for training a Safety reward

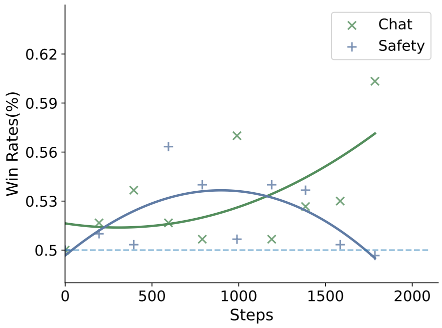

Figure 6: Win rates for the policy model trained with the HAF reward model using RLHF compared to the baseline reward model, with each comparison made at the same training steps.

model using the BS and Harmless datasets; and the other for training a Chat reward model using the AHP, CA, and Helpful datasets. We compare the response quality of the policy models optimized after the same number of PPO steps by the baseline reward model and the HAF reward model.

As shown in Figure 6, the improvement of HAF is particularly evident on the Chat dataset, with its win rate increasing throughout the training, highlighting the superiority of the HAF reward model. In contrast, during safety training, the HAF reward model only shows a significant advantage over the baseline model primarily in the middle stages of training. This is likely because both models have largely achieved harmless responses on the test set, resulting in minimal differentiation between the two reward models.

5 Related Work

Reward model was proposed to modeling human language preferences (model that outputs preference values based on questions and answers) (Christiano et al., 2017), then the explosive growth of research on reward models (McKinney et al., 2023) and large language models (Wei et al., 2022; Park et al., 2023; Zheng et al., 2023) emerged after the popularity of ChatGPT.

From training to practical applications, an increasing number of studies have also featured the presence of quantifiable preferences(usually known as "reward"). For example, RLHF (Christiano et al., 2017; Stiennon et al., 2020) uses the PPO algorithm (Schulman et al., 2017) to maximize the reward of the policy model; RAFT (Dong et al., 2023) and RRHF (Yuan et al., 2023) remove substandard data by scoring the candidate responses with reward model; LLM-as-a-judge (Zheng et al., 2023) employs GPT-4 to score the text.

Therefore, how to construct a model offering explicit preference feedback has naturally become a focal point of much research. To train a precise and robust reward model, many studies start from training with human preference data, and many works in the data field are largely centered around this. Touvron et al. (2023) and Zhao et al. (2022) provided different methods for using ranking data; Wang et al. (2024a) explored ways of measuring the strength of the data; while concerning datasets themselves, Azar et al. (2023), Knox et al. (2022) and Hong et al. (2022) analyzed the impact of data preference strength on training from theoretical or practical perspectives. In addition, similar to the RAG technique (Lewis et al., 2020) in large language models, many methods (Li et al., 2023a; Sun et al., 2023) using external tools or references have also emerged, injecting new vitality into the development of reward models.

Although many data-oriented methods have greatly enhanced the performance of reward models, the field of reward model optimization has been rarely explored. Currently, the training of reward models basically follows the process proposed by OpenAI (Christiano et al., 2017). Considering the widespread practical applications of reward models, the attention given to their training paradigms does not match their importance.

6 Conclusion

In this paper, we extend and improve the training framework of the current reward model. We split the training mechanism of the reward model into two stages: aligning model preference and optimizing the reward layer. Through introducing an additional constraint of policy loss, our hybrid alignment framework supervises the internal preference model at the token level while simultaneously optimizing the mapping layer at the sequence level, significantly improving the training effectiveness. We theoretically verify the validity of our method and demonstrate its reliability through systematic experiments.

Our method allows for a consistent customization of the reward model. In the future, we will thoroughly explore the potential of the reward model and its variants across various tasks, and investigate whether the logistic distribution is the optimal prior for reward modeling.

Impact Statements

This paper presents work whose goal may benefit the training of large language models in the field of deep learning. Among the many possible consequences, we do not believe that there is a significant possibility of adverse effects on society.

Limitations

In this paper, we discuss the potential of enhancing the alignment process of reward models by incorporating policy constraints, where the policy loss functions similarly to a regularization loss, acting as an auxiliary function to guide model training. However, since DPO can be directly used to train an implicit reward model, replacing the reward model with a DPO model for downstream tasks can also be a feasible approach, while we do not explore methods for combining the outputs of the policy layer and the reward layer, which remains a direction for our future research.

References

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences. *ArXiv*, abs/2310.12036.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, T. J. Henighan, Nicholas Joseph, Saurav Kadavath, John Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Christopher Olah, Benjamin Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *ArXiv*, abs/2204.05862.

Ralph Allan Bradley and Milton E. Terry. 1952. RANK ANALYSIS OF INCOMPLETE BLOCK DESIGNS: THE METHOD OF PAIRED COMPARISONS. *Biometrika*, 39(3-4):324–345.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

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

Angelica Chen, Sadhika Malladi, Lily H Zhang, Xinyi Chen, Qiuyi Zhang, Rajesh Ranganath, and Kyunghyun Cho. 2024. Preference learning algorithms do not learn preference rankings. *arXiv preprint arXiv:2405.19534*.

Paul Francis Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *ArXiv*, abs/1706.03741.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and T. Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *ArXiv*, abs/2304.06767.

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

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 10835–10866. PMLR.

Dongyoung Go, Tomasz Korbak, Germán Kruszewski, Jos Rozen, and Marc Dymetman. 2023. Compositional preference models for aligning lms. *Preprint*, arXiv:2310.13011.

Joey Hong, Kush Bhatia, and Anca D. Dragan. 2022. On the sensitivity of reward inference to misspecified human models. *ArXiv*, abs/2212.04717.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. 2023. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *Preprint*, arXiv:2310.11564.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, Suriya Gunasekar, Piero Kauffmann, Yin Tat Lee, Yuanzhi Li, Anh Nguyen, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Michael Santacroce, Harkirat Singh Behl, Adam Taumann Kalai, Xin Wang, Rachel Ward, Philipp Witte, Cyril Zhang, and Yi Zhang. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2023. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *ArXiv*, abs/2307.04657.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Yuu Jinnai, Tetsuro Morimura, Kaito Ariu, and Kenshi Abe. 2024. Regularized best-of-n sampling to mitigate reward hacking for language model alignment. *Preprint*, arXiv:2404.01054.

W. B. Knox, Stephane Hatgis-Kessell, Serena Booth, Scott Niekum, Peter Stone, and Alessandro Allievi. 2022. Models of human preference for learning reward functions. *ArXiv*, abs/2206.02231.

Andreas Kopf, Yannic Kilcher, Dimitri von Rutte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Rich'ard Nagyfi, ES Shahul, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. 2023. Openassistant conversations - democratizing large language model alignment. *ArXiv*, abs/2304.07327.

Yuhang Lai, Siyuan Wang, Shujun Liu, Xuanjing Huang, and Zhongyu Wei. 2024. Alarm: Align language models via hierarchical rewards modeling. *Preprint*, arXiv:2403.06754.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Rewardbench: Evaluating reward models for language modeling. *Preprint*, arXiv:2403.13787.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401.

Lei Li, Yekun Chai, Shuohuan Wang, Yu Sun, Hao Tian, Ningyu Zhang, and Hua Wu. 2023a. Tool-augmented reward modeling. *ArXiv*, abs/2310.01045.

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

Lev McKinney, Yawen Duan, David Krueger, and Adam Gleave. 2023. On the fragility of learned reward functions. *ArXiv*, abs/2301.03652.

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

Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, Jilin Chen, Alex Beutel, and Ahmad Beirami. 2024. Controlled decoding from language models. *Preprint*, arXiv:2310.17022.

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

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*.

Silviu Pitis. 2023. Failure modes of learning reward models for LLMs and other sequence models. In *ICML 2023 Workshop The Many Facets of Preference-Based Learning*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *ArXiv*, abs/2305.18290.

Stuart Russell. 2014. Of myths and moonshine. online.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

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

Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liangyan Gui, Yu-Xiong Wang, Yiming Yang, Kurt Keutzer, and Trevor Darrell. 2023. Aligning large multimodal models with factually augmented rlhf. *ArXiv*, abs/2309.14525.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Neural Information Processing Systems*.

Bing Wang, Rui Zheng, Luyao Chen, Yan Liu, Shihan Dou, Caishuang Huang, Wei Shen, Senjie Jin, Enyu Zhou, Chenyu Shi, Songyang Gao, Nuo Xu, Yuhao Zhou, Xiaoran Fan, Zhiheng Xi, Jun Zhao, Xiao Wang, Tao Ji, Hang Yan, Lixing Shen, Zhan Chen, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yuanyuan Jiang. 2024a. Secrets of rlhf in large language models part ii: Reward modeling. *ArXiv*, abs/2401.06080.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024b. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *EMNLP*.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023. Large language models are not fair evaluators. *Preprint*, arXiv:2305.17926.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. In *Annual Meeting of the Association for Computational Linguistics*.

Jason Wei, Maarten Bosma, Vincent 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*, abs/2109.01652.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-grained human feedback gives better rewards for language model training. In *Advances in Neural Information Processing Systems*, volume 36, pages 59008–59033. Curran Associates, Inc.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Feiran Huang. 2023. Rrhf: Rank responses to align language models with human feedback without tears. *ArXiv*, abs/2304.05302.

Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Wei Lin, et al. 2023. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *arXiv preprint arXiv:2309.11325*.

Yao Zhao, Misha Khalman, Rishabh Joshi, Shashi Narayan, Mohammad Saleh, and Peter J. Liu. 2022. Calibrating sequence likelihood improves conditional language generation. *ArXiv*, abs/2210.00045.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Haotong Zhang, Joseph Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *ArXiv*, abs/2306.05685.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

A Experiments Setup

Our default setup is shown in Table 5.

To train the reward model, we use DPO Loss as the policy loss in HAF and set policy ratio $\alpha = 0.2$. The learning rate is 1.0×10^{-5} for Phi-2 and Mistral-Instruct with the baseline method, 3.0×10^{-5} for Mistral-Base and Mistral-Instruct using other methods, and the batch size is set at 16. These configurations are the optimal combination of learning rates (among 1.0×10^{-4}, 3.0×10^{-5}, 1.0×10^{-5}, 3.0×10^{-6}) and batch sizes (among 4,

setup	value	setup	value	setup	value
lora rank	64	**optimizer**	AdamW	**precision**	fp16
lora alpha	16	**adam_beta1**	0.9	**max gradient norm**	1.0
training steps	3200	**adam_beta2**	0.999	**max sequence length**	512
evaluation steps	0.025	**weight_decay**	0.0	**global random seed**	0
batch size	16	**adam_epsilon**	1e-5	**framework**	PyTorch

Table 5: Default setup

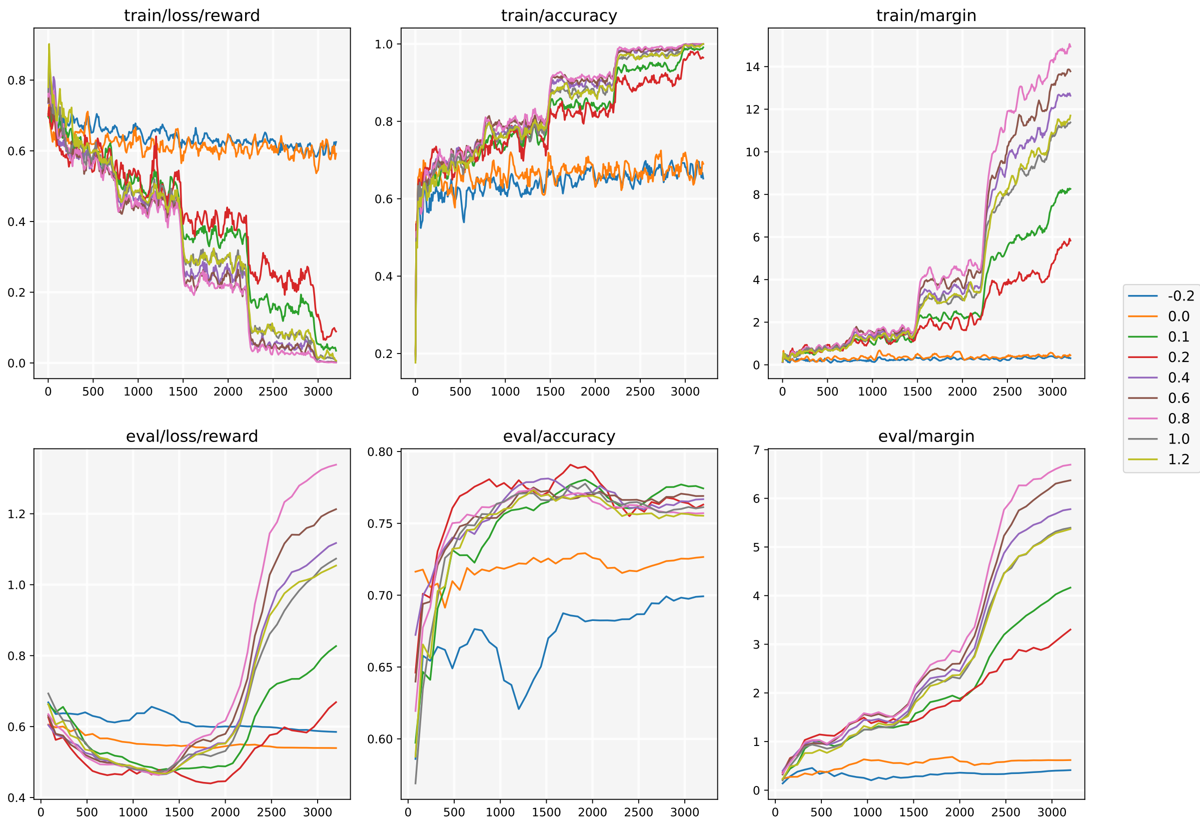

Figure 7: Results for different policy ratios. "margin" is the average difference between a better and worse response's rewards. A policy ratio of 0 equals to Baseline method.

16, 128). A single RTX A6000 with 48GB memory is used for training the reward model. The model used for testing is the checkpoint that achieves the highest reward on the validation set.

For PPO training in Section 4.2.2, we set the total batch size at 16. The maximum number of new tokens generated is set to 256, and the learning rate is 2.0×10^{-5}. The training is conducted over a maximum of 100,000 episodes. All other settings follow the implementation in the LLaMA-Factory library. The generation config includes top_p=0.9, do_sample=True.

B Discussions for Policy Loss Ratio

Figure 7 reveals that incorporating even a mere 0.1x of policy loss can significantly impact the results. Using reward loss alone leads to slow training; to achieve the same loss value, the model with policy loss requires only a fraction of the time. However, this rapid training characteristic also accelerates overfitting, necessitating the use of early stopping strategies to halt training in time. When the policy loss ratio is negative, model performance deteriorates, and the variations in various metrics resemble those of the baseline. This indicates a correlation between the policy model and the reward model.

C Loss Functions

C.1 Deriving the Reward Loss Functions

In the Bradley-Terry model's assumption, Oracle reward model outputs rewards in connection with the win rates:

$$\mathop{\mathbb{E}}_{p\sim J}\mathbb{I}(y > y'; x, p) = -\log\sigma[\boldsymbol{r}^*(x,y) - \boldsymbol{r}^*(x,y')] \tag{6}$$

where p is a judge (annotator) sampled from the judge distribution J.

As we only focus on the reward differences between responses to the same prompt, there exists another metric denoted as D_1' for calculating the reward loss:

$$\begin{aligned}\mathcal{L}_s = \mathop{\mathbb{E}}_{x,y,y'}\mathrm{D}_1'[&r(x,y) - r(x,y'),\\ &r^*(x,y) - r^*(x,y')]\end{aligned}$$

As $-\log\sigma(\cdot)$ is monotonically increasing, so there exists a metric D_1'', such that

$$\begin{aligned}&\mathrm{D}_1'[r(x,y) - r(x,y'), r^*(x,y) - r^*(x,y')]\\ =&\mathrm{D}_1''[-\log\sigma(\boldsymbol{r}(x,y) - \boldsymbol{r}(x,y')),\\ &\qquad -\log\sigma(\boldsymbol{r}^*(x,y) - \boldsymbol{r}^*(x,y'))]\\ =&\mathrm{D}_1''[-\log\sigma(\boldsymbol{r}(x,y) - \boldsymbol{r}(x,y')),\\ &\qquad \mathop{\mathbb{E}}_{p\sim J}\mathbb{I}(y > y'; x, p)]\end{aligned}$$

Let D_1'' be the cross-entropy loss, and let $\mathrm{P}(x,y,y') = -\log\sigma(\boldsymbol{r}(x,y) - \boldsymbol{r}(x,y'))$,

$$\begin{aligned}\mathcal{L}_s =& \mathop{\mathbb{E}}_{x,y,y'}[\mathrm{P}(x,y,y')\cdot\mathop{\mathbb{E}}_{p\sim J}\mathbb{I}(y > y'; x, p)\\ &+ (1-\mathrm{P}(x,y,y'))\cdot(1-\mathop{\mathbb{E}}_{p\sim J}\mathbb{I}(y > y'; x, p))]\\ =& \mathop{\mathbb{E}}_{\substack{x,y,y'\\ p\sim J}}[\mathrm{P}(x,y,y')\cdot\mathbb{I}(y > y'; x, p)\\ &+ (1-\mathrm{P}(x,y,y'))\cdot(1-\mathbb{I}(y > y'; x, p))]\end{aligned}$$

which is exactly Eq. 2 when we sample from $\mathcal{D}$.

C.2 DPO as the Policy Loss

The derivation for policy loss is the same as reward loss in their essence. The policy model can be treated as a reward model with sequence probabilities reflecting the rewards (Rafailov et al., 2023, 2024). $reward(x,y) = \log[\pi(x,y)/\pi_{ref}(x,y)]$.

From this perspective, the DPO loss and reward loss share the same assumption (Eq. 6). The reward model and the DPO-trained policy model are essentially doing the same task despite some formal differences (Rafailov et al., 2023, 2024).

D Mathematical Enlightenment

D.1 Theoretical Explanation for the Claims

Inequality for claim 1. Unless K can exactly fit K^*, there exists $\epsilon > 0$, such that

$$\begin{aligned}&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_2(\mathrm{K}_\mathrm{H}\circ\phi_\mathrm{H}(d), \mathrm{K}^*\circ\phi^*(d))]\\ \leqslant& \min_{\mathrm{K}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_2(\mathrm{K}\circ\phi_\mathrm{s}(d), \mathrm{K}^*\circ\phi^*(d))] - \frac{\epsilon}{\alpha}\end{aligned}$$

holds for all $\alpha \in (0.1, 2)$, where $K_H, \phi_H = \mathop{\mathrm{argmin}}_{\mathrm{K},\phi}\mathcal{L}_H$ in Equation 5 and $\phi_s = \mathop{\mathrm{argmin}}_{\phi}\mathcal{L}_s$ in Equation 2. Here we use argmin to represent the best models optimized with the corresponding loss functions, so ϕ_H and ϕ_s are not equal to ϕ^* although ϕ^* is the minimum mathematically.

Inequality for claim 2. Assume that ϕ^* is unique, K^* is locally Lipschitz continuous, , and $0.1 < \alpha < 2$, there exists $k, \delta > 0$, such that

$$\begin{aligned}&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[|\phi_H(d) - \phi^*(d)| - |\phi_s(d) - \phi^*(d)|] <\\ &\frac{g_{\max} - g_{\min}}{g_{\min}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\phi_s(d) - \phi^*(d)| + 2\delta - \frac{\epsilon}{\alpha\cdot k}\end{aligned}$$

We obtain informally here an upper bound on the model preference error. By tuning the hyperparameter α, the right term can be strictly negative.

D.2 Inequality Scaling

$$\begin{aligned}&\min_{\mathrm{F},\phi,\mathrm{K}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}\circ\phi(d), \mathrm{F}^*\circ\phi^*(d))\\ &\qquad +\alpha\cdot\mathrm{D}_2(\mathrm{K}\circ\phi(d), \mathrm{K}^*\circ\phi^*(d))]\\ \leqslant& \min_{\substack{\mathrm{F}=\mathrm{F}_\mathrm{s}\\ \phi=\phi_\mathrm{s}\\ \mathrm{K}}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}\circ\phi(d), \mathrm{F}^*\circ\phi^*(d))\\ &\qquad +\alpha\cdot\mathcal{L}_2(\mathrm{K}\circ\phi(d), \mathrm{K}^*\circ\phi^*(d))]\\ =& \min_{\mathrm{K}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\alpha\cdot\mathrm{D}_2(\mathrm{K}\circ\phi_\mathrm{s}(d), \mathrm{K}^*\circ\phi^*(d))]\\ &\quad +\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}_\mathrm{s}\circ\phi_\mathrm{s}(d), \mathrm{F}^*\circ\phi^*(d))]\end{aligned}$$

With the definition of $\phi_H, \mathrm{K}_H, \mathrm{F}_H$, we have:

$$\begin{aligned}&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}_\mathrm{H}\circ\phi_\mathrm{H}(d), \mathrm{F}^*\circ\phi^*(d))\\ &\qquad +\alpha\cdot\mathrm{D}_2(\mathrm{K}_\mathrm{H}\circ\phi_\mathrm{H}(d), \mathrm{K}^*\circ\phi^*(d))]\\ \leqslant&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}_\mathrm{s}\circ\phi_\mathrm{s}(d), \mathrm{F}^*\circ\phi^*(d))]\\ &\qquad +\min_{\mathrm{K}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\alpha\cdot\mathrm{D}_2(\mathrm{K}\circ\phi_\mathrm{s}(d), \mathrm{K}^*\circ\phi^*(d))]\\ \leqslant&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}_\mathrm{H}\circ\phi_\mathrm{H}(d), \mathrm{F}^*\circ\phi^*(d))]\\ &\qquad +\min_{\mathrm{K}}\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\alpha\cdot\mathrm{D}_2(\mathrm{K}\circ\phi_\mathrm{s}(d), \mathrm{K}^*\circ\phi^*(d))]\end{aligned}$$

In practical settings, "$\leqslant$"s do not hold at the same time (simultaneously optimizing two objectives is preferable to optimizing them sequentially). With the premise that the model is fully optimized with the hybrid alignment loss for any $\alpha \in (0.1, 2)$, which means both of the objectives have an impact on the final optimization result, namely $\phi_H \neq \phi_s$, there exists a little gap $\epsilon > 0$ such that

$$\begin{aligned}
&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}_\mathrm{H} \circ \phi_\mathrm{H}(d), \mathrm{F}^* \circ \phi^*(d)) \\
&\qquad + \alpha \cdot \mathrm{D}_2(\mathrm{K}_\mathrm{H} \circ \phi_\mathrm{H}(d), \mathrm{K}^* \circ \phi^*(d))] \\
&\leqslant \mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_1(\mathrm{F}_\mathrm{H} \circ \phi_\mathrm{H}(d), \mathrm{F}^* \circ \phi^*(d))] \\
&\quad + \min_{\mathrm{K}} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\alpha \cdot \mathrm{D}_2(\mathrm{K} \circ \phi_\mathrm{s}(d), \mathrm{K}^* \circ \phi^*(d))] - \epsilon
\end{aligned}$$

Then, there goes

$$\begin{aligned}
&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_2(\mathrm{K}_\mathrm{H} \circ \phi_\mathrm{H}(d), \mathrm{K}^* \circ \phi^*(d))] \\
&\leqslant \min_{\mathrm{K}} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_2(\mathrm{K} \circ \phi_\mathrm{s}(d), \mathrm{K}^* \circ \phi^*(d))] - \frac{\epsilon}{\alpha}
\end{aligned}$$

Here we get the first inequality.

D.3 Derive the Final Inequality with the 3 Properties

Convergence:

Since the trained model $\mathrm{K} \circ \phi$ is close to $\mathrm{K}^* \circ \phi^*$, we can therefore linearize D_2 with a certain positive number k:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{d\sim\mathcal{P}}[\mathrm{D}_2(\mathrm{K} \circ \phi(d), \mathrm{K}^* \circ \phi^*(d))] \\
&= \mathop{\mathbb{E}}_{d\sim\mathcal{P}} k|\mathrm{K} \circ \phi(d) - \mathrm{K}^* \circ \phi^*(d)|
\end{aligned} \tag{7}$$

Separating little disturbance:

$$\mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{N} \circ \phi(d)| < \delta \tag{8}$$

holds for any fully-optimized model $\mathrm{K} \circ \phi$ with $\mathrm{N} := \mathrm{K} - \mathrm{K}^*$. Given that the trained model and its preferences closely approximate those of the true model and preferences, we are able to scale down the error terms by a small margin.

Gradient scaling:

Intuitively, the optimal model is unique, so $\mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{K}^* \circ \phi(d) - \mathrm{K}^* \circ \phi^*(d)| > 0$. Here we make a slightly stronger assumption that K^* is locally g_{max}-Lipschitz continuous and has the lower bound g_{min}, which means for any ϕ that is close to ϕ^*, there exists

$$\begin{aligned}
&g_{min} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}||\phi(d) - \phi^*(d)|| \\
&< \mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{K}^* \circ \phi(d) - \mathrm{K}^* \circ \phi^*(d)| \\
&< g_{max} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}||\phi(d) - \phi^*(d)||
\end{aligned} \tag{9}$$

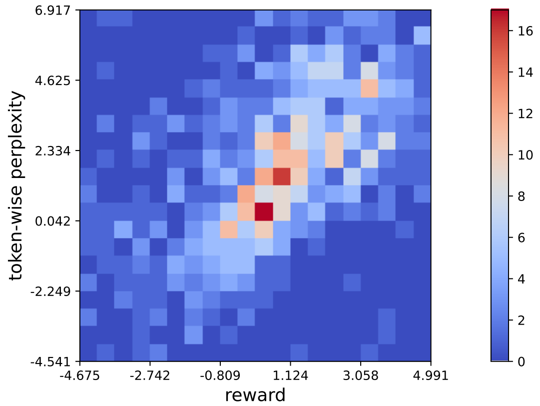

Figure 8: The distribution of the reward model's and DPO model's outputs on test data when trained with identical data.

Based on these three properties, we can derive the result from Appendix D.1.

Inequality 2

$$\begin{aligned}
&\overset{\text{Eq. 7}}{\Longrightarrow} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{K}_\mathrm{H} \circ \phi_H(d) - \mathrm{K}^* \circ \phi^*(d)| \\
&\quad \leqslant \min_{\mathrm{K}} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{K} \circ \phi_s(d) - \mathrm{K}^* \circ \phi^*(d)| - \frac{\epsilon}{\alpha \cdot k} \\
&\overset{\text{Ineq. 8}}{\Longrightarrow} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{K}^* \circ \phi_H(d) - \mathrm{K}^* \circ \phi^*(d)| - \delta \\
&\quad < \mathop{\mathbb{E}}_{d\sim\mathcal{P}}|\mathrm{K}^* \circ \phi_s(d) - \mathrm{K}^* \circ \phi^*(d)| + \delta - \frac{\epsilon}{\alpha \cdot k} \\
&\overset{\text{Ineq. 9}}{\Longrightarrow} \\
&g_{min} \mathop{\mathbb{E}}_{d\sim\mathcal{P}}[||\phi_H(d) - \phi^*(d)|| - ||\phi_s(d) - \phi^*(d)||] \\
&< (g_{max} - g_{min}) \mathop{\mathbb{E}}_{d\sim\mathcal{P}}||\phi_s(d) - \phi^*(d)|| \\
&\qquad\qquad + 2\delta - \frac{\epsilon}{\alpha \cdot k}
\end{aligned}$$

E Experiment Results

E.1 Consistency

The x-axis in Figure 3 represents the reward difference between the responses generated by the DPO model and those generated by the HAF model's policy head. This difference is scored by the reward model trained on the same data distribution, which we refer to as the Oracle reward model. We retain the checkpoints from the training processes of both DPO and HAF model and identify potential model pairs with similar performance using five methods (corresponding to the five colors in the figure). This similarity in performance ensures that the higher reward is not a result of better response quality. The five methods include "reward" (similar scores from the Oracle reward model), "acc" (similar binary classification accuracy), "loss" (similar

Model	Metric	Helpful	Harmless	CA	BS	AHP
Phi-2	pp_{win}	0.74	1.00	**0.60**	0.74	2.52
	pp_{lose}	0.92	0.97	1.09	**0.60**	2.55
Mistral-base	pp_{win}	0.42	0.65	0.51	**0.38**	0.75
	pp_{lose}	0.62	0.63	0.87	**0.28**	0.98
Mistral-Instruct	pp_{win}	3.50	5.13	2.33	**1.58**	1.98
	pp_{lose}	6.08	5.81	3.52	**1.31**	2.67

Table 6: Variances of the corresponding metrics. "pp" means token-wise perplexity. The subscript "win" refers to the better response while "lose" refers to the worse response.

	pp_{win}	pp_{lose}	pp_{win}-pp_{lose}
$corr$	-0.8166	-0.9492	-0.9064
p	0.0916	0.0136	0.0339

Table 7: The Pearson correlation coefficient between the variance of the token-wise perplexity of Mistral-Instruct and the difference in accuracy between the reward model trained with DPO and the accuracy of the baseline training. "corr" indicates the Pearson correlation coefficient, while "p" indicates significance.

loss values), "margin" (similar average margins of model predictions), and "step" (same training steps).

It can be observed that the differences in HAF scores are almost always higher than those from the Oracle reward model. This suggests that the preferences of the reward model are influenced by the preferences of the shared parameter policy model, providing some evidence for the existence of an Internal Preference Model.

Also shown in Figure 8, we independently trained a DPO model and a reward model using the same data and observed a strong positive correlation (even linearity) in their predictions on the test data. This indicates a significant similarity in the preference modeling processes of the DPO model and the reward model. A response preferred by the reward model will also be preferred by the DPO model, which we introduce the concept of the "Internal Preference Model" to explain.

E.2 Overall Performance

Table 6 shows the token-wise perplexity calculated by each model for each dataset.

$$pp = -\frac{\log \text{Prob}(sequence)}{\text{Length}(sequence)}$$

Another interesting finding is that the variance of the token-wise perplexity (pp) values for Mistral-Instruct shows a very strong negative correlation with the performance of the DPO reward model. Table 7 calculates the Pearson correlation coefficient between the variance of the pp values and the performance difference between the DPO reward model and the baseline reward model, indicating that this negative correlation is highly significant. This finding may provide valuable insights for aligning well-trained (but not yet well-aligned) models.

E.3 Best of N

In Table 11 we list the recall value on each dataset. We show in Figure 9 and Figure 10 the win rates on each dataset judged by `gpt-4-turbo-2024-04-09` and `ArmoRM-Llama3-8B-v0.1` (Wang et al., 2024b), respectively.

F GPT Judgement

Comparing two responses The prompt we used for judgement is listed in Table 9. The sentence between "<SYSTEM PROMPT>" is the system prompt, and the others are the user prompt. "**{question}**", "**{response 1}**", "**{response 2}**" will be replaced with the actual query or responses respectively. As GPT does not exhibit a strong "positional bias" (Wang et al., 2023), so we just randomly interchange the order of the two responses rather than prompting twice with the responses swapped.

Ranking responses Table 8 shows the consumption approximation for getting top-1, top-2 responses and the complete order out of 4/8 responses. We consider that performing a single sorting operation on eight responses with the model may result in a loss of precision. Besides, while binary comparisons exhibit high accuracy, repeated binary comparisons inevitably lead to cumulative errors and erroneous outcomes. Therefore,

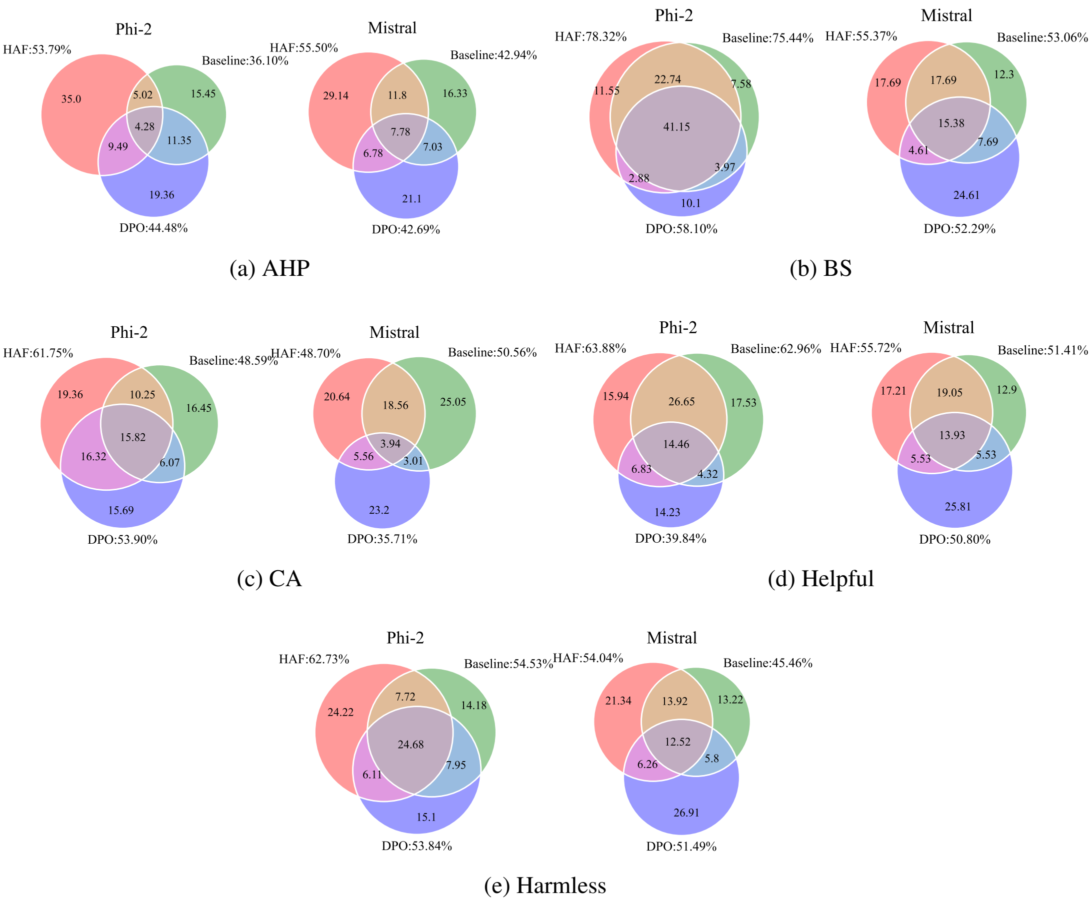

Figure 9: Win rates on each dataset judged by `gpt-4-turbo-2024-04-09`

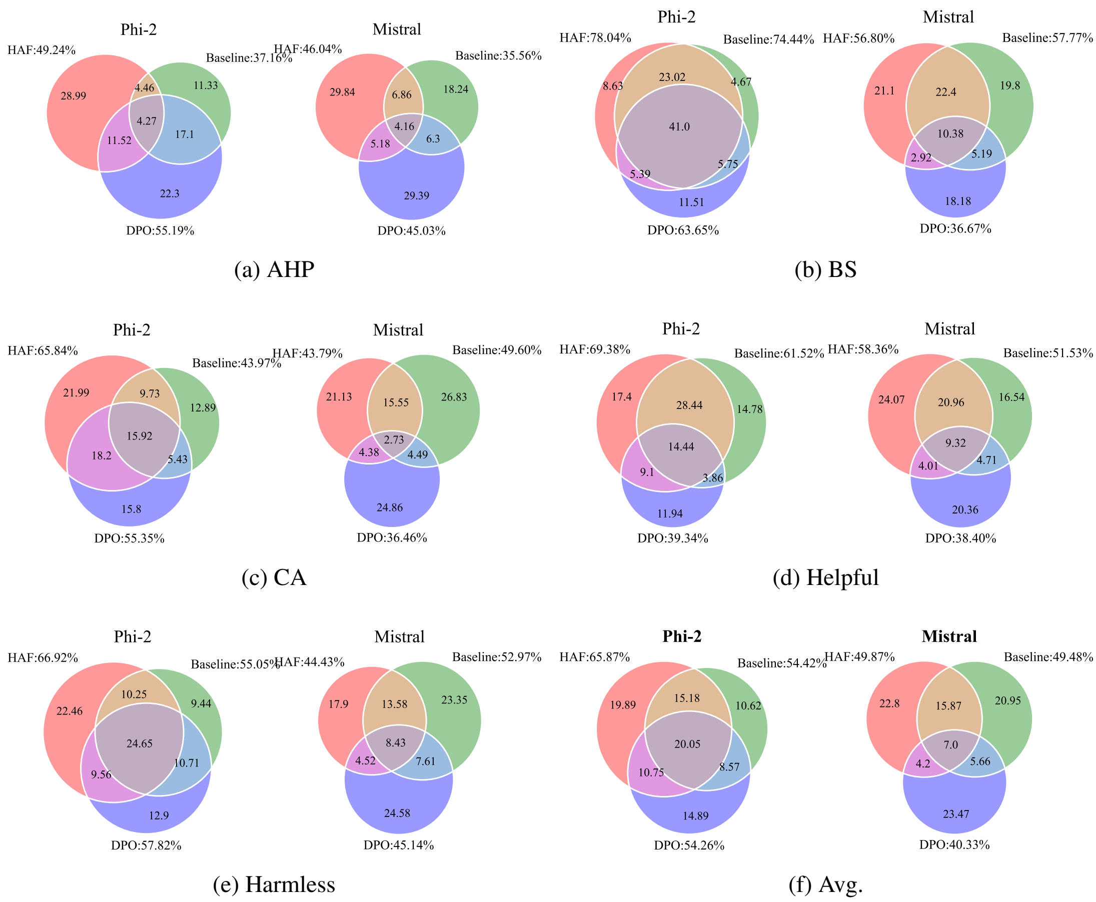

Figure 10: Win rates on each dataset judged by `ArmoRM-Llama3-8B-v0.1`

	Top-1		Top-2		Complete sort	
# responses	4	8	4	8	4	8
binary comparison	$\mathbf{6}_{3\times 2}$	$\mathbf{14}_{7\times 2}$	$\mathbf{8}_{4\times 2}$	$\mathbf{20}_{10\times 2}$	$\mathbf{10}_{5\times 2}$	$\mathbf{32}_{16\times 2}$
rank 4 responses	$\mathbf{4}_{1\times 4}$	$\mathbf{12}_{3\times 4}$	$\mathbf{4}_{1\times 4}$	$\mathbf{12}_{3\times 4}$	$\mathbf{4}_{1\times 4}$	$\mathbf{20}_{5\times 4}$
rank 8 responses	$\mathbf{4}_{1\times 4}$	$\mathbf{8}_{1\times 8}$	$\mathbf{4}_{1\times 4}$	$\mathbf{8}_{1\times 8}$	$\mathbf{4}_{1\times 4}$	$\mathbf{8}_{1\times 8}$

Table 8: Approximation for resources consumption. The first column is three different ways of interacting with GPT. The first row is the target response(s) and the second row is the number of candidate responses. "$a \times b$" means we should engage with GPT-3.5 a total of a times, with each interaction requiring an input of b responses. For example, "$\mathbf{6}_{3\times 2}$" means when using binary comparison, to get the top-1 response among 4 candidate responses, we need 3 turns of interactions with each turn requiring an input of 2 responses, hence our expenditure amounts to approximately 6 units

whether from a cost or accuracy standpoint, it is not a favorable option. In practice, we obtain the top 2 responses by ranking 4 responses with `gpt-4-turbo-2024-04-09` at once. For 8 candidate responses, we first evenly divide them into two groups and use GPT to rank the responses of each group, then we rank the two sets of the top 2 responses to get the top 2 responses among 8 candidates.

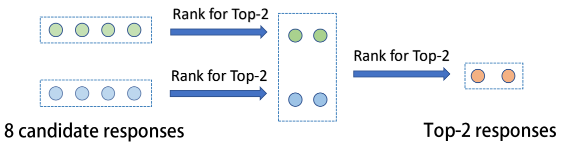

Figure 11: Three times of interactions with GPT to get top-2 responses

The prompt for ranking four responses is shown in Table 10. GPT's answer will be parsed in JSON format.

Prompt for comparing two responses.

<SYSTEM PROMPT>You are a helpful instruction-following assistant that prints the best model by selecting the best outputs for a given instruction.<SYSTEM PROMPT>
Select the output (a) or (b) that best matches the given instruction. Choose your preferred output, which can be subjective. Your answer should ONLY contain: Output (a) or Output (b).
Here's an example:

Example:
Instruction:
Give a description of the following job: "ophthalmologist"

Output (a):
An ophthalmologist is a medical doctor who pokes and prods at your eyes while asking you to read letters from a chart.

Output (b):
An ophthalmologist is a medical doctor who specializes in the diagnosis and treatment of eye diseases and conditions.

Which is best, Output (a) or Output (b)?
Output (b)

Here the answer is Output (b) because it provides a comprehensive and accurate description of the job of an ophthalmologist. In contrast, output (a) is more of a joke.

Task:
Now is the real task, do not explain your answer, just say Output (a) or Output (b).

Instruction:
{question}

Output (a):
{response 1}

Output (b):
{response 2}

Which is best, Output (a) or Output (b)?

Table 9: We use 1-shot for response comparison.

Prompt for ranking four responses.

<SYSTEM PROMPT>You are a helpful assistant, that ranks models by the quality of their answers<SYSTEM PROMPT>
I want you to create a leaderboard of different models. To do so, I will give you the instructions (prompts) given to the models, and the responses of four models. Please rank the models based on which responses would be preferred by humans. All inputs and outputs should be python dictionaries.

Here is the prompt:
```
{
    "instruction": {question},
}
```

Here are the outputs of the models:
```
[
    {
        "model": "model_1",
        "answer": {output_1}
    },
    {
        "model": "model_2",
        "answer": {output_2}
    },
    {
        "model": "model_3",
        "answer": {output_3}
    },
    {
        "model": "model_4",
        "answer": {output_4}
    }
]
```

Now please rank the models by the quality of their answers, so that the model with rank 1 has the best output. Then return a list of the model names and ranks, i.e., produce the following output:
```
[
    {"model": "model_1", "rank": <model-rank>},
    {"model": "model_2", "rank": <model-rank>},
    {"model": "model_3", "rank": <model-rank>},
    {"model": "model_4", "rank": <model-rank>}
]
```

Your response must be a valid Python dictionary and should contain nothing else because we will directly execute it in Python. Please provide the ranking that the majority of humans would give.

Table 10: We rank four responses in order of quality in a single interaction.

	AHP		BS		CA		Helpful		Harmless	
	Top-1	Top-2	Top-1	Top-2	Top-1	Top-2	Top-1	Top-2	Top-1	Top-2
Phi-2$_{\text{HAF}}$	**28.67**	**52.51**	32.49	**53.06**	**37.46**	**65.94**	**45.44**	**74.25**	**24.79**	**45.67**
Phi-2$_{\text{DPO}}$	20.85	45.62	17.68	39.71	31.89	56.07	26.42	53.07	17.87	36.67
Phi-2$_{\text{baseline}}$	15.45	34.63	**32.85**	50.90	27.84	51.64	43.62	73.91	17.41	37.48
Mistral$_{\text{HAF}}$	**22.86**	**41.95**	**14.61**	**25.38**	24.12	42.69	**16.39**	**35.24**	**12.99**	**29.23**
Mistral$_{\text{DPO}}$	14.57	32.91	10.00	24.61	13.68	30.62	13.31	31.14	12.52	25.98
Mistral$_{\text{baseline}}$	14.82	30.40	13.07	24.61	**25.05**	**43.85**	14.95	31.14	12.29	23.20

Table 11: Top-k recall for best-of-N sampling on each dataset. The results are presented as the percentage of the chosen responses included in top-k responses.