# DORM: Preference Data Weights Optimization for Reward Modeling in LLM Alignment

Rongzhi Zhang[†,*], Chenwei Zhang[‡], Xinyang Zhang[‡], Liang Qiu[‡]
Haoming Jiang[‡], Yuchen Zhuang[†], Qingru Zhang[†], Hyokun Yun[‡]
Xian Li[‡], Bing Yin[‡], Tuo Zhao[‡], Chao Zhang[†]

[†]Georgia Institute of Technology [‡]Amazon

## Abstract

Aligning large language models (LLMs) with human preferences relies heavily on high-quality reward models. However, existing approaches struggle with two critical challenges: noisy preference labels and the varying importance of preference samples. We introduce DORM, a method that enhances reward modeling by *learning to dynamically weigh* preference data. DORM initializes data importance using a combination of model uncertainty and prediction disagreement, then iteratively refines them via bilevel optimization to maximize validation performance. Using only 50k samples, DORM trains a 12B reward model that achieves 90.5% accuracy on RewardBench, matching the performance of models trained on significantly larger datasets. Furthermore, downstream alignment tasks show that fine-tuned LLMs with DORM achieve a 61.2% win rate against baseline methods, highlighting its data efficiency and generalizability.

## 1 Introduction

Aligning large language models (LLMs) with human preferences is crucial to ensure LLMs adhere to human values. Recent advances in LLM alignment have primarily focused on algorithmic innovations, ranging from reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) to alternative preference learning approaches like sequence likelihood calibration (SLiC) (Zhao et al., 2023) and direct preference optimization (DPO) (Rafailov et al., 2023). Although such algorithmic advancements have enhanced LLM alignment, the quality of preference data used for reward modeling has received limited attention.

Recent studies highlight two key challenges in reward modeling for LLM alignment: (1) **Preference data are often noisy**: Zheng et al. (2023) report 19-37% preferences provided by crowd workers are noisy. Gao et al. (2024) reveal that preference noise is observed in a wide range of tasks, including QA, Summarization, and Dialogue, where the noise rate ranges between 20-40%. (2) **Not all preference data contributes equally**: Noisy preference data forces reward models to require more training data to achieve desired performance. For instance, Wang et al. (2024a) categorize preference data into groups based on preference strength and find that some groups negatively impact reward modeling, which highlights the need for selective data curation to boost reward model performance.

Existing works addressing preference data quality fall into two categories, each with significant limitations. The first category, *heuristic-based filtering*, employs predefined quality criteria to remove low-quality data. For example, Liu and Zeng (2024) retains only top-scored data points based on reward model predictions, potentially discarding informative examples that could aid in learning decision boundaries. Dong et al. (2024) develop dataset-specific filtering rules based on response length, semantic similarity, and sentiment analysis. While effective, these rigid, manually crafted criteria often fail to capture the nuanced aspects of preference quality and require substantial human effort to adapt to new domains. On the other hand, *denoising techniques* attempts to mitigate the impact of noisy samples during model training. Yu et al. (2024) leverage LLMs to self-refine the reward difference between positive and negative pairs based on the data quality as-

[*] Work done during internship at Amazon.



*Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 22721–22739
November 4-9, 2025 

sessed through the LLMs-as-judges framework. Cheng et al. (2024) compute the KL divergence between predicted preferences and annotated labels to retain reliable preference labels and flip unreliable ones. However, these methods struggle in achieving both robust denoising performance and high data efficiency.

These existing approaches face three key limitations: (1) they often rely on heuristics or expensive oracle feedback, (2) they treat ambiguous preferences as unreliable samples rather than potential learning opportunities, and (3) they fail to adaptively adjust data importance based on the reward modeling progress. These limitations raise a critical research question: Can we develop a learning-based approach to automatically estimate the importance of preference data, thereby enhancing reward modeling?

We address this challenge by introducing DORM (preference **D**ata weights **O**ptimization for **R**eward **M**odeling) – a two-stage approach that combines preference data quality estimation with adaptive data weighting. The key intuition of the first stage is that data points where the model is uncertain are often informative and should be emphasized, whereas mislabeled or unreliable samples should be down-weighted. To this end, we estimate preference data weights by integrating *model uncertainty* and *prediction disagreement*. Specifically, we employ approximate Bayesian inference techniques for uncertainty estimation (Gal and Ghahramani, 2016), while measuring disagreement by the discrepancy between predictions and labels. This procedure prioritizes informative and reliable data and mitigates the impact of unreliable data.

The second stage further refines data weighting by dynamically adjusting weights through a bilevel optimization framework. This approach maximizes validation performance while using initial uncertainty-based data weights as regularization. The framework consists of two levels: (1) The *upper-level optimization* adjusts data weights to minimize validation loss, guided by initial weight estimates; (2) The *lower-level optimization* trains a reward model using the weights determined by the upper-level problem. This learning process automatically identifies which data points are most useful for improving validation performance while mitigating overfitting through uncertainty-based regularization.

To evaluate our method, we conduct experiments on both the reward model and policy model levels. Our 12B reward model, trained with only 50k preference data samples, achieves 90.5% overall performance on RewardBench, matching the performance of similar-sized models trained with significantly more data. Analysis of data weight tracking reveals that high-quality datasets are progressively assigned greater weights, while noisy samples are gradually down-weighted. Furthermore, the policy model aligned using our reward model shows $61.2\%$ win rates compared to using the baseline reward model.

Our contributions can be summarized as follows: 1) **Preference Data Quality Estimation:** We introduce a method for estimating preference data quality by combining epistemic uncertainty and disagreement measures. 2) **Bilevel Optimization Framework:** We formulate learning to weigh preference data as a bilevel optimization problem, enabling data-driven refinement of data weights while incorporating initial quality estimates. 3) **Data-Efficient Reward Model Training:** Our approach achieves $90.5\%$ accuracy on RewardBench, with $10-40\times$ less preference data compared to baselines.

## 2 Preliminaries

**Multi-Objective Reward Modeling.** Let $\mathcal{X}$ and $\mathcal{Y}$ denote the space of prompts and responses, respectively. Unlike traditional reward models that rely solely on pairwise preferences (Bradley and Terry, 1952; Ouyang et al., 2022; Bai et al., 2022a; Rafailov et al., 2023), multi-objective reward modeling leverages fine-grained ratings across multiple reward attributes (Wang et al., 2024b,g,f) to capture richer preference signals. In this setup, human feedback is collected in the form of structured ratings, where each response $y \in \mathcal{Y}$ is assigned a rating vector $r \in \mathbb{R}^m$, with each dimension corresponding to a different reward attribute.

Given a dataset $\mathcal{D} = \{(x_i, y_i, \mathbf{r_i})\}_{i=1}^{|\mathcal{D}|}$, where $x_i$ represents the input prompt, $y_i$ is the generated response, and $\mathbf{r_i}$ is the associated multi-

dimensional rating, the objective is to learn a reward model $r_\theta$ that predicts these ratings accurately. Specifically, given an input pair $(x, y)$, we concatenate the prompt and response as $x \oplus y$ and pass it through a pre-trained decoder-only LLM $\mathcal{M}_\psi$, extracting a $d$-dimensional representation from the final decoder layer. A linear regression head $V \in \mathbb{R}^{d \times m}$ is then applied to produce the predicted rating vector, which is optimized using the following regression loss:

$$\min_{\psi, V} \mathbb{E}_{x,y,\mathbf{r} \sim \mathcal{D}} \left\| V^\top f_\psi(x \oplus y) - \mathbf{r} \right\|_2^2. \quad (1)$$

**Projection for Scalar Metrics.** Since the reward model operates in a multi-objective setting, where each response $y$ is evaluated across $m$ different reward attributes, it is often necessary to project the reward vector $\mathbf{r} \in \mathbb{R}^m$ onto a scalar space. To achieve this, we introduce a projection vector $\lambda \in \mathbb{R}^m$, which allows us to map the multi-dimensional reward vector onto a single scalar value by $r = \lambda^\top \mathbf{r}$.

**Quality-based Data Weighting.** Given the dataset $\mathcal{D}$, we estimate the quality of each data point to derive appropriate weights for reward modeling. Let $s_i \in \mathbb{R}$ denote the estimated quality score for each sample $(x_i, y_i, \mathbf{r_i})$. The data weight $w_i$ is derived as $w_i = h(s_i)$, where $h : \mathbb{R} \to \mathbb{R}^+$ is a monotonically increasing function that maps quality measures to data weights. These weights are then incorporated into the reward modeling objective:

$$\min_{\psi, V} \mathbb{E}_{(x,y,\mathbf{r}) \sim \mathcal{D}} w \left\| V^\top f_\psi(x \oplus y) - \mathbf{r} \right\|_2^2. \quad (2)$$

This formulation allows the model to adaptively focus on samples based on their estimated quality during training.

## 3 Preference Data Weight Optimization with Estimated Quality

In this section, we formulate the weight optimization for preference data as a bilevel optimization problem. Our approach operates in two stages. In the first stage, we estimate initial data weights based on epistemic uncertainty (Gal and Ghahramani, 2016; Hüllermeier and Waegeman, 2021) and a disagreement function. These measures help identify more reliable and informative data points, guiding the model towards higher-quality examples. The second stage employs a bilevel optimization framework (Kwon et al., 2023a). The lower level optimizes reward model parameters using weighted preference data, while the upper level refines these weights to minimize loss on a high-quality validation set. Critically, we incorporate the initial weights as regularization terms in the upper-level problem, anchoring the optimization to the prior quality estimates and stabilizing the process. We present our method overview in Figure 1.

### 3.1 Quality-aware Preference Data Weighting

In the context of preference data optimization, it is crucial to account for the varying quality of data samples (Wang et al., 2024a). Assigning appropriate weights to each data point based on its estimated quality can enhance model performance by emphasizing informative samples and down-weighting noisy or unreliable ones. To achieve this, we propose a method to estimate prior weights for each data point by leveraging measures of *epistemic uncertainty* and *prediction disagreement*.

**Estimating Epistemic Uncertainty.** For each data point $x_i$, we estimate the epistemic uncertainty using Monte Carlo (MC) dropout (Gal and Ghahramani, 2016)[1]. We perform $N$ stochastic forward passes through the reward model, where dropout introduces randomness by deactivating different subsets of neurons in each pass. This process yields a set of predictions $\{\hat{r}_{i,1}, \hat{r}_{i,2}, \ldots, \hat{r}_{i,N}\}$ for the data point $x_i$.

To compute the uncertainty, we first compute the mean prediction by $\bar{r}_i = \frac{1}{N} \sum_{n=1}^{N} \hat{r}_{i,n}$, and then take the variance as uncertainty:

$$u_i = \sigma_i^2 = \frac{1}{N} \sum_{n=1}^{N} \left( \hat{r}_{i,n} - \bar{r}_i \right)^2. \quad (3)$$

Here, $u_i$ represents the epistemic uncertainty associated with $x_i$. A higher variance $u_i$ indicates that the model's predictions are more sensitive to changes in its internal parameters, reflecting less confidence in its output for that data point.

[1]While we use MC dropout for simplicity and computational efficiency, our method can work with other Bayesian uncertainty estimation techniques.

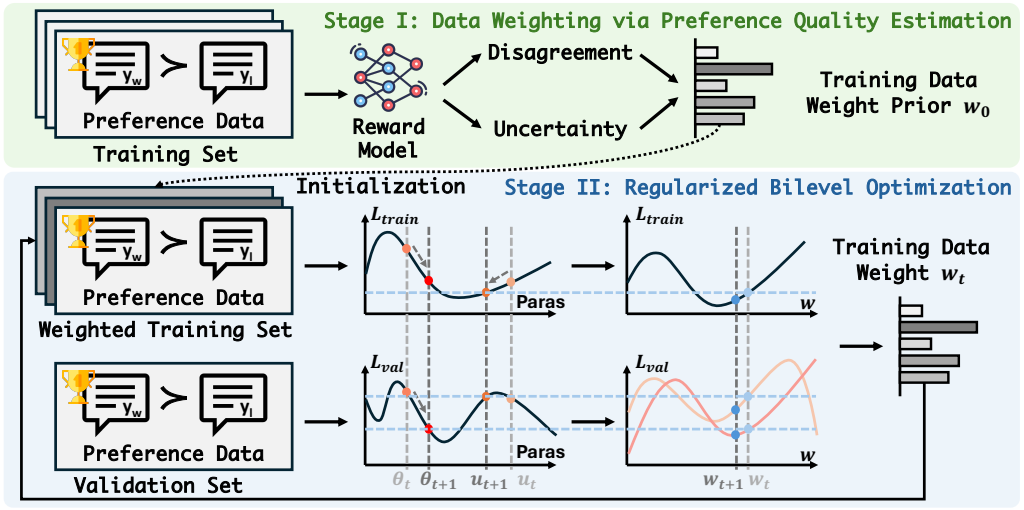


Figure 1: Overview of the two-stage bilevel optimization framework for reward modeling. The first stage assigns initial data weights based on epistemic uncertainty and disagreement. The second stage refines these weights in a bilevel optimization process, using a validation set and the initial weights as regularization.

**Measuring Prediction Disagreement.** To assess the alignment between the model's predictions and labels, we define a *disagreement function* $q_i$ for each data point as:

$$q_i = \frac{|r_i - \bar{r}_i|}{\max_n \hat{r}_{i,n} - \min_n \hat{r}_{i,n}}, \tag{4}$$

where $r_i$ is the label, $\bar{r}_i$ is the mean prediction from the MC dropout ensemble, the normalization ensures that $q_i$ is scale-invariant.

A higher value of $q_i$ suggests a greater discrepancy between the model's prediction and the label, potentially indicating mislabeling or noise in the data.

**Defining Prior Weights** To effectively prioritize data points during training, we combine the epistemic uncertainty $u_i$ and the disagreement measure $q_i$ to formulate the prior weight $w_i^0$ for each data point:

$$w_i^0 = \exp\left(u_i - \gamma q_i\right), \tag{5}$$

where $\gamma > 0$ is a hyperparameter controlling the balance of uncertainty and disagreement.

This formulation is grounded in the intuition that data points with high epistemic uncertainty (large $u_i$) are situated in regions where the model lacks confidence but have the potential to provide significant learning gains. By assigning higher weights to these points, we encourage the model to focus on inputs where additional information could most improve its performance.

Conversely, the disagreement measure $q_i$ serves as a penalty term. Data points where the model's predictions significantly deviate from the labels (large $q_i$) may indicate mislabeling or noisy data. By subtracting $\gamma q_i$ in the exponent, we reduce the weights of these potentially unreliable points, thereby mitigating their influence on the training process. For further empirical evidence and characterization of how this weighting formulation differentiates between hard and noisy samples, please refer to Appendix H.

**Interpretation in Extreme Cases.** The prior weight formulation behaves intuitively under extreme scenarios. When $u_i \to \infty$ and $q_i \approx 0$, the weight $w_i^0$ becomes very large. This indicates that the data point is highly informative (high uncertainty) and reliable (low disagreement), deserving significant emphasis during training. Conversely, when $u_i \approx 0$ and $q_i \to \infty$, the weight $w_i^0$ approaches zero. In this case, the model is confident in its prediction (low uncertainty) but disagrees with the label (high disagreement), suggesting possible mislabeling; down-weighting such points prevents them from negatively impacting the model. For data points where both uncertainty and disagreement are high, the weight depends on the relative values of $u_i$ and $\gamma q_i$, allowing for careful consideration based on the specified hyperparameters.

### 3.2 Regularized Bilevel Optimization

To incorporate prior knowledge about data quality into our optimization framework, we include a regularization term for the upper-level opti-

**Algorithm 1** Solve the Bilevel Optimization Problem via A First-order Hypergradient Method

**Require:** initialization $w_0, u_0, \theta_0$, learning rates $\{\eta_w, \eta_u, \eta_\theta\}$, and coefficients $\alpha, \beta$

**for** $t = 0$ **to** $T - 1$ **do**
  Sample mini-batches $\{D_{\text{tr}}^t, D_{\text{val}}^t\}$ from the training set and the validation set $\{D_{\text{tr}}, D_{\text{val}}\}$
  $u_{t+1} = u_t - \alpha\eta_u \nabla_u L_{tr}(w_t, u_t; D_{\text{tr}}^t)$
  $\theta_{t+1} = \theta_t - \eta_\theta(\nabla_\theta L_{val}(w_t, \theta_t; D_{\text{val}}^t) + \alpha \nabla_\theta L_{tr}(w_t, \theta_t; D_{\text{tr}}^t))$
  $w_{t+1} = w_t - \eta_w(\nabla_w L_{val}(w_t, \theta_t; D_{\text{val}}^t) + 2\beta(w_t - w_i^0) + \alpha(\nabla_w L_{tr}(w_t, \theta_t; D_{\text{tr}}^t) - \nabla_w L_{tr}(w_t, u_t; D_{\text{tr}}^t)))$
**end for**
**return** $(w_T, \theta_T, u_T)$

mization problem that penalizes deviations from the estimated prior weights. Specifically, we use the prior weights $w_i^0$ derived from the measures of epistemic uncertainty and prediction disagreement discussed earlier.

Let $\theta$ represent the model parameters, and $w = [w_1, w_2, \ldots, w_k] \in \mathbb{R}^k$ be the vector of weights assigned to each training sample $s_i$, where $i = 1, \ldots, k$ indexes over the training data, and $k$ is the total number of samples. We formulate the bilevel optimization problem as:

$$\begin{aligned} &\min_{w \in \mathcal{W}} L_{val}(\theta^*(w)) + \beta \sum_{i=1}^{k} (w_i - w_i^0)^2 \\ &\text{s.t.} \quad \theta^*(w) = \arg\min_{\theta} \sum_{i=1}^{k} w_i L_{tr}(\theta, s_i) \end{aligned} \tag{6}$$

where $L_{\text{tr}}(\theta, s_i)$ and $L_{\text{val}}(\theta^*(w))$ are the training and validation loss, respectively, and $\beta > 0$ is a hyperparameter controlling the strength of the regularization. This L2 regularization encourages the optimized weights to stay close to the prior weights while allowing for data-driven adjustments. The gradient with respect to each weight $p_i$ is simply:

$$\frac{\partial L}{\partial w_i} = \frac{\partial L_{val}}{\partial w_i} + 2\beta(w_i - w_i^0) \tag{7}$$

Here we choose L2 regularization instead of KL divergence because this formulation provides a simpler, more computationally efficient way to incorporate prior knowledge, avoiding the numerical instabilities associated with KL divergence while allowing for unconstrained optimization of weights.

### 3.3 Solve the Bilevel Optimization via A First-order Hypergradient Method

Solving the bilevel optimization problem in Eq. (6) directly is challenging due to the nested optimization and the dependency of the upper-level objective on the lower-level solution $\theta^*(w)$. To address this, we adopt a fully first-order method inspired by (Kwon et al., 2023a; Lu and Mei, 2024), which circumvents the computational burdens associated with second-order derivatives and is well-suited for stochastic optimization settings.

We begin by reformulating the bilevel problem as a single-level optimization with an equality constraint representing the optimality of the lower-level problem:

$$\begin{aligned} &\min_{w \in \mathcal{W}, \theta} L_{val}(w, \theta) + \beta \sum_{i=1}^{k} (w_i - w_i^0)^2 \\ &\text{s.t.} \quad L_{tr}(w, \theta) - \min_{u} L_{tr}(w, u) = 0. \end{aligned} \tag{8}$$

Here an auxiliary variable $u$ is introduced to transform the lower problem $\theta^*(w) = \arg\min_\theta L_{tr}(w, \theta)$ to be the constraint $L_{tr}(w, \theta) - \min_u L_{tr}(w, u) = 0$ where $u$ serves as the proxy of $\theta^*(w)$. This leads to a minimax problem defined as $\min_{w \in \mathcal{W}} \max_u \mathcal{L}^\alpha(w, \theta, u)$, where

$$\begin{aligned} \mathcal{L}^\alpha(w, \theta, u) = L_{val}(w, \theta) + \beta \sum_{i=1}^{k} (w_i - w_i^0)^2 \\ + \alpha(L_{tr}(w, \theta) - L_{tr}(w, u)). \end{aligned} \tag{9}$$

This reformulation circumvents the upper-lower dependency in the original bilevel optimization via an equivalent min-max problem. We summarize our algorithm in Algorithm 1.

## 4 Experiments

### 4.1 Experiment Setup

**Datasets.** We experiment with five datasets, with details provided in Appendix A.

- *HelpSteer2* (Wang et al., 2023): the most recent high-quality human-annotated preference data, a follow-up to the popular HelpSteer.

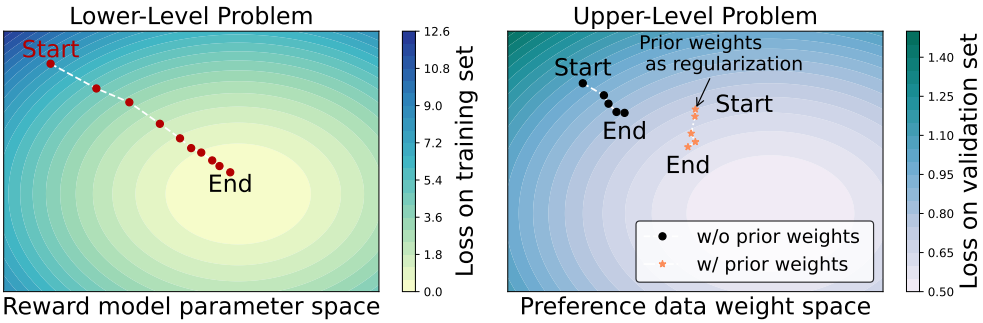


Figure 2: Regularized bilevel optimization for reward modeling, where the prior weights are initialized from preference data quality estimation.

- *OpenAssistant2* (Köpf et al., 2024): human-annotated assistant-style conversation corpus with a multi-dimension label.
- *Magpie-QWen-2.5* (Xu et al., 2024): a synthetically generated dataset for supervised fine-tuning using Qwen2-72B-Instruct.
- *OffsetBias* (Park et al., 2024): a pairwise preference dataset intended to reduce common biases inherent in judge models.
- *WildGuard* (Han et al., 2024): with examples designed to evaluate the safety of LLM responses under various conditions.

**Baselines.** We consider two baseline groups:

- *LLM-as-a-judge*: These models generate a preference label given a prompt with a pair of responses as input. We include Llama-3.1-405B (Dubey et al., 2024), GPT-4 (Achiam et al., 2023), GPT-4o (Hurst et al., 2024), Gemini-1.5-pro (Reid et al., 2024) and self-taught evaluator (Wang et al., 2024d) which is based on Llama-3-70B.

- *Standard Reward Models*: This category consists of models those explicitly trained on preference data. We compare against standard RM (Stiennon et al., 2020), Cohere-0514, Llama3-70B-SteerLM-RM (Wang et al., 2024f), URM-Llama3-8B (Lou et al., 2024), ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024c), Pair-preference-model-LLaMA3-8B (Dong et al., 2024), and Internlm2-20B-reward (Cai et al., 2024).

**Evaluation.** We evaluate DORM at both the reward model and policy model levels. For reward model performance, we benchmark on RewardBench (Lambert et al., 2024). For policy model evaluation, we conduct experiments on UltraFeedback (Cui et al., 2023).

**Implementation.** We use Mistral NeMo 12B-Instruct (NVIDIA and Mistral AI) as the backbone of our reward model. We set multi-attribute head (total of 12 attributes for our recipe) with regression loss for reward modeling. Additional implementation details are provided in Appendix C.

### 4.2 Main Experiments on RewardBench

Table 1 presents the main results on RewardBench, we summarize the findings below.

- **Learning to weigh preference data helps:** DORM outperforms the best LLM-as-a-judge baseline by $+2.2\%$ in the overall RewardBench score, demonstrating that incorporating preference data weighting leads to better reward modeling accuracy than direct response ranking by large LLMs. Furthermore, DORM achieves performance on par with the strongest standard reward model baselines (e.g., *ArmoRM-Llama3-8B-v0.1*), showing that our weighting strategy effectively optimizes the use of available preference data without requiring significantly larger model capacity or data volume.

- **Significant data efficiency:** While achieving comparable performance to the strongest reward model baselines, our approach uses only 50k preference data, which is $10\times$ less than *ArmoRM-Llama3-8B-v0.1* and *Internlm2-20B-reward* $40\times$ less than *Internlm2-20B-reward*. This demonstrates the data efficiency of our approach. We further analyze the data efficiency in Appendix B.

- **Improvements on challenging subtasks:**

| Models | Chat | Chat_Hard | Reasoning | Safety | Overall |
|---|---|---|---|---|---|
| *LLM-as-a-judge* | | | | | |
| Llama3.1-405B-Instruct (Dubey et al., 2024) | 97.2 | 74.6 | 77.6 | 87.1 | 84.1 |
| GPT-4-0125 (Achiam et al., 2023) | 95.3 | 74.3 | 87.6 | 86.9 | 86.0 |
| GPT-4o-0806 (Hurst et al., 2024) | 96.1 | 76.1 | 88.1 | 86.6 | 86.7 |
| Gemini-1.5-pro-0514 (Reid et al., 2024) | 92.3 | 80.6 | 92.0 | 87.9 | 88.2 |
| Self-taught Evaluator (Wang et al., 2024d) | 96.6 | 84.2 | 91.5 | 81.0 | 88.3 |
| *Standard Reward Models* | | | | | |
| RM (Stiennon et al., 2020) | 98.3 | 74.5 | 88.0 | 83.8 | 86.4 |
| Cohere-0514 | 96.4 | 71.3 | 92.3 | 97.7 | 89.4 |
| Llama3-70B-SteerLM-RM (Wang et al., 2024f) | 91.3 | 80.3 | 90.6 | 92.8 | 88.8 |
| URM-Llama3-8B (Lou et al., 2024) | 96.9 | 78.7 | 95.7 | 88.2 | 89.9 |
| ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024c) † | 96.9 | 76.8 | 97.3 | 90.5 | 90.4 |
| Pair-preference-model-LLaMA3-8B (Dong et al., 2024) † | 98.3 | 65.8 | 94.7 | 89.7 | 87.1 |
| Internlm2-20B-reward (Cai et al., 2024)† | 98.9 | 76.5 | 95.8 | 89.5 | 90.2 |
| DORM | 95.6 | 83.4 | 93.1 | 89.8 | 90.5 |

Table 1: Main results on RewardBench, † indicates models trained with at least x10 preference data. The baselines are selected based on contamination awareness, public reproducibility, and parameter count fairness (see Appendix C for detailed discussion of selection principles).

DORM exhibits notable improvements on Reasoning (+1.5%) compared to the best LLM-as-a-judge baseline and Chat_Hard (+3.1%) compared to the best RM baseline. This suggests that our data weighting strategy effectively prioritizes informative and reliable preference data, enhancing model performance on difficult tasks.

### 4.3 Training Cost Analysis.

While bilevel optimization is typically considered computationally intensive, DORM leverages a first-order method that avoids second-order Hessian computation. As detailed in Appendix E, despite introducing bilevel optimization, DORM remains computationally feasible at scale. The per-epoch training time increases moderately from 2.4 hours (standard SFT) to 3.5 hours, yet this overhead results in a significant RewardBench performance improvement—from 85.6 to 91.5—highlighting the efficiency of our first-order hypergradient method and the strong return on added computation.

### 4.4 Component Effectiveness

We study the effectiveness of each component in our two-stage method. Specifically, we evaluate whether applying the prior weights and the bilevel optimization alone can improve the performance, and if the integration of them can bring extra benefits. As shown in Table 2, we compare the entire method with the direct SFT baseline, the method with only prior weights, and the method of bilevel optimization without prior weights. We found that both prior weighting and bilevel optimization individually enhance the results over the simple SFT approach. Applying only prior weights leads to a noticeable improvement, especially on the subtask of chat hard, and employing bilevel optimization without prior weights yields a substantial gain on subtasks of chat hard and safety. Moreover, incorporating both prior weights and bilevel optimization together further boosts the performance, achieving the highest overall score. This indicates that the two components are complementary, and their combination yields better outcomes than using either alone.

| Method | Reward-Bench Subsets | | | | |
|---|---|---|---|---|---|
| | Chat | Chat Hard | Safety | Reasoning | Avg |
| SFT | 90.7 | 72.8 | 86.7 | 91.1 | 85.3 |
| Prior weights only | 91.9 | 77.0 | 85.2 | 91.6 | 86.4 |
| Bilevel w/o prior | 93.2 | 81.3 | 89.8 | 92.2 | 89.1 |
| DORM | 94.7 | 84.8 | 88.5 | 92.9 | 90.2 |

Table 2: Study of component effectiveness. For methods involving bilevel optimization, we have 3.5k validation data for the upper-level optimization. For a fair comparison, we incorporate the validation data to the training set for the SFT method and the method with prior weights only.

### 4.5 Policy Model Alignment Results

We evaluate the effectiveness of our trained reward models in downstream policy alignment through DPO (Rafailov et al., 2024). We use the

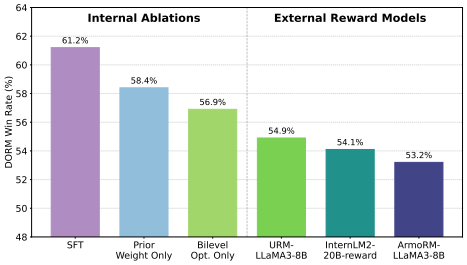


Figure 3: Policy model alignment results.

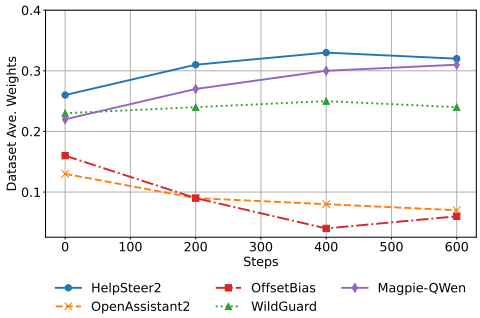


Figure 4: Data weights tracking in RM training.

UltraFeedback (Cui et al., 2023) dataset, splitting its 64k samples into 20k for SFT, 40k for DPO, and 4k for testing. A Mistral-7B model serves as the policy, which is first fine-tuned for one SFT epoch, followed by two DPO epochs. To evaluate alignment quality, we compute win rate using an independent RM – pair-preference-model-LLaMA3-8B (Dong et al., 2024).

We compare DORM-trained reward models against several baselines in terms of the policy win rate after DPO. These include both strong internal ablations—such as SFT-trained reward models, prior-weight-only training, and bilevel optimization without prior—and top-performing publicly available reward models from the RewardBench leaderboard, including ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024c), Internlm2-20B-reward (Cai et al., 2024), and URM-Llama3-8B (Lou et al., 2024). All reward models are used to generate preference labels on the same DPO training set.

Figure 3 shows DORM outperforms both internal ablations and strong external reward models in terms of win rate when aligning the policy model. The consistent performance advantage across both reward modeling and alignment tasks indicates that DORM not only improves reward modeling accuracy but also enhances policy alignment - the primary objective of reward modeling. This end-to-end improvement suggests that the enhanced reward signals produced by DORM translate effectively into better-aligned policies.

### 4.6 Track Data Weights Change

Figure 4 demonstrates the data weights dynamics by tracking the average weights assigned to studied datasets. HelpSteer2 consistently receives the highest weights, increasing from 0.26 to 0.32, suggesting DORM identifies it as the most informative and reliable dataset. Magpie-QWen-2.5 shows moderate importance with gradual weight increase from 0.22 to 0.31, while WildGuard maintains relatively stable weights around 0.24. OpenAssistant2 and OffsetBias receive decreasing weights (0.13 to 0.07 and 0.16 to 0.06), indicating DORM identifies them as noisy or less reliable data and down-weights them based on the validation loss. These varying trajectories demonstrate DORM's capability to automatically adjust data importance based on their contribution to the learning objective.

## 5 Related Work

### 5.1 Preference Learning for LLM Alignment

Preference learning is crucial for aligning LLMs with human intent. Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Dubey et al., 2024; Reid et al., 2024) trains a reward model on human preferences to guide policy optimization but is susceptible to reward hacking (Skalse et al., 2022; Liu et al., 2024b) and training instability (Engstrom et al., 2020; Wang et al., 2022).

To mitigate these issues, alternative methods have been proposed. Direct Preference Optimization (DPO) (Rafailov et al., 2023) bypasses the explicit reward model, while methods like Sequence Likelihood Calibration (SLiC) (Zhao et al., 2023), reward model distillation (Bai et al., 2022a; Zhang et al., 2024a), and test-time alignment (Kong et al., 2024) also offer more stable, non-reinforcement learning approaches. Recent works incorporate uncertainty estimation into reward models. For instance, Uncertainty-aware Reward Model (URM) (Lou et al., 2024) introduces a probabilistic value head to model aleatoric uncertainty and uses an ensemble-based approach to quantify epis-

temic uncertainty. In contrast, our method performs uncertainty estimation with a single reward model and, more importantly, introduces a dynamic data weighting mechanism to address preference data quality, a dimension largely overlooked by prior methods.

### 5.2 Quality-aware LLM Alignment

Recent studies confirm that the quality of preference data significantly impacts LLM alignment, noting that datasets often contain noise (Zhang et al., 2023a,b; Chen et al., 2024; Wang et al., 2024e; Zhang et al., 2024b; Gao et al., 2024; Wang et al., 2024a). Existing approaches to address this fall into the following categories.

Heuristic-based filtering methods aim to select high-quality data based on predefined criteria, such as reward scores (Liu and Zeng, 2024) or dataset-specific rules (Dong et al., 2024). However, these heuristics can discard informative examples or require considerable human effort to generalize across diverse datasets. Denoising techniques aim to improve data quality during training, for example, by self-refining reward differences (Yu et al., 2024) or using a discriminator to filter preference labels (Cheng et al., 2024), but they face challenges in balancing efficiency and robustness. An alternative direction is synthetic data generation. Methods proposed by (Bai et al., 2022b) and (Zhu et al., 2023) replace human feedback with AI-generated feedback conditioned on human-written principles. While this can potentially scale the creation of preference data, it introduces new challenges in ensuring the quality and diversity of synthetic data.

These methods often depend on heuristic rules or external oracle feedback, struggle to leverage ambiguous preferences as informative training signals, or lack mechanisms to dynamically adjust data importance throughout training. To overcome these challenges, DORM equips a quality-aware preference data weighting strategy and enables dynamic weighting in the reward modeling process.

### 5.3 Data Weighting and Bilevel Optimization.

Data weighting strategies are effective for improving model robustness against noisy labels (Ren et al., 2018; Zhang et al., 2020; Zhang and Pfister, 2021; Wu et al., 2022). Bilevel optimization provides a principled framework for learning these weights by optimizing a validation objective (Wu et al., 2022; Pan et al., 2024). However, traditional second-order methods often suffer from computational inefficiencies due to Hessian-vector product computations (Domke, 2012; Franceschi et al., 2017; Ji et al., 2020). Recent advancements address these computational bottlenecks through first-order hypergradient approximations, allowing bilevel optimization to scale to large neural networks (Kwon et al., 2023a; Pan et al., 2024). For example, Pan et al. (2024) introduced *ScaleBiO*, a scalable first-order method designed for data reweighting in the instruction-following training stage of LLMs.

Our method targets a distinct scope — the reward modeling stage — and significantly differs in its methodological design. We introduce a customized weighting function explicitly tailored for preference data as an initial quality estimator, which is subsequently incorporated as a prior-based regularization term within the bilevel optimization framework. This design effectively incorporates prior knowledge, while enhancing the convergence and training stability of the bilevel optimization problem.

## 6 Conclusions

In this study, we presented DORM to tackle the critical issue of preference data quality estimation in aligning LLMs. By integrating epistemic uncertainty with a disagreement measure, we developed a method to assess the informativeness and reliability of each preference data point. By utilizing these quality estimates as prior weights and refining them through a bilevel optimization framework, we balance prior knowledge with data-driven insights and enhance the robustness of reward models to handle diverse preference data. DORM achieves high performance with significantly less data, leading to more robust reward models and better-aligned policy models. This work highlights the importance of data quality in model alignment and provides a promising avenue for developing more reliable and human-aligned language models.

## Limitations

### 6.1 Dataset-level Weight Assignment

For computational efficiency, our current implementation assigns weights at the dataset level. While this simplification maintains good performance and computational efficiency, it may not fully capture the fine-grained quality variations within each dataset. Future work could explore hierarchical weight optimization strategies for finer-grained assignments.

### 6.2 Dependence on a High-Quality Validation Set

DORM relies on a validation set for bilevel optimization, which is essential for guiding dynamic data reweighting. However, the effectiveness of this approach depends on the quality of the validation set. A low-quality validation set with inconsistent or incorrect preference labels may lead to suboptimal weight adjustments, compromising reward model reliability.

That said, the validation set used in our approach is relatively small, making it feasible in practice to acquire high-quality human annotations. Given its critical role in optimizing preference weights, prioritizing label accuracy in the validation set is a worthwhile investment. Future work could further explore ways to refine validation selection, such as automated validation data curation or self-refinement techniques to enhance label quality.

## References


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

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

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022b. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. 2024. Internlm2 technical report.

Guiming Hardy Chen, Shunian Chen, Ziche Liu, Feng Jiang, and Benyou Wang. 2024. Humans or llms as the judge? a study on judgement biases. *arXiv preprint arXiv:2402.10669*.

Jie Cheng, Gang Xiong, Xingyuan Dai, Qinghai Miao, Yisheng Lv, and Fei-Yue Wang. 2024. Rime: Robust preference-based reinforcement learning with noisy preferences. *arXiv preprint arXiv:2402.17257*.

Ganqu Cui, 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.

Justin Domke. 2012. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, pages 318–326. PMLR.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. 2020. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*.

Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. 2017. Forward and reverse gradient-based hyperparameter optimization. In *International conference on machine learning*, pages 1165–1173. PMLR.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR.

Yang Gao, Dana Alon, and Donald Metzler. 2024. Impact of preference noise on the alignment performance of generative language models. *arXiv preprint arXiv:2404.09824*.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. 2024. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms.

Eyke Hüllermeier and Willem Waegeman. 2021. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine learning*, 110(3):457–506.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

K Ji, J Yang, and Y Liang. 2020. Bilevel optimization: Convergence analysis and enhanced design. arxiv e-prints, art. *arXiv preprint arXiv:2010.07962*.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lingkai Kong, Haorui Wang, Wenhao Mu, Yuanqi Du, Yuchen Zhuang, Yifei Zhou, Yue Song, Rongzhi Zhang, Kai Wang, and Chao Zhang. 2024. Aligning large language models with representation editing: A control perspective. *Advances in Neural Information Processing Systems*, 37:37356–37384.

Andreas Köpf, 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.

Jeongyeol Kwon, Dohyun Kwon, Stephen Wright, and Robert D Nowak. 2023a. A fully first-order method for stochastic bilevel optimization. In *International Conference on Machine Learning*, pages 18083–18113. PMLR.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023b. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

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.

Chris Yuhao Liu and Liang Zeng. 2024. Skywork reward model series. `https://huggingface.co/Skywork`.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024a. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Tianqi Liu, Wei Xiong, Jie Ren, Lichang Chen, Junru Wu, Rishabh Joshi, Yang Gao, Jiaming Shen, Zhen Qin, Tianhe Yu, et al. 2024b. Rrm: Robust reward model training mitigates reward hacking. *arXiv preprint arXiv:2409.13156*.

Xingzhou Lou, Dong Yan, Wei Shen, Yuzi Yan, Jian Xie, and Junge Zhang. 2024. Uncertainty-aware reward model: Teaching reward models to know what is unknown. *arXiv preprint arXiv:2410.00847*.

Zhaosong Lu and Sanyou Mei. 2024. First-order penalty methods for bilevel optimization. *SIAM Journal on Optimization*, 34(2):1937–1969.

NVIDIA and Mistral AI. Mistral-NeMo-12B-Instruct. `https://huggingface.co/nvidia/Mistral-NeMo-12B-Instruct`.

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.

Rui Pan, Jipeng Zhang, Xingyuan Pan, Renjie Pi, Xiaoyu Wang, and Tong Zhang. 2024. Scalebio: Scalable bilevel optimization for llm data reweighting. *arXiv preprint arXiv:2406.19976*.

Junsoo Park, Seungyeon Jwa, Meiying Ren, Daeyoung Kim, and Sanghyuk Choi. 2024. Offsetbias: Leveraging debiased data for tuning evaluators. *arXiv preprint arXiv:2407.06551*.

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 preprint arXiv:2305.18290*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, and Timothy Lillicrap. 2024. Alayrac, et al. gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 1(3):5.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471.

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

Binghai Wang, Rui Zheng, Lu Chen, Yan Liu, Shihan Dou, Caishuang Huang, Wei Shen, Senjie Jin, Enyu Zhou, Chenyu Shi, et al. 2024a. Secrets of rlhf in large language models part ii: Reward modeling. *arXiv preprint arXiv:2401.06080*.

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

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024c. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. 2024d. Self-taught evaluators. *arXiv preprint arXiv:2408.02666*.

Xu Wang, Sen Wang, Xingxing Liang, Dawei Zhao, Jincai Huang, Xin Xu, Bin Dai, and Qiguang Miao. 2022. Deep reinforcement learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):5064–5078.

Yuan Wang, Xuyang Wu, Hsin-Tai Wu, Zhiqiang Tao, and Yi Fang. 2024e. Do large language models rank fairly? an empirical study on the fairness of llms as rankers. *arXiv preprint arXiv:2404.03192*.

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

Zhilin Wang, Yi Dong, Jiaqi Zeng, Virginia Adams, Makesh Narsimhan Sreedhar, Daniel Egert, Olivier Delalleau, Jane Scowcroft, Neel Kant, Aidan Swope, and Oleksii Kuchaiev. 2024g. HelpSteer: Multi-attribute helpfulness dataset for SteerLM. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3371–3384, Mexico City, Mexico. Association for Computational Linguistics.

Zhilin Wang, Yi Dong, Jiaqi Zeng, Virginia Adams, Makesh Narsimhan Sreedhar, Daniel Egert, Olivier Delalleau, Jane Polak Scowcroft, Neel Kant, Aidan Swope, and Oleksii Kuchaiev. 2023. Helpsteer: Multi-attribute helpfulness dataset for steerlm.

Linzhi Wu, Pengjun Xie, Jie Zhou, Meishan Zhang, Chunping Ma, Guangwei Xu, and Min Zhang. 2022. Robust self-augmentation for named entity recognition with meta reweighting. *arXiv preprint arXiv:2204.11406*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *arXiv preprint arXiv:2406.08464*.

Runsheng Yu, Yong Wang, Xiaoqi Jiao, Youzhi Zhang, and James T Kwok. 2024. Direct alignment of language models via quality-aware self-refinement. *arXiv preprint arXiv:2405.21040*.

Jizhi Zhang, Keqin Bao, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023a. Is chatgpt fair for recommendation. *Evaluating Fairness in Large Language Model Recommendation*.

Rongzhi Zhang, Jiaming Shen, Tianqi Liu, Jialu Liu, Michael Bendersky, Marc Najork, and Chao Zhang. 2024a. Knowledge distillation with perturbed loss: From a vanilla teacher to a proxy teacher. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4278–4289.

Rongzhi Zhang, Jiaming Shen, Tianqi Liu, Haorui Wang, Zhen Qin, Feng Han, Jialu Liu, Simon Baumgartner, Michael Bendersky, and Chao Zhang. 2024b. Plad: Preference-based large language model distillation with pseudo-preference pairs. *arXiv preprint arXiv:2406.02886*.

Rongzhi Zhang, Yue Yu, Jiaming Shen, Xiquan Cui, and Chao Zhang. 2023b. Local boosting for weakly-supervised learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3364–3375.

Zizhao Zhang and Tomas Pfister. 2021. Learning fast sample re-weighting without reward data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 725–734.

Zizhao Zhang, Han Zhang, Sercan O Arik, Honglak Lee, and Tomas Pfister. 2020. Distilling effective supervision from severe label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9294–9303.

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

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

Banghua Zhu, Michael Jordan, and Jiantao Jiao. 2023. Principled reinforcement learning with human feedback from pairwise or k-wise comparisons. In *International Conference on Machine Learning*, pages 43037–43067. PMLR.

# Appendix

## A Dataset Statistics and Preprocessing

We evaluate our method on multiple preference datasets, each with distinct characteristics. Table 3 summarizes the dataset sizes and key attributes.

Table 3: Summary of the preference datasets used in our experiments.

| Dataset | Number of Attributes | Train Set Size | Validation Set Size | Annotation Type | Primary Use |
|---|---|---|---|---|---|
| HelpSteer2 (Wang et al., 2024f) | 5 | 20k | {0.7k, 1k, 2k} | Human-annotated | General alignment |
| OpenAssistant2 (Köpf et al., 2024) | 4 | {5k, 20k} | {0.7k, 1k, 2k} | Human-annotated | Assistant-style conversations |
| Magpie-QWen-2.5 (Xu et al., 2024) | 1 | {10k, 32k} | {0.7k, 1k, 2k} | Synthetic | Fine-tuning |
| OffsetBias (Park et al., 2024) | 1 | {10k, 16k} | {0.7k, 1k, 2k} | Human-annotated | Bias mitigation |
| WildGuard (Han et al., 2024) | 1 | {5k, 12k} | {0.7k, 1k, 2k} | Adversarial | Safety evaluation |

**Normalization of Rating Scales.** These datasets use varying rating scales, which can introduce inconsistencies in preference modeling. For instance, HelpSteer2 employs a scale of 0-4, whereas other datasets use binary or continuous scores in different ranges. To standardize these ratings, we apply a linear transformation to convert all scores to a common 0-4 scale. Binary labels are mapped such that the preferred response corresponds to 4 and the non-preferred to 0.

**Merging Reward Attributes.** Some datasets contain overlapping but non-identical preference objectives. Since these datasets follow different annotation rubrics and evaluation protocols, we adopt the approach of (Wang et al., 2024c) and treat such objectives separately to maintain consistency. This ensures that preference signals remain distinct, preventing biases introduced by differing annotation standards. As a result, our reward model incorporates a 12-attribute head.

These preprocessing steps ensure that our dataset is well-structured for reward modeling and policy optimization, reducing the impact of inconsistencies across different sources.

## B Training Recipe Comparison

Table 4 presents the training data sizes of baseline models in our main experiments. `DORM` achieves strong performance on RewardBench while using only 50k training data (with 5k validation set), which is $10\times$ less than *ArmoRM-Llama3-8B-v0.1* and *Pair-preference-model-LLama3-8B*, and $40\times$ less than *Internlm2-20B-reward*. Despite the significantly smaller dataset size, our model attains a RewardBench overall score of 90.5, outperforming models trained on substantially larger datasets.

This data efficiency stems from our quality-aware preference weighting strategy and dynamic weighting in reward modeling. By prioritizing informative and reliable signals, our method reduces reliance on large-scale preference data. Additionally, bilevel optimization ensures effective use of high-value samples, mitigating performance degradation from noise or redundancy. These techniques allow our model to achieve strong performance with significantly fewer training samples.

## C Implementation Details

**Reward Model Training.** Our reward model is initialized with Mistral NeMo 12B-Instruct (NVIDIA and Mistral AI) and fine-tuned using a 12-attribute head. Table 5 outlines the hyperparameters used.

**Bilevel Optimization.** We initialize the proxy $u_0$ from $\theta_0$, and initialize $w_0$ being uniform. The learning rate is$\eta_\theta = \eta_u = 3e^-6, \eta_w = 0.001$ The coefficients are set to $\alpha = 10.0, \beta = 1.0$. We use Adam (Kingma, 2014) as the optimizer for both levels.

The training is conducted on two nodes (16 A100-80G GPUs in total), and it takes about 10.4 hours for 3-epoch training. During inference, we implement a Bayesian hyperparameter search to

| Model | Number of Training Data | RewardBench Overall Score |
|---|---|---|
| Llama3-70B-SteerLM-RM (Wang et al., 2024f) | 20k | 88.8 |
| URM-Llama3-8B (Lou et al., 2024) | 20k | 89.9 |
| ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024c) † | 585.4k | 90.4 |
| Pair-preference-model-LLaMA3-8B (Dong et al., 2024) † | 585.4k | 87.1 |
| Internlm2-20B-reward (Cai et al., 2024) † | 2.4 million | 90.2 |
| DORM | 50k | 90.5 |

Table 4: Models and their corresponding training dataset sizes.

Table 5: Training hyperparameters for reward model fine-tuning.

| Hyperparameter | Value | Notes |
|---|---|---|
| Reward Head Dimension | 12 | Correspond to Number of Reward Attributes |
| Batch Size | 4 | Per GPU |
| Learning Rate | $3 \times 10^{-6}$ | Cosine decay |
| Weight Decay | 0.01 | Applied to Adam optimizer |
| Betas | 0.9, 0.98 | Applied to Adam optimizer |
| Warm-up Steps | 200 | Cosine scheduler |
| Mixed Precision | BF16 | Reduces memory consumption |
| Number of Epochs | 3 | Maximum Training Epochs |

determine the projection vector $\lambda$ on the validation set. This vector is used to compute a weighted sum from the multi-attribute outputs of our reward model, enabling evaluation against the binary labels of RewardBench.

**Baseline Selection Criteria.** To ensure a fair and reproducible comparison on RewardBench, we selected baseline models based on the following principles:

- **Contamination-aware selection:** Some models on RewardBench were reported to involve data contamination that their training data may overlap with prompts present in the RewardBench evaluation set. We restrict our comparisons to models with no known data contamination on RewardBench, in order to avoid inflated performance due to train/test overlap.
- **Public availability and reproducibility:** We prioritize openly accessible reward models with released checkpoints or official links on the RewardBench leaderboard. Several top-ranked models lack publicly available implementations, making them impossible to reproduce or evaluate consistently.
- **Model size fairness:** Our method uses a 12B backbone. To ensure fair comparison, we exclude models with significantly larger parameter counts, where improvements may be largely attributed to model scale rather than training or data strategies.

We acknowledge that the field evolves rapidly. A new *decontaminated dataset*, Skywork-Reward-Preference-80K-v0.2 (Liu et al., 2024a), was recently released, enabling the training of stronger models such as `nicolinho/QRM-Llama3.1-8B-v2` and `Skywork/Skywork-Reward-Llama-3.1-8B-v0.2`, both achieving 93.1 on RewardBench. We note that Skywork's improvements stem from carefully curated data pipelines—e.g., filtering high-quality subsets and subsampling based on RM scores (as in ArmoRM), as documented by their authors. In contrast, our method automates preference weighting through bilevel optimization over heterogeneous sources. These approaches are orthogonal and potentially complementary:

our method could further benefit from applying its reweighting mechanism to cleaner datasets such as Skywork's.

## D Results with Different Data Size

We investigate the impact of the training/validation set size on model performance. Table 6 summarizes the results.

Table 6: Effect of validation set size on model performance.

| Training Size | Validation Size | Chat | Chat Hard | Reasoning | Safety | Overall |
|---|---|---|---|---|---|---|
| 50k | 3.5k | 94.7 | 84.8 | 88.5 | 92.9 | 90.2 |
| 50k | 5k | 95.6 | 83.4 | 89.8 | 93.1 | 90.5 |
| 50k | 10k | 96.8 | 84.9 | 89.1 | 95.3 | 91.5 |
| 100k | 3.5k | 96.9 | 85.2 | 88.7 | 95.1 | 91.5 |
| 100k | 5k | 96.8 | 85.5 | 89.2 | 96.0 | 91.9 |
| 100k | 10k | 97.1 | 85.6 | 90.6 | 94.9 | 92.1 |

**Larger validation sets improve model performance.** As the validation set size increases from 3.5k to 10k, the overall performance improves consistently across all training configurations. For instance, with 50k training samples, expanding the validation set from 3.5k to 10k leads to an increase in the overall score from 90.2 to 91.5, with notable gains in safety (92.9 to 95.3). This suggests that a larger validation set provides a more reliable signal for optimizing data weights, contributing to better model performance.

**Increased training data enhances robustness.** Expanding the training set from 50k to 100k also yields consistent improvements in overall performance, particularly in safety and reasoning. With a 10k validation set, the overall score increases from 91.5 (50k training) to 92.1 (100k training), while safety improves from 95.3 to 94.9. These results indicate that additional training data strengthens the model's ability to generalize across different tasks.

## E Training Overhead and Cost-Effectiveness of DORM

One common concern with bilevel optimization methods is the additional computational overhead due to extra steps and models. We clarify and quantify this overhead both theoretically and empirically.

**Theoretical Overhead.** The main overhead of DORM comes from maintaining an auxiliary model $u$ (with the same architecture as the primary model $\theta$) and computing additional gradients on the validation set. Specifically, $u$ is a transient proxy that is updated only once per iteration (Algorithm 1) to approximate the lower-level optimum for hypergradient calculation. By employing a first-order hypergradient approximation method (Kwon et al., 2023b; Lu and Mei, 2024), we avoid the need to compute expensive second-order Hessians required by traditional bilevel optimization approaches.

**Empirical Cost.** Training our 12B reward model with DORM for 3 epochs takes approximately 10.4 hours on 16 A100 GPUs. This time includes all components of the bilevel optimization—updates for $u$, $\theta$, and $w$, and computations on both training set $D_{\text{tr}}$ and validation set $D_{\text{val}}$. This demonstrates that DORM is practically feasible even at scale.

**Cost-Benefit Tradeoff.** To justify this overhead, we compare DORM against standard baselines in terms of training time and downstream performance. As shown in Table 7, even when the baselines are given the validation data for training, DORM achieves significantly higher RewardBench scores with only moderate overhead.

Table 7: Training overhead and performance of different optimization strategies.

| Method | Training Data | Optimization | Time / Epoch | RewardBench Score |
|---|---|---|---|---|
| SFT | 60k (train + val) | Standard | ∼2.4 hrs | 85.6 |
| Prior weights only | 60k (train + val) | Standard | ∼2.4 hrs | 86.7 |
| Bilevel w/o prior | 50k train, 10k val | Bilevel | ∼3.5 hrs | 89.4 |
| DORM | 50k train, 10k val | Bilevel | ∼3.5 hrs | **91.5** |

These results demonstrate that DORM not only improves performance over strong baselines, but also makes efficient use of validation data through bilevel training, justifying its moderate computational overhead.

## F Study on Uncertainty Measurement

In 3.1, we use MC dropout for epistemic uncertainty estimation. Our method is compatible with other uncertainty estimation techniques. We include sigmoid-based and ensemble-based methods for comparison and illustrate our rational for using MC dropout.

Table 8: Ablation study results on uncertainty-based weighting.

| Method | Chat | Chat Hard | Reasoning | Safety | Overall |
|---|---|---|---|---|---|
| Baseline | 90.7 | 72.8 | 86.7 | 91.1 | 85.3 |
| MC Dropout | 91.9 | 77.0 | 85.2 | 91.6 | 86.4 |
| Ensemble | 87.7 | 74.6 | 89.8 | 92.5 | 86.1 |
| Sigmoid | 86.8 | 72.8 | 89.2 | 91.9 | 85.2 |

MC dropout achieves the highest overall score of $86.4$, outperforming the baseline $85.3$ and other uncertainty-estimation methods. It provides balanced improvements across all tasks, particularly in chat hard $+4.2$ and chat $+1.2$. While ensemble-based estimation achieves slightly better reasoning and safety scores, its overall performance remains lower.

MC dropout is also computationally efficient, requiring only stochastic forward passes within a single model, whereas ensemble methods require multiple model instances. Although the sigmoid-based approach is computationally lightweight, it does not provide sufficient performance gains.

In summary, our method supports various uncertainty estimation techniques. MC dropout is preferred due to its strong empirical performance and efficiency.

### F.1 Why Epistemic Uncertainty and Disagreements Rather Than Aleatory Uncertainty?

In our quality estimation for the preference data, we consider uncertainty to identify which data points are more informative for the reward model. There are two common forms of uncertainty: epistemic and aleatory (Hüllermeier and Waegeman, 2021). Epistemic uncertainty arises from the model's lack of knowledge about the underlying preference function, while aleatory uncertainty is related to inherent randomness or noise in the data generation process. Our quality estimation focuses on epistemic uncertainty and disagreements rather than aleatory uncertainty for the following reasons:

1. **Epistemic uncertainty is reducible through data or model improvements:** Epistemic uncertainty reflects gaps in the model's current representation. If a data point causes the model to be uncertain, it suggests the model does not fully understand the underlying preference structure at that point. Prioritizing such data helps refine model parameters. In contrast, aleatory uncertainty stems from inherent randomness and cannot be mitigated through training.

2. **Disagreements highlight label reliability issues:** Discrepancies between model predictions and labels indicate potential label noise or annotation errors. These disagreements help us identify data that may be mislabeled or difficult for the model. Adjusting data weights based on these disagreements allows the model to focus on data that can provide more reliable learning signals.

3. **Aleatory uncertainty does not provide actionable insights for weighting:** Since aleatory uncertainty persists regardless of training, incorporating it into data weighting does not improve learning. Epistemic uncertainty and disagreements, however, offer actionable insights for data selection and model refinement.

For these reasons, our quality estimation leverages epistemic uncertainty and disagreements to guide the model toward more informative data, improving training efficiency and alignment.

## G Robustness to Noisy Preference Labels

We conduct additional experiments to evaluate the robustness of DORM to noisy preference labels, and compare it against standard SFT under controlled label corruption. We use a LLaMA-3.2-3B-IT reward model and progressively introduce noise into the training set by randomly flipping chosen/rejected labels. For a fair comparison, DORM uses 50k noisy training samples and 10k clean validation samples, while SFT uses the combined 60k data as training data (i.e., no validation set). Evaluation is done across four RewardBench categories and aggregated using the standard average.

As shown in Table 9, DORM consistently outperforms SFT across all noise levels. The performance gap becomes more pronounced as noise increases, indicating that DORM's dynamic reweighting mechanism makes it more robust to noisy supervision. In contrast, SFT lacks such robustness and treats clean and corrupted samples uniformly.

Table 9: Performance under different amounts of training label noise.

| Method | # Noisy Samples | Chat | Chat Hard | Safety | Reasoning | Overall |
|---|---|---|---|---|---|---|
| DORM | 0 | 84.7 | 43.4 | 80.2 | 69.6 | **69.5 (+2.7)** |
| SFT | 0 | 82.0 | 41.1 | 77.6 | 66.3 | 66.8 |
| DORM | 5k | 81.9 | 39.8 | 78.9 | 68.5 | **67.3 (+3.2)** |
| SFT | 5k | 82.4 | 37.9 | 72.4 | 63.5 | 64.1 |
| DORM | 10k | 78.8 | 39.0 | 75.7 | 68.8 | **65.6 (+3.6)** |
| SFT | 10k | 79.6 | 37.2 | 69.5 | 61.7 | 62.0 |

These results show that DORM is more robust to label noise due to its dynamic weighting over training samples. This advantage becomes more significant under higher noise levels, highlighting the value of bilevel optimization for real-world, imperfect preference data.

## H Distinguish Hard and Noisy Samples

A key question in our weighting approach is how to distinguish hard samples (truly ambiguous or difficult) from noisy ones (incorrectly labeled), especially in cases where both the uncertainty $u$ and the disagreement $q$ are high. Our method addresses this challenge in two stages:

**1. Initial Prior Weighting ($w^0$).** Theoretically, both $u$ (uncertainty) and $q$ (disagreement with the ground-truth label) can be high for either hard or noisy samples, but their statistical profiles

often differ:

- *Hard samples*, such as those near decision boundaries or with genuinely ambiguous preferences, tend to induce high model uncertainty $u$ but only moderate disagreement $q$, especially if the model oscillates around the true label.
- *Noisy samples*, *i.e.*, mislabeled data, may show low or moderate $u$ (as the model is confident in its incorrect prediction), but exhibit high $q$ with the correct label.

To exploit this distinction, we use a heuristic prior weighting function:

$$w^0 = \exp(u - \gamma q),$$

which favors high-uncertainty but low-disagreement samples and penalizes confident-but-wrong predictions. The parameter $\gamma$ controls the trade-off, providing intermediate weights when both $u$ and $q$ are high.

**2. Bilevel Optimization Refinement.** Beyond the initial heuristic, our method employs a second-stage bilevel optimization to refine weights using validation loss. This allows the model to learn from data which samples are genuinely informative versus detrimental, even when the prior signal is ambiguous.

By leveraging validation performance as a signal, the model adjusts $w$ to down-weight noisy or misleading samples and to emphasize helpful hard cases, surpassing the expressiveness of the initial $u/q$-based heuristic.

Table 10: Statistical comparison between hard and noisy samples.

| **Sample Type** | **# Samples** | **Avg.** $u$ **(± std)** | **Avg.** $q$ **(± std)** |
|---|---|---|---|
| Hard samples | 1,000 | 0.23 ± 0.08 | 0.34 ± 0.11 |
| Noisy samples | 1,000 | 0.14 ± 0.05 | 0.62 ± 0.17 |

**Empirical Validation.** To verify the distinguishable characteristics of hard vs. noisy samples under our weighting scheme, we conduct an analysis using 10k validation samples from the HelpSteer2 dataset. We construct two subsets:

- **1k Hard samples**: Selected where human-assigned preference scores for chosen vs. rejected candidates are close (within 1.0), indicating human-level ambiguity or disagreement.
- **1k Noisy samples**: Generated by injecting Gaussian noise $\epsilon \sim \mathcal{N}(0, 1)$ into scalar human scores (range [0, 5]), and clipping results to stay within bounds.

We compute the average $u$ and $q$ for each category, as shown in Table 10. The results confirms that our prior weighting rule $w^0 = \exp(u - \gamma q)$ tends to favor truly ambiguous but trustworthy samples (high $u$, low $q$), while de-emphasizing potentially mislabeled ones (low $u$, high $q$). While imperfect, this initialization provides a useful inductive bias that is further refined through bilevel optimization.