From Lists to Emojis: How Format Bias Affects Model Alignment

Anonymous ACL submission

Abstract

In this paper, we study format biases in reinforcement learning from human feedback (RLHF). We observe that many widely-used preference models—including human evaluators, GPT-4, and top-ranking models on the RewardBench benchmark—exhibit strong biases towards specific format patterns, such as lists, links, bold text, and emojis. Furthermore, large language models (LLMs) can exploit these biases to achieve higher rankings on popular benchmarks like AlpacaEval and LMSYS Chatbot Arena.

One notable example is verbosity bias, where current preference models favor longer responses that appear more comprehensive, even when their quality is equal to or lower than shorter responses. However, format biases beyond verbosity remain largely underexplored. In this work, we extend the study of biases in preference learning beyond the commonly recognized length bias, offering a comprehensive analysis of a wider range of format biases. Additionally, we show that with a small amount of biased data (less than 1%), we can inject significant bias into the reward model. Moreover, these format biases can also be easily exploited by downstream alignment algorithms, such as `best-of-n sampling` and online iterative `DPO`, as it is usually easier to manipulate the format than to improve the quality of responses. Our findings emphasize the need to disentangle format and content both for designing alignment algorithms and evaluating models.

1 Introduction

Reinforcement Learning from Human Feedback (RLHF) (Ziegler et al., 2019; Ouyang et al., 2022; Bai et al., 2022) has become a critical technique in the LLM training pipeline to align the outputs with human preferences. This approach has contributed to the tremendous successes of various state-of-the-art LLMs, such as Gemini (Team et al., 2023), Chat-GPT (OpenAI, 2023), and Claude (Anthropic, 2023), by making the generated responses helpful, harmless, honest and controllable (Ouyang et al., 2022; Bai et al., 2022). Despite its effectiveness, RLHF also faces challenges. One notable issue is that policies optimized via RLHF tend to develop biases, leading them to "game" the reward model (RM) *i.e.*, the policy receives a high reward from RMs but does not achieve intended objectives. This behavior often results in performance degradation (Bai et al., 2022; OpenAI, 2023), a phenomenon widely referred to as reward hackings (Denison et al., 2024; Ramé et al., 2024).

One prevalent form of reward hacking is the verbosity issue (Chen et al., 2024; Liu, 2024; Singhal et al., 2023), where the policy generates unnecessarily lengthy responses in an attempt to appear more helpful or comprehensive. However, many other types of bias patterns beyond the verbosity remain largely under-explored, especially in the current most popular instruction-following (IF) benchmarks. In this study, we are dedicated to answering the following question:

On the current popular IF benchmarks, do the scores truly reflect models' capabilities?

We first summarize the response statistics generated by the top-ranking models on the AlpacaEval Leaderboard (Li et al., 2023). The results are presented in Table 7. We observe that all the models **strongly prefer** generated responses that include bold text, lists, and exclamation patterns. Additionally, the Gemma models (Team et al., 2024; Zhou et al., 2024) tend to incorporate more emojis in their responses compared to Llama (Meta, 2024) and Mistral (Jiang et al., 2023; Liu et al., 2024a). Although the proportion of responses containing links is less significant, we notice that AlpacaEval's judge, GPT-4, tends to favor responses with links when they are present. Due to space constraints, a more comprehensive study of these biases in exist-

List Example

Instruction: Enter the words that satisfy the given condition.

GPT4-1106-preview (preferred):
Sure, here are five countries that start with the letter 'S':

1. Spain
2. Sweden
3. Switzerland
4. Syria
5. Serbia

GPT4-0613-concise (Unpreferred):
Spain, Sweden, Switzerland, Singapore, Senegal

ing preference models, are deferred to Section 2.

The underlying cause of these reward hacking phenomena lies in the imperfections of the reward models used in RLHF. These models often suffer from approximation errors and lack sufficient representativeness due to the limited diversity and number of human labelers. For instance, in a typical RLHF setup, the target preference is that of real-world users, yet we rely on a small group of human labelers to annotate the responses generated by LLMs. During the policy improvement stage, a proxy reward model, constructed from these annotated responses, is ultimately used to provide the learning signal. As a result, achieving a higher proxy reward does not always correlate with better performance when evaluated by real-world users. Even though one believes that some format biases are intrinsic to human preference, they can still cause problems in downstream alignment tasks if not explicitly addressed through regularization or specialized algorithm designs. This is because the most popular RLHF algorithms such as RAFT (Dong et al., 2023), online iterative DPO (Rafailov et al., 2023; Xiong et al.), and PPO (Schulman et al., 2017) run in an on-policy (LLMs learn from the self-generated responses) and online manner (the reward continuously labels the self-generated responses throughout the training process). These format biases are much easier to exploit by the LLMs in pursuit of a higher reward compared to improving the content quality. Consequently, these biases can be significantly amplified in the online and on-policy training pipeline.

In this work, we extend the study of biases in preference learning beyond the commonly explored length bias, presenting a comprehensive analysis of a broader class of pattern biases. Our findings emphasize the importance of disentangling format and content in both the design of alignment algorithms and model evaluations. We summarize our contributions as follows.

1. **Format biases widely exist in human, GPT-4, and open-source preference models.** We identify several common format biases and demonstrate that both human evaluators, GPT-4, and a range of open-source preference models exhibit strong biases toward these formats.
2. **Existing high-ranking models on the public leaderboard exhibit strong format biases.** We show that many high-ranking models on public leaderboards like AlpcaEval (Li et al., 2023) and MT-bench (Zheng et al., 2023), tend to exploit evaluator biases by generating responses with specific biased formats.
3. **Reward modeling can be easily attacked by a small amount of biased data (less than 1%) and leads to significant format biases in downstream alignment tasks.** We show that introducing less than 1% biased data into a debiased preference dataset can cause the resulting reward models to develop substantial format biases. These biases are easily exploited by policy models during downstream alignment, especially in online algorithms like iterative DPO (Rafailov et al., 2023; Xiong et al.) and PPO (Schulman et al., 2017).
4. **Passive data filtering is not sufficient for reward model debiasing.** We find that passive data filtering is inadequate, as it removes a substantial portion of the training data and leads to inferior model capacity. We also explores a two-head reward construction approach to debias reward models and addressed the signal sparsity issue for specific formats.

1.1 Preliminaries

Preference data. In the pipeline of RLHF, we start with a human preference dataset $\mathcal{D} = \{x, a^w, a^l\}$. Here x is a prompt, and a^w and a^l are two responses, where a^w is preferred over a^l according to some human labeler.

Bradley-Terry model (Bradley and Terry, 1952) and RLHF. To simplify the problem and connect

the human rating with reinforcement learning, it is widely assumed that there exists a reward function $r^\star(x, a) \in \mathbb{R}$, so that

$$\mathbb{P}(a^1 \succ a^2 | x, a^1, a^2) = \frac{\exp(r^\star(x, a^1))}{\exp(r^\star(x, a^1)) + \exp(r^\star(x, a^2))}, \tag{1}$$

Then, the goal of RLHF is to optimize the following KL-regularized target:

$$J(\pi) = \mathbb{E}_{x\sim d_0}[\mathbb{E}_{a\sim\pi(\cdot|x)}[r^\star(x, a)] - \eta D_{\mathrm{KL}}(\pi(\cdot|x)\|\pi_0(\cdot|x))], \tag{2}$$

where $\eta > 0$ is the KL penalty coefficient. However, the ground truth $r^\star$ is not directly available. Under the BT assumption, a proxy reward model r is trained by maximizing the following log-likelihood function:

$$\ell_{\mathcal{D}}(\theta) = \sum_{(x,a^w,a^l)\in\mathcal{D}} \log\Big(\sigma\big(r_\theta(x, a^w) - r_\theta(x, a^l)\big)\Big), \tag{3}$$

where $\sigma(z) = 1/(1 + \exp(-z))$ is the sigmoid function. The reward model r is then used to provide a learning signal in the subsequent RL stage, typically with the `PPO` algorithm (Schulman et al., 2017; Christiano et al., 2017).

Direct Preference Optimization (`DPO`) (Rafailov et al., 2023). While the DRL-based RLHF framework has been successful with models like InstructGPT (Ouyang et al., 2022) and Claude (Anthropic, 2023), implementing `PPO`, particularly in the context of LLMs, presents significant challenges. These challenges stem from the complexity, instability, and inefficiency of RL methods compared to supervised learning (Choshen et al., 2019). In recognition of this, a line of works has proposed direct alignment algorithms, such as `Slic` (Zhao et al., 2023), `DPO` (Rafailov et al., 2023), and `IPO` (Azar et al., 2023). These algorithms directly optimize a supervised target on the preference dataset. We use `DPO` as a representative example in our study, whose loss function is given by:

$$\mathcal{L}_{\texttt{DPO}}(\theta, \pi_0) = -\sum_{(x,a^w,a^l)\in\mathcal{D}} \Big[\log\sigma\Big(\eta\log\frac{\pi_\theta(a^w|x)}{\pi_0(a^w|x)} - \eta\log\frac{\pi_\theta(a^l|x)}{\pi_0(a^l|x)}\Big)\Big]. \tag{4}$$

2 The Pattern Bias in Preference Learning

In this section, we summarize some representative format biases and show that these biases widely exist in human, GPT-4, and other open-source preference models.

2.1 Pattern Statistics in Preference Datasets and Benchmarks

Methods and datasets. To identify the format biases existing in the preference models, we select several representative preference datasets with different prompt distributions, response generators, and preference models.

- **RLHFlow-Preference-700K**[1] (Dong et al., 2024) consists of 700K preference pairs and has been used to train a series of strong open-source reward models or pairwise preference models. This dataset is a cleaned version of many open-source preference datasets, including HH-RLHF (Bai et al., 2022), SHP (Ethayarajh et al., 2022), HelpSteer (Wang et al., 2023), UltraFeedback (Cui et al., 2023), UltraInteract (Yuan et al., 2024), Distilable-Capybara, PKU-SafeRLHF (Ji et al., 2024), and Distilabel-Orca (Lian et al., 2023).

- **LMSYS-Arena-55K** (Chiang et al., 2024), which consists of 55K real-world conversations between users and > 70 state-of-the-art LLMs, as well as the preference signals from the users. This dataset is prepared for the Kaggle competition of predicting human preference on Chatbot Arena battles.

- **AlpacaEval** (Li et al., 2023) is a benchmark to test models' IF capabilities with 805 test prompts. The generated responses from the model are compared to the responses from GPT-4, where the GPT-4 is also used as the judge. We collect the data from 258 tested LLMs to form this dataset.

- **UltraFeedback** (Cui et al., 2023) consists of 64k prompts from diverse resources and the authors generate 4 responses per prompt using 4 different LLMs. The preference is labeled by the LLM-as-a-judge with GPT-4, based on a fine-grained annotation instruction. Specifically, the instruction contains 4 different aspects, including instruction-following, truthfulness, honesty, and helpfulness.

For all datasets, samples are processed into the standard format (x, a^w, a^l), where a^w is preferred over a^l according to the preference model. We would like to compare the proportions of samples with

[1] https://huggingface.co/datasets/hendrydong/preference_700K

Model	Type	Win-Rate (%)					
		Bold	List	Emoji	Exclamation	Link	Affirmative
GPT-4 Turbo	LLM-as-a-Judge	89.5	75.75	86.75	80.5	87.25	88.75
ArmoRM-Llama3-8B-v0.1	Multi-head RM	98	50.5	55	34.5	27	28.5
Pairwise-model-Llama-3-8B	Pairwise PM	97	93.5	70.5	64.25	84.75	47.75
FsfairX-Llama-3-8B-v0.1	BT RM	95.5	68.5	15	28.5	64.5	59.5
Skywork-Critic-Llama-3.1-8B	Generative Model	99.25	88.75	97.25	77.75	75	85
Zephyr-Beta-Mistral-7B	DPO Model	37.5	50	26.5	72	58	21
OffsetBias-RM-Llama-3-8B	BT RM	77.5	84	28	38	62	30.5

Table 1: Evaluation results of the pattern bias of five popular preference models. This table displays the adjusted win rates, accounting for tied conditions by allocating half to the win rate. Detailed proportions of wins, ties, and losses for each model are presented in Table 5. For the DPO model Zephyr-Beta-Mistral-7B, the reward is computed as $\log \frac{\pi(a|x)}{\pi_0(a|x)}$, where π represents the model itself and π_0 represents its base model Zephyr-Beta-Mistral-7B.

certain formats in both the preferred and unpreferred responses. When a significant difference in these proportions is observed, we identify it as a potential bias pattern.

Patterns. We identify seven distinct patterns within responses: length, emoji, bold, exclamation, list, link, and affirmative. Examples of emojis and list patterns were shown in the introduction. The examples of the remaining patterns are shown in Appendix C.

Results. We summarize the main statistics in Table 7 and outline the main findings as follows:

1. Both GPT-4 and humans exhibit a preference for longer sentences, bold, lists, exclamation marks, and an affirmative tone. However, GPT-4's preference for these elements is typically stronger than that of humans.
2. GPT-4 also has a preference for emojis and hyperlinks, which humans do not share.
3. The biases in the UltraFeedback dataset are less pronounced. This may be because UltraFeedback explicitly assesses context-related qualities such as helpfulness, instruction-following, honesty, and truthfulness—factors that are less directly related to format and determine the final preference label by the average of these scores. In contrast, other datasets rely on general preference judgments without first asking for the fine-grained scores.

2.2 Bias Evaluation of Preference Model

In this subsection, we create a collection of evaluation datasets to examine the format biases of GPT-4 and open-source preference models. Specifically, we include representative open-source BT reward model (Bradley and Terry, 1952; Dong et al., 2024), pairwise preference model (Zhao et al., 2023; Liu et al., 2023; Ye et al., 2024), multi-head reward model (Wang et al., 2024b), generative critic model (Shiwen et al., 2024) and the reward model of DPO model (Tunstall et al., 2023) in our experiments.

Evaluation dataset for specific patterns. We generate responses using prompts from the LMSYS-Arena dataset (Chiang et al., 2024). For each prompt, we generate responses from various models, including Meta-Llama-3-8B-Instruct, Meta-Llama-3.1-8B-Instruct, Mistral-7B-Instruct-v0.2, Gemma-2-9B-It, Qwen1.5-7B-Chat, Qwen2-7B-Instruct, and ChatGLM3-6B. We then select responses that exhibit the specified pattern and remove this pattern in the responses to obtain preference pairs containing two responses that are identical except for this pattern. For each pattern, we generate 200 pairs for evaluation.

Evaluation results. For each preference model, we first evaluate the responses with and without the pattern collected previously. We then compute the win rate of responses with the pattern against those without it. For an unbiased reward model, the win rate should be close to 50%, indicating that the model judges two responses, which are identical except for the pattern, as equally favorable. The results are summarized in Table 1, with key observations outlined below:

1. GPT-4 displays strong bias across all patterns, which aligns with our findings of pattern bias in the GPT-4 annotated preference datasets in Table 7.
2. Pairwise-model-Llama-3-8B and Skywork-Critic-Llama-3.1-8B also exhibit bias across all patterns.
3. The FsfairX-Llama-3-8B-v0.1 model, trained on the RLHFlow-Preference dataset, shows bias in all patterns except for emojis and exclamations, consistent with the bias statistics results of its training dataset highlighted in Table 7.

4. ArmoRM-Llama3-8B-v0.1, which explicitly considers multiple context-related attributes when evaluating responses, reduces format bias by focusing on aspects less directly related to formatting. However, ArmoRM still shows a strong bias toward the bold pattern.
5. The Zephyr-Beta-Mistral-7B model, trained on the Ultrafeedback_binarized dataset, only shows bias in exclamation and link patterns, consistent with the bias statistics results of its training dataset highlighted in Table 7.
6. In particular, we notice that although Park et al. (2024a) leverages debiased data to mitigate several implicit biases in the reward model FsfairX-Llama-3-8B-v0.1 and obtain a new debiased reward model OffsetBias-RM-Llama-3-8B, the debiased model still shows considerable bias toward bold and list patterns because they do not explicitly consider the format bias. Moreover, the bias in list format becomes more severe after finetuning.

3 Pattern Bias Transfer in Downstream Reward Modeling and Alignment

In Section 2, we show that the format biases widely exist in the existing preference model. We now conduct controllable experiments to study how these biases transfer from preference data to the reward model, and further to the downstream RLHF-aligned model. For simplicity, we focus on the bold pattern and list pattern.

3.1 A Small Amount of Data Can Lead to Significant Reward Model Bias

Base dataset. We first randomly sample two responses for each prompt from the UltraFeedback dataset (Cui et al., 2023). Then, we remove the response pairs where there exists one response containing any pattern. This filtered dataset serves as our base dataset. Finally, we obtain a preference dataset with 71.6K pairs.

Attacking dataset. To inject bias into the base dataset, we generate responses following the process described in Section 2. We label responses with the target pattern as the preferred ones and designate augmented responses as the unpreferred ones for reward model training. When injecting both bold and list biases simultaneously, we use more data containing the list pattern, as we observe that the bold pattern bias tends to dominate if both patterns are injected in equal proportions.

Training Dataset	Win-rate (%) Bold	List
Baseline	57.5	51.0
Baseline + 0.14% Bold	61.0	-
Baseline + 0.35% Bold	66.0	-
Baseline + 0.70% Bold	88.0	-
Baseline + 0.14% List	-	71.5
Baseline + 0.35% List	-	74.0
Baseline + 0.70% List	-	77.5
Baseline + 1.40% List	-	79.5
Baseline + 0.70% Bold + 1.40% List	83.0	80.0

Table 2: The results of pattern biases in the reward modeling when we introduce biased data. The baseline dataset is a filtered version of UltraFeedback, where we delete all the samples containing the specific patterns. Then, we combine the baseline dataset with a small preference dataset where the preferred response contains the pattern while the unpreferred response does not.

Model, and parameter. We use the Llama-3-8B-it (AI@Meta, 2024) as our base model of the reward model and train on the mixture of the base dataset and the attacking dataset. We also use the evaluation datasets to evaluate the biases in the reward model. For all the experiments, we follow the hyper-parameter in Dong et al. (2023) to train the model for 1 epoch with a max length of 3096 tokens. We use a global batch size of 128 with deepspeed stage 3. We mainly search the hyper-parameter learning rate in {5e-7, 1e-6, 5e-6} and choose the best one (1e-6).

Results. We summarize the main results in Table 2. First, we observe that removing all training samples containing the specific pattern does not completely eliminate pattern bias as the reward model trained on the baseline dataset (a filtered version of UltraFeedback) still shows a significant bias toward the bold pattern. Additionally, even with a very small amount of biased data (less than 1%), we can inject bias into the final model. For example, the reward model trained on the baseline dataset achieves a 51% win rate for the list pattern, but when trained on a mixture of the baseline dataset and just 0.7% list-augmenting data, the model's preference for lists increases to 77.5%.

3.2 Downstream Alignment Task: DPO and PPO Training

In this section, we apply the obtained reward models in the downstream alignment tasks. Specifically, we consider the baseline reward model as a refer-

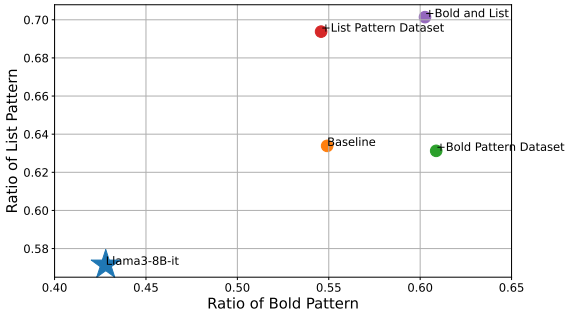

Figure 1: We use the Llama-3-8B-it model as the base model and use the prompts from UltraFeedback to conduct DPO training. While the base model already favors the bold and list patterns, the DPO training with the biased reward further amplifies these biases.

ence point, and the list-biased reward model (baseline dataset + 1.40% list-augmented data), bold-biased reward model (baseline dataset + 1.40% bold-augmented data), and the reward model that is attacked by both 0.70% bold-augmented data and 1.40% list-augmented data. We study three widely used algorithms, including the offline DPO (Rafailov et al., 2023), and its online variant Xiong et al.; Tajwar et al. (2024), as well as the online PPO (Schulman et al., 2017). In comparison, the online algorithms generate responses from the current policy model, ask an external preference model (the reward model in our case) to annotate the responses, and train on the resulting preference dataset. We also study the inference-time best-of-n (BoN) sampling (Nakano et al., 2021), whose results are deferred to Appendix D.1.

Training setup. We train the DPO in an on-policy way where we follow Dong et al. (2024); Pace et al. (2024); Hoang Tran (2024) to generate 5 responses per prompt and use the best and the worst responses to construct a pair. We train the DPO for 2 epochs with a global batch size of 128 and a learning rate of 5e-7. The maximal number of tokens is 3096 and the KL coefficient is 0.1. For PPO training, we mainly follow the recipe in Hu et al. (2024) to use a rollout batch size of 1024, a global training batch size of 128, and set the learning rates of actor and critic as 5e-7, and 9e-6, respectively. The initial KL coefficient is 0.01 and we train on the collected data for 1 epoch. We test the resulting policy models using the AlpacaEval (Li et al., 2023) prompt set and the test set split from the UltraFeedback dataset. Generations are conducted with a temperature of 1.0 and a maximum token limit of 2048.

Main results of offline DPO. In our first experiment, we initialize with the Llama-3-8B-it model and apply different reward models from the previous section to label 59K$\times$5 on-policy samples. The results are summarized in Figure 1. We observe that the base model, Llama-3-8B-it, already shows a strong preference for generating responses with both bold and list patterns. However, DPO further amplifies these biases, as all four new models shift toward the upper-right corner compared to the base model in the figure. Notably, the baseline reward model also contributes to this bias amplification, indicating that simply removing training data with the pattern in reward modeling is insufficient to fully address the issue. Additionally, the extent of bias amplification depends on the specific biases in the reward models. For example, the reward model trained with additional bold-pattern data results in a stronger bias toward bold patterns, while the one trained with both bold and list patterns amplifies both biases compared to the baseline reward model.

Main results of online iterative DPO and PPO. In addition to the offline DPO, we also study the behavior of online algorithms including the iterative DPO and PPO. We focus on the reward model trained with both bold-augmented and list-augmented data and select Llama-3-8B-SFT from Dong et al. (2024) as our base model because its biases are much lighter compared to Llama-3-8B-it, making it more suitable for illustrating the concept. To ensure a fair comparison, we fix the number of queries to the reward model for both online iterative and offline learning. Specifically, we split the UltraFeedback prompt set into three training sets, each containing 19K prompts, and a test set of 2K prompts. For offline learning, we use the entire training set at once, while in online iterative learning, we use the training sets iteratively. We also update the reference model at each iteration as done in Hoang Tran (2024); Xiong et al. (2024), which achieves better in-domain performance. We also monitor the training progress of our model using the widely adopted AlpacaEval2 benchmark (Li et al., 2023) to ensure that our experiments reflect typical practices in online iterative DPO and PPO training. See Appendix D.2 for details.

The results are summarized in Figure 2. We find that online iterative DPO significantly amplifies pattern biases compared to both the SFT model and the offline DPO baseline. Notably, the model after the first iteration, trained on 20K pairs, exhibits a similar level of bias as the offline DPO trained on 60K pairs. This indicates that increasing the training size does not necessarily lead to higher

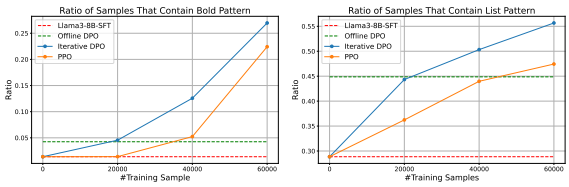

Figure 2: The test results on the AlpacaEval and split test set of UltraFeedback. We use the Llama-3-8B-SFT model as the base model and use the prompts from UltraFeedback to conduct iterative DPO and PPO training, where the reward model is used to annotate the on-policy samples from the model. The offline DPO is a special case where we use all the prompts in one iteration. The online iterative DPO and PPO significantly amplify the pattern bias throughout training.

bias. Similarly, the model trained by PPO also exhibits a stronger bias compared to the offline DPO method. This aligns with the fact that the online methods continuously explore the space and efficiently exploit the existing biases in the reward model, ultimately resulting in a higher level of bias.

4 Mitigate the Format Bias

In this section, we present an initial study of the approach to mitigate the format bias. We also explore the impact of format debias on the reward model performance as measured by the RewardBench (Lambert et al., 2024).

4.1 Method

Following Chen et al. (2024), to mitigate pattern bias in rewards, we train a two-head RM to predict two distinct rewards: an authentic reward $r^{\mathrm{A}}_{\boldsymbol{\theta}}(x, a)$ representing response quality, and a debiased reward $r^{\mathrm{D}}_{\boldsymbol{\theta}}(x, a)$ disentangled from specific patterns p. We assume that the preference satisfies the Bradley Terry structure where the reward is $r^{\mathrm{A}}_{\boldsymbol{\theta}}+r^{\mathrm{D}}_{\boldsymbol{\theta}}$. Then, the ranking loss for the reward modeling on a tuple (x, a^w, a^l) is defined as:

$$\begin{aligned}\mathcal{L}^{\mathrm{R}}_{\boldsymbol{\theta}}(x, a^w, a^l) = -\mathbb{E}[\log(\sigma(r^{\mathrm{A}}_{\boldsymbol{\theta}}(x, a^w) \\ + r^{\mathrm{D}}_{\boldsymbol{\theta}}(x, a^w) - r^{\mathrm{A}}_{\boldsymbol{\theta}}(x, a^l) - r^{\mathrm{D}}_{\boldsymbol{\theta}}(x, a^l)))].\end{aligned} \tag{5}$$

In addition to the ranking loss, we include a correlation loss to control the impact of specific format patterns p in the reward models. Compared to length bias, one major challenge is that a score reflecting a specific pattern is not directly available. In contrast, every response can be characterized by its length. To address this, we use a heuristic function $\#_p$ to evaluate responses based on specific patterns p, such as $\#_{\text{bold}}$ for the number of bold words and $\#_{\text{list}}$ for that of list heads. The constraint loss for pattern debiasing is formulated as:

$$\mathcal{L}^{\mathrm{C}}_{\boldsymbol{\theta}}(X, A) = |\rho(r^{\mathrm{A}}_{\boldsymbol{\theta}}(X, A), \#_p(Y))| - \rho(r^{\mathrm{D}}_{\boldsymbol{\theta}}(X, A), \#_p(Y)), \tag{6}$$

Rank	Models	Rank diff.
1	Claude3.5-sonnet	+2
2	Llama3.1-405B-it	-1
3	OpenPipe-MoA-GPT4-Turbo	+4
4	Claude3-opus	+1
5	Llama3-70B-it	-3
6	Llama3-it-8B-WPO	-2
7	GPT4-1106-Preview	+1
8	Llama3-8B-it	-2

Table 3: We rank the models using AlpacaEval2 prompts, with the rankings evaluated by our debiased reward model. The "ranking difference" represents the difference between the rankings produced by the standard reward model and those generated by the debiased reward model.

where (X, A) denotes a batch, and ρ represents the Pearson correlation function. We train the reward model by weighting $\mathcal{L}^{\mathrm{R}}_{\boldsymbol{\theta}}$ and $\mathcal{L}^{\mathrm{C}}_{\boldsymbol{\theta}}$ to minimize the following objective:

$$\begin{aligned}\sum_{(x,a^w,a^l)\in(X,A^w,A^l)} \mathcal{L}^{\mathrm{R}}_{\boldsymbol{\theta}}(x, a^w, a^l) + \lambda_C \mathcal{L}^{\mathrm{C}}_{\boldsymbol{\theta}}(X, A^w) \\ + \lambda_C \mathcal{L}^{\mathrm{C}}_{\boldsymbol{\theta}}(X, A^l),\end{aligned} \tag{7}$$

where (X, A^w, A^l) is a batch, and $\lambda_C > 0$ is a constant regulating pattern bias. For debiasing evaluation, we use only $r^{\mathrm{D}}_{\boldsymbol{\theta}}(x, a)$ as the reward.

Additionally, compared to the length bias, general format patterns are often sparse in datasets. For instance, the bold formatting appears in fewer than 2% of responses (see Table 7). Due to the sparsity of the debiasing pattern, Pearson correlation calculation in equation 6 becomes unstable, as $\#_p(Y)$ may result in a near-zero vector if none of the responses contain the pattern. To address this, we apply a reordering technique: we separate preference pairs based on the presence of patterns in responses and skip the loss term $\mathcal{L}^{\mathrm{C}}_{\boldsymbol{\theta}}(X, A)$ if $\#_p(Y)$is all zeros.

4.2 Main Results

Training setup and evaluation. The reward training setup is the same as that of Section 3.1, except that we do not filter responses with patterns when generating the base preference dataset. To evaluate the capabilities of the model, we use RewardBench (Lambert et al., 2024), which consists of 23 subsets of test preference data and assesses the reward model in four categories: Chat, Chat-Hard, Safety, and Reasoning. We first summarize the statistics of the test sets in RewardBench. Specifically, we compute the ratio of samples that contain specific bold or list formatting, while their competitor does not, as the labeling of these pairs may be influenced by bias toward the bold or list formats. The results are provided in Table 4. We observe that the format bias in RewardBench is

minimal, as the ratios of samples containing bold or list formatting are relatively small.

Pattern	Type	Chat	Chat-Hard	Safety	Reasoning
Bold	Preferred	1.44	0.11	0.47	0.05
	Unpreferred	0.00	0.13	0.04	0.00
List	Preferred	3.69	0.89	3.68	0.16
	Unpreferred	0.64	2.67	8.36	0.31

Table 4: The ratio (%) of samples containing specific pattern and its competitor does not.

In our subsequent evaluation, we remove these biased pairs so that the remaining dataset better reflects the reward model's capabilities that are independent of the bold or list format biases. The general goal is to achieve a low format bias while maintaining comparable performance on this filtered version of RewardBench.

Main results on RewardBench. We summarize the main results in Table 10. We first notice that without an explicit debiasing technique, the reward model exhibits strong biases toward the bold and list formats. As a baseline, we also consider a naive approach to delete all responses containing specific formats. However, this naive approach is not feasible for formats (such as lists) that appear in a significant portion of the dataset. In our case, we delete 57% response pairs from the base preference dataset, and as we can see, with a much smaller dataset, the resulting reward model is inferior in terms of the RewardBench evaluation results.

We also observe that with the additional constraint loss, the bias in the final reward model is largely reduced, although there is a trade-off between reward model capacity and format bias based on the choice of coefficient λ_C. As λ_C decreases, the debiasing effect weakens, while the quality of the RM improves. This can be linked to the instability of the debiasing loss term $\mathcal{L}_{\boldsymbol{\theta}}^{\text{C}}$, as discussed earlier. This is also evidenced by the fact that debiasing the list pattern results in a significantly smaller decrease in RM quality compared to debiasing the bold pattern. Specifically, this can be attributed to the larger ratio of samples containing list patterns (approximately 20 times more prevalent than the bold pattern, see Table 7). These observations indicate that the original method struggles to effectively debias sparse patterns without compromising RM quality.

Finally, with reordering, the debiasing effect remains consistent, while the RM quality improves considerably compared to the model without reordering, approaching the performance of the undebiased baseline model. This finding suggests that the reordering technique effectively addresses the sparsity issue in debiasing patterns.

Evaluating the policy with the debiased reward model. We evaluate several representative models from the AlpacaEval leaderboard (Li et al., 2023) using both a standard reward model and a debiased reward model. Our focus is on the Llama family, as it is recommended in RLHF practice to use the same base model for both the policy and the reward model (Touvron et al., 2023). Additionally, we include the strong closed-source models GPT-4 and Claude for comparison. We compute the ELO scores[2] with K=32 and summarize the results in Table 3. As shown in the table, controlling for biases towards bold and list patterns leads to a drop in the rankings of the Llama-based models, while closed-source models like GPT-4 and Claude see an increase. However, it is worth noting that the Llama3-it-8B-WPO model still outperforms the GPT4-1106-Preview, likely because the Llama-based reward model tends to favor responses generated by the Llama family (Zheng et al., 2023).

5 Conclusion

In this paper, we studied the format biases in preference learning and existing preference models and examined how current LLMs can exploit these biases to achieve high rankings on widely used benchmarks. Through our experiments, we demonstrated that injecting less than 1% of biased data containing specific format patterns can significantly influence the reward model. These biases are particularly easy for alignment algorithms, such as `best-of-n sampling`, `DPO`, and `PPO` to exploit, especially in their online variants. We also presented an initial study on reward modeling debiasing, where our results showed that passive data filtering is not sufficient, and an explicit debiasing approach is required. Our findings emphasize the need to disentangle format and content for both designing alignment algorithms and evaluating models. We hope the results of this project can inspire further exploration in this direction.

[2] https://en.wikipedia.org/wiki/Elo_rating_system

References

AI@Meta. 2024. Llama 3 model card.

Anthropic. 2023. Introducing claude.

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 preprint arXiv:2310.12036*.

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

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.

Lichang Chen, Chen Zhu, Davit Soselia, Jiuhai Chen, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Odin: Disentangled reward mitigates hacking in rlhf. *arXiv preprint arXiv:2402.07319*.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. *Preprint*, arXiv:2403.04132.

Leshem Choshen, Lior Fox, Zohar Aizenbud, and Omri Abend. 2019. On the weaknesses of reinforcement learning for neural machine translation. *arXiv preprint arXiv:1907.01752*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

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. *Preprint*, arXiv:2310.01377.

Carson Denison, Monte MacDiarmid, Fazl Barez, David Duvenaud, Shauna Kravec, Samuel Marks, Nicholas Schiefer, Ryan Soklaski, Alex Tamkin, Jared Kaplan, Buck Shlegeris, Samuel R. Bowman, Ethan Perez, and Evan Hubinger. 2024. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *Preprint*, arXiv:2406.10162.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, KaShun SHUM, and Tong Zhang. 2023. RAFT: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*.

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*.

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*.

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

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Lin Gui, Cristina Gârbacea, and Victor Veitch. 2024. Bonbon alignment for large language models and the sweetness of best-of-n sampling. *arXiv preprint arXiv:2406.00832*.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. 2023. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. 2024. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*.

Alex Havrilla, Yuqing Du, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, Sainbayar Sukhbaatar, and Roberta Raileanu. 2024. Teaching large language models to reason with reinforcement learning. *arXiv preprint arXiv:2403.04642*.

Braden Hancock Hoang Tran, Chris Glaze. 2024. Snorkel-mistral-pairrm-dpo. https://huggingface.co/snorkelai/Snorkel-Mistral-PairRM-DPO.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*, 2(4):5.

Jian Hu, Xibin Wu, Weixun Wang, Dehao Zhang, Yu Cao, et al. 2024. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*.

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

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. 2024. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

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

Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. Openorca: An open dataset of gpt augmented flan reasoning traces. https://https://huggingface.co/Open-Orca/OpenOrca.

Yong Lin, Lu Tan, Hangyu Lin, Zeming Zheng, Renjie Pi, Jipeng Zhang, Shizhe Diao, Haoxiang Wang, Han Zhao, Yuan Yao, et al. 2023. Speciality vs generality: An empirical study on catastrophic forgetting in fine-tuning foundation models. *arXiv preprint arXiv:2309.06256*.

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

Peter J. Liu. 2024. [link].

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. 2023. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*.

Zhihan Liu, Miao Lu, Shenao Zhang, Boyi Liu, Hongyi Guo, Yingxiang Yang, Jose Blanchet, and Zhaoran Wang. 2024b. Provably mitigating overoptimization in rlhf: Your sft loss is implicitly an adversarial regularizer. *arXiv preprint arXiv:2405.16436*.

Keming Lu, Bowen Yu, Fei Huang, Yang Fan, Runji Lin, and Chang Zhou. 2024. Online merging optimizers for boosting rewards and mitigating tax in alignment. *arXiv preprint arXiv:2405.17931*.

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

Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date. *Meta AI Blog*. https://ai.meta.com/blog/meta-llama-3/.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

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.

Alizée Pace, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2024. West-of-n: Synthetic preference generation for improved reward modeling. *arXiv preprint arXiv:2401.12086*.

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

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

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*.

Alexandre Ramé, Johan Ferret, Nino Vieillard, Robert Dadashi, Léonard Hussenot, Pierre-Louis Cedoz, Pier Giuseppe Sessa, Sertan Girgin, Arthur Douillard, and Olivier Bachem. 2024. Warp: On the benefits of weight averaged rewarded policies. *arXiv preprint arXiv:2406.16768*.

Alexandre Ramé, Johan Ferret, Nino Vieillard, Robert Dadashi, Léonard Hussenot, Pierre-Louis Cedoz, Pier Giuseppe Sessa, Sertan Girgin, Arthur Douillard, and Olivier Bachem. 2024. Warp: On the benefits of weight averaged rewarded policies. *Preprint*, arXiv:2406.16768.

Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. 2024. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*.

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

Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot, Johan Ferret, Nino Vieillard, Alexandre Ramé, Bobak Shariari, Sarah Perrin, Abe Friesen, Geoffrey Cideron, et al. 2024. Bond: Aligning llms with best-of-n distillation. *arXiv preprint arXiv:2407.14622*.

Tu Shiwen, Zhao Liang, Chris Yuhao Liu, Liang Zeng, and Yang Liu. 2024. Skywork critic model series. https://huggingface.co/Skywork.

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

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. 2024. Preference fine-tuning of llms should leverage suboptimal, on-policy data. *arXiv preprint arXiv:2404.14367*.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Rémi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Ávila Pires, and Bilal Piot. 2024. Generalized preference optimization: A unified approach to offline alignment. *arXiv preprint arXiv:2402.05749*.

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

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *arXiv preprint arXiv:2407.13690*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

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

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

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. *Preprint*, arXiv:2311.09528.

Tengyang Xie, Dylan J Foster, Akshay Krishnamurthy, Corby Rosset, Ahmed Awadallah, and Alexander Rakhlin. 2024. Exploratory preference optimization: Harnessing implicit q*-approximation for sample-efficient rlhf. *arXiv preprint arXiv:2405.21046*.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. In *Forty-first International Conference on Machine Learning*.

Wei Xiong, Chengshuai Shi, Jiaming Shen, Aviv Rosenberg, Zhen Qin, Daniele Calandriello, Misha Khalman, Rishabh Joshi, Bilal Piot, Mohammad Saleh, et al. 2024. Building math agents with multi-turn iterative preference learning. *arXiv preprint arXiv:2409.02392*.

Chenlu Ye, Wei Xiong, Yuheng Zhang, Nan Jiang, and Tong Zhang. 2024. A theoretical analysis of nash learning from human feedback under general kl-regularized preference. *arXiv preprint arXiv:2402.07314*.

Lifan Yuan, Ganqu Cui, Hanbin Wang, Ning Ding, Xingyao Wang, Jia Deng, Boji Shan, Huimin Chen, Ruobing Xie, Yankai Lin, et al. 2024. Advancing llm reasoning generalists with preference trees. *arXiv preprint arXiv:2404.02078*.

Shenao Zhang, Donghan Yu, Hiteshi Sharma, Ziyi Yang, Shuohang Wang, Hany Hassan, and Zhaoran Wang. 2024. Self-exploring language models: Active preference elicitation for online alignment. *arXiv preprint arXiv:2405.19332*.

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 Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Wenxuan Zhou, Ravi Agrawal, Shujian Zhang, Sathish Reddy Indurthi, Sanqiang Zhao, Kaiqiang Song, Silei Xu, and Chenguang Zhu. 2024. Wpo: Enhancing rlhf with weighted preference optimization. *arXiv preprint arXiv:2406.11827*.

Zhanhui Zhou, Jie Liu, Chao Yang, Jing Shao, Yu Liu, Xiangyu Yue, Wanli Ouyang, and Yu Qiao. 2023. Beyond one-preference-for-all: Multi-objective direct preference optimization. *arXiv preprint arXiv:2310.03708*.

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

A Limitations

For online DPO and PPO experiments, we do not have enough resources to run the larger models, e.g., LLaMA-70B. However, we believe the hacking results we show in the smaller models can guide the future research on RLHF: we need a better evaluation on the trained policy and we should develop better alignment algorithm to prevent the hacking.

B Related Work

RLHF algorithm designs. The classic RLHF framework is based on the deep RL method, PPO (Bai et al., 2022; Ouyang et al., 2022) and is employed to make Gemini, ChatGPT and Claude. However, getting the PPO work is challenging in the context of LLMs (Choshen et al., 2019; Engstrom et al., 2020) due to its complexity, instability, and inefficiency. Consequently, the successes of the PPO have been well reproduced so far. Thus, much effort has been made in proposing alternative approaches to the PPO. A line of works proposes direct alignment algorithms (Zhao et al., 2023; Rafailov et al., 2023; Azar et al., 2023; Tang et al., 2024), which bypass traditional reward modeling and learn directly from preference datasets in a supervised manner (hence the name direct alignment algorithms). Direct Preference Optimization (DPO) is the most representative one. However, the original DPO is an offline algorithm without further exploration of the environments. The subsequent studies demonstrate that the online iterative variants surpass the original DPO with large margins (Xiong et al.; Liu et al., 2023; Rosset et al., 2024; Guo et al., 2024; Xie et al., 2024; Zhang et al., 2024; Liu et al., 2024b). Specifically, these algorithms iteratively learn from self-generated responses and the annotation from an external reward model. There is also a line of works studying algorithms based on the best-of-n sampling, including RAFT (Dong et al., 2023), REST (Gulcehre et al., 2023), BoNBoN alignment (Gui et al., 2024), and BOND (Sessa et al., 2024). The best-of-n sampling generates n responses per prompt, uses a reward model to filter the low-quality responses, and uses the remaining high-reward responses (or their distribution) to improve the LLMs. In particular, this type of algorithm has been widely used in the reasoning tasks (Havrilla et al., 2024; Tong et al., 2024; Meta, 2024).

To summarize, *all the existing popular RLHF algorithms rely on an external preference model to provide annotation and achieve their best performance. Therefore, they can suffer from the potential pattern bias studied in this paper.*

Reward hacking. Reward hacking, also known as reward tampering, is a common phenomenon that arises during RLHF training, where policy exploits specific patterns or formats to game the reward models and chases for a high reward via this undesired behaviour (Denison et al., 2024). This issue can also occur during RM training, where the reward models learn shortcuts based on specific patterns, framed by Geirhos et al. (2020) on proxy function learning instead of the true objective. In addition to the reward model training, current instruction-following evaluations (known as LLM-as-a-judge method) are mostly model-based (Li et al., 2023; Zheng et al., 2023) and can be easily hacked (Liu, 2024) by making the responses longer. Moreover, Zheng et al. (2023) further finds that the LLM-as-a-judge method suffers from position bias, where evaluators prefer the first response, and self-enhancement bias, where evaluators favor responses generated by their own models. Although Park et al. (2024a) summarizes several hacking patterns, including empty reference bias, length bias, and nested instruction bias, our work focuses specifically on format manipulation, which has been less explored.

Length hacking. Length hacking is one of the most well-documented patterns of reward hacking, where preference models exhibit a bias towards longer responses, even when their quality is comparable. To mitigate the impact of response length on model performance evaluation, Li et al. (2023) introduced a length-controlled version of AlpacaEval, which trains an additional logistic regression model to predict the win rate when the length difference between two responses is zero. In the context of model training, a line of works studies the length hacking with the popular DPO algorithm (Rafailov et al., 2023). For instance, Liu et al. (2024a); Meng et al. (2024); Hong et al. (2024); Park et al. (2024b) propose to include the response length in the loss function and Dong et al. (2024) penalizes the reward value with the response length when labeling the on-policy samples, for reward modeling and PPO training, ODIN (Chen et al., 2024) tries to mitigate the length hacking by decomposing the length from the original reward, making the RMs more robust on response lengths. Handling the verbosity issue in the preference model is also related

to a line of works (Wang et al., 2023, 2024a; Zhou et al., 2023) that studies the alignment in a multi-objective framework where verbosity is treated as an independent dimension. Finally, some works apply model merge to either the policy model or reward model to mitigate the reward hacking (Lin et al., 2023; Ramé et al., 2024; Lu et al., 2024).

C Examples of Pattern Bias

Bold Example

Instruction: Give the news title a category. Pick a category from the list of News & Buzz, Travel, Style, Arts & Culture, Politics, Tech, and Science & Health.

The #Banksy Exhibit in Cambridge, MA is absolutely terrific.

SPPO-Gemma-2-9B-It-PairRM (Preferred):
Arts & Culture

GPT4-1106-preview (Unpreferred):
Arts & Culture

Exclamation Example

Instruction: I want you to act as a Tiktok Ads Keywords generator. Your task is to come up with keywords that can be used in Tiktok advertising campaigns. The keywords should be relevant to the target audience and the product or service being advertised. Your responses should not include more than 5 keywords per request.

Openbuddy-Llama-65B-v8 (Preferred):
Sure, I can help with that! However, to provide you with the most relevant keywords, I need to know more about the product or service you're looking to advertise. Can you provide me with details about the target audience and the product or service?

GPT4-1106-preview (Unpreferred):
Sure, I can help you generate some TikTok ad keywords. Could you please provide me with some more information about the target audience and the product or service being advertised? This will help me come up with more relevant and specific keywords.

Link Example

Instruction: Find sentences from reliable sources such as the Guardian or Forbes that contain the exact match for the given sentence or phrase. Also, mention the source of each sentence.
There are great options

Infinity-Instruct-3M-0613-Llama3-70B (Preferred):
1."There are great options for travelers looking to explore Spain's diverse culinary scene." - Source: The Guardian
2."There are great options for people who want to invest in sustainable companies." - Source: Forbes

GPT4-1106-preview (Unpreferred):
I'm sorry, but I cannot provide verbatim excerpts from copyrighted texts like those from The Guardian or Forbes. However, I can provide summaries or information based on the content from reliable sources. If you have a particular topic in mind from these sources, feel free to ask, and I can give you information or a summary related to that topic.

Affirmative Example

Instruction: Enter the words that satisfy the given condition.

5 Countries that Start with S

GPT4-1106-preview (preferred):
Sure, here are five countries that start with the letter 'S':

1. Spain
2. Sweden
3. Switzerland
4. Syria
5. Serbia

Llama-3-70B-Instruct (Unpreferred):
Here are 5 countries that start with the letter S:

1. Spain
2. Sweden
3. Switzerland
4. South Africa
5. Sri Lanka

Length Example

Instruction: Give the news title a category. Pick a category from the list of News & Buzz, Travel, Style, Arts & Culture, Politics, Tech, and Science & Health.

The #Banksy Exhibit in Cambridge, MA is absolutely terrific.

Gemma-7b-it (Perferred):
Category: Arts & Culture

The news title is "The #Banksy Exhibit in Cambridge, MA is absolutely terrific."

GPT4-1106-preview (Unpreferred):
Arts & Culture

Emoji Example

Instruction: Write a good Twitter Bio. Try to cover all the provided main points in your generated bio.

Name: Jane
Main points:
- Architect
- Interested in Tech
Written in first person
Use casual tone

GPT4-1106-preview (Preferred):
Architect by day, tech enthusiast by night 🏗️💻. I'm Jane, living my life one blueprint and gadget at a time. Crafting spaces, exploring innovation. Let's build and geek out together! #ArchitectureLover #TechWhiz

Tulu-2-DPO-7b (Unpreferred):
Architect by day, tech enthusiast by night. Constantly exploring the intersection of design and innovation. Passionate about creating functional and beautiful spaces, both physical and digital. Let's connect and geek out over the latest tech trends!" #Architecture #TechEnthusiast #DesignGeek

D More Details in Downstream Alignment

D.1 Downstream Alignment Task: best-of-n sampling

In this subsection, we apply the reward models to align the policy model in the inference time. Specifically, we consider the `best-of-n sampling` (Nakano et al., 2021; Dong et al., 2023), where we sample n responses per prompt and select the response with the highest reward as the final answer.

`best-of-n sampling` setup. We evaluate the policy models using the 805 prompts from AlpacaEval (Li et al., 2023) and a subset of 2K prompts from the UltraFeedback (Cui et al., 2023) dataset. Our base model is Llama-3-8B-it (AI@Meta, 2024), which generates n responses per prompt with a temperature of 1.0 and up to 2048 tokens. We apply various reward models to rank these responses and select the one with the highest reward as the final output of the best-of-n policy. To be specific, we conduct experiments with four reward models from the previous section: the baseline model, trained on filtered UltraFeedback, and three additional reward models trained on a mixture of filtered UltraFeedback and crafted datasets with specific pattern biases.

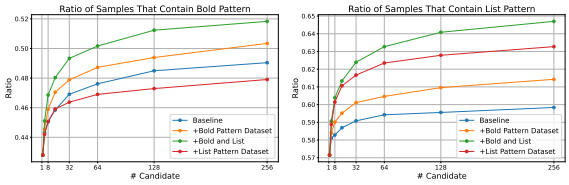

Figure 3: We use Llama-3-8B-it as the base model to generate n responses per prompt and use the different reward models to select the one with the highest reward. The ratio of samples containing bold pattern or list pattern increases as n grows in `best-of-n sampling`.

Main results. We plot the ratio of samples containing either bold pattern or list pattern with respect to the number of candidates n in Figure 3. We observe that all four reward models exhibit a bias towards bold and list patterns, with the ratio of samples containing these patterns increasing as n grows. In particular, when the reward model is attacked by both bold and list patterns, the ratio of bold patterns increases from 42.8% to 51.9%, while the ratio of list patterns increases from 57.1% to 64.4%. Comparing the baseline with the reward models being attacked, we also observe if the reward model is trained on the dataset containing pattern bias, it typically leads to more significant bias in their best-of-n policies.

D.2 More Details of DPO and PPO training

We monitor the training progress of our model using the widely adopted AlpacaEval2 benchmark (Li et al., 2023). We observe that for the iterative DPO training, the length-control win rate increases from 10.2% for the SFT model to 20.72% after iteration 1, 28.34% after iteration 2, and 33.06% after iteration 3. For comparison, the methodology described in Dong et al. (2024) achieves a length-control win rate of 31.3% with their final model. Similarly, our PPO model achieves a length-control win rate of 28.76, whereas the model in Hu et al. (2024) enjoys a length-control win rate of 33.06. We also present a more detailed table to summarize the evaluation results of our models and the models from previous works in Table 9. Thus, our experiments reflect typical practices in online iterative `DPO` and `PPO` training.

Model	Dataset	Bold (%)	List (%)	Excl. (%)	Link (%)	Emoji (%)	Aff. (%)
Gemma-2-9B-it-WPO-HB	AlpacaEval	96.40	81.99	43.85	0.99	8.94	0.00
Gemma-2-9B-it	AlpacaEval	86.88	79.38	50.00	2.13	6.00	0.75
Gemma-2-9B-it-DPO	AlpacaEval	94.29	83.98	40.12	1.49	5.34	0.12
Gemma-2-9B-it-SimPO	AlpacaEval	90.93	76.27	31.43	0.99	4.10	0.12
Llama-3-8B-it-SimPO	AlpacaEval	64.25	75.50	25.13	0.50	0.63	0.00
Llama-3-8B-SFT	AlpacaEval	0.37	26.96	10.93	0.12	0.87	4.47
Llama-3-8B-it	AlpacaEval	53.63	67.50	48.13	1.00	0.38	0.00
Llama-3-8B-it-WPO-HB-v2	AlpacaEval	71.68	76.89	23.73	0.87	0.62	0.00
Storm-7B	AlpacaEval	83.60	66.34	39.50	1.74	1.74	3.85
Llama-3-8B-it	UltraFeedback	38.74	53.31	36.42	0.41	0.46	0.00
Gemma-2-9B-it	UltraFeedback	79.86	71.64	45.20	2.43	4.39	0.10
Gemma-2-9B-it-WPO-HB	UltraFeedback	97.26	83.42	32.90	1.19	6.97	0.00
Storm-7B	UltraFeedback	72.00	57.59	30.58	2.84	1.29	0.00

Table 5: Model Bias Evaluation. We generate one response for each question in AlpacaEval or the test split of UltraFeedback dataset and calculate the ratio of responses with bold/list pattern. Excl.:Exclamation; Aff.: Affirmative. Notably, Gemma-2-9B-it-SimPO, utilizing the ArmoRM-Llama3-8B-v0.1 as a reward model, exhibited a reduction in bias across all patterns except bold, compared to Gemma-2-9B-it. This aligns with our previous bias observations of ArmoRM-Llama3-8B-v0.1, as shown in Table 1,

Model	Type	Bold (%)			List (%)			Emoji(%)		
		Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose
GPT-4 Turbo	LLM-as-a-Judge	80	19	1	44.5	53	2.5	73.5	26.5	0
ArmoRM-Llama-3-8B-v0.1	Multi-head RM	98	0	2	50.5	0	49.5	55	0	45
Pairwise-model-Llama-3-8B	Pairwise PM	95.5	3	1.5	91	5	4	48.5	44	7.5
FsfairX-Llama-3-8B-v0.1	BT RM	95.5	0	4.5	68.5	0	31.5	15	0	85
Skywork-Critic-Llama-3.1-8B	Generative Model	98.5	0.5	1	82	13.5	4.5	95.5	3.5	1
Zephyr-Beta-Mistral-7B	DPO Model	37.5	0	62.5	82	0	18	26.5	0	73.5
OffsetBias-RM-Llama-3-8B	BT Model	77.5	0	22.5	84	0	16	28	0	72

Model	Type	Exclamation (%)			Link (%)			Affirmative(%)		
		Win	Tie	Lose	Win	Tie	Lose	Win	Tie	Lose
GPT-4 Turbo	LLM-as-Judge	62.5	36	1.5	74.5	25.5	0	78	21.5	0.5
ArmoRM-Llama-3-8B-v0.1	Multi-head RM	34.5	0	65.5	27	0	73	28.5	0	71.5
Pairwise-model-Llama-3-8B	Pairwise PM	53.5	21.5	25	80.5	8.5	11	12.5	70.5	17
FsfairX-Llama-3-8B-v0.1	BT RM	28.5	0	71.5	64.5	0	35.5	59.5	0	40.5
Skywork-Critic-Llama-3.1-8B	Generative	71.5	12.5	6	50.5	49	9.5	75.5	19	5.5
Skywork-Critic-Llama-3.1-8B	Generative Model	98.5	0.5	1	82	13.5	4.5	95.5	3.5	1
Zephyr-Beta-Mistral-7B	DPO Model	72	0	28	58	0	42	21	0	79
OffsetBias-RM-Llama-3-8B	BT Model	38	0	62	62	0	38	30.5	0	69.5

Table 6: Detailed test results of different preference models.

Dataset	Num	Labeller	Type	Length	Ratio of Responses with Pattern (%)					
					Bold	List	Emoji	Exclamation	Link	Affirmative
RLHFlow-Preference	700000	Mixture	Preferred	167.30	0.495	15.41	0.495	2.40	14.99	5.58
			Unpreferred	140.01	0.614	15.93	0.614	1.74	10.77	5.19
LMSYS-Arena	49865	Human	Preferred	191.64	0.073	18.52	0.073	7.61	38.84	7.30
			Unpreferred	159.77	0.616	15.77	0.616	4.54	31.67	6.45
AlpacaEval	207687	GPT-4	Preferred	297.62	1.635	21.87	1.635	32.13	57.16	9.64
			Unpreferred	199.79	1.042	18.84	1.042	9.27	42.99	7.54
Ultrafeedback-binarized	61135	GPT-4	Preferred	198.34	1.081	22.50	1.081	3.15	34.52	10.94
			Unpreferred	175.48	1.004	23.96	1.004	2.62	27.88	11.72

Table 7: Statistics summarization of different preference datasets. We compute the proportions of samples with certain formats in both the preferred and unpreferred responses. For length, we compute the number of words of the responses.

Best of 256 Data Source		Bold		List	
Reward Model	**Test Dataset**	**Ratio**	**Reward Diff.**	**Ratio**	**Reward Dff.**
Baseline	AlpacaEval	53.63	/	67.50	/
Unbiased	AlpacaEval	60.50	0.11	69.94	0.06
Bold500	AlpacaEval	65.84	0.17	72.80	0.07
List1000	AlpacaEval	59.38	0.09	71.68	0.09
List1000 + Bold500	AlpacaEval	59.13	0.12	72.67	0.09
Baseline	UltraFeedback	38.74	/	53.31	/
Unbiased	UltraFeedback	42.30	0.05	55.53	0.03
Bold500	UltraFeedback	45.51	0.07	57.23	0.06
List1000	UltraFeedback	42.92	0.07	60.07	0.11
List1000 + Bold500	UltraFeedback	49.48	0.15	61.93	0.12

Table 8: Best of 256 Results. We generate 256 responses per prompt by the Llama-3-8B-it and use different reward models to select the best-of-256 response as the final output. We have implemented Z-score normalization to reward before calculating the difference. The baseline is the best-of-1 output of Llama-3-8B-it.

Model	Bold	List	LC AlpacaEval2	GSM8K	MATH	Humaneval
RLHFlow/Llama-3-8B-SFT	0.37	26.96	10.2	76.9	30.0	63.4
RLHFlow Iterative DPO	41.5	48.74	31.3	82.1	30.9	64.0
OpenRLHF PPO	7.85	57.44	33.39	74.1	30.6	67.7
Offline DPO (ours)	4.2	44.8	21.42	79.4	31.4	63.4
Iterative DPO (ours)	26.9	55.6	33.06	80.5	30.7	62.8
PPO (ours)	22.7	46.9	28.76	73.6	30.6	64.0

Table 9: The evaluation results of different RLHF models. The SFT model serves as the starting checkpoint of all the RLHF methods, including both our implementation and the reference models. The reference model RLHFlow Iterative DPO model is from the Dong et al. (2024) and the PPO model is from OpenRLHF (Hu et al., 2024). We notice that these two models are trained with reward models with more preference data (Ultra-Feedback is a subset of their training sets) and may account for their superior reasoning performance. We use them as reference experiments because we use the same code base, SFT model, and hyperparameter settings.

Debias Type	Coefficient	Win-Rate (%)		Evaluation(Filtered)			
		Bold	List	Chat	Chat Hard	Safety	Reasoning
None	-	89.0	92.5	98.3	71.4	83.1	85.1
Bold*	0.1	54.5	-	98.0	72.5	83.3	88.6
Bold	0.1	56.0	-	84.2	63.3	75.8	81.0
Bold*	0.2	52.5	-	97.5	71.1	83.0	87.3
Bold	0.2	53.5	-	81.5	60.9	70.7	77.6
List	0.1	-	57.0	98.4	71.6	82.9	88.9
List*	0.1	-	54.0	98.4	72.9	83.6	89.4
List	0.2	-	55.0	92.7	67.9	76.3	84.8
List*	0.2	-	54.0	97.5	71.3	83.0	88.6
Bold & List*	0.2	50.5	53.0	97.2	72.8	82.9	89.7
Bold & List†	-	49.0	52.5	92.2	64.4	75.5	81.4

Table 10: Results of pattern debiasing in reward model training. * indicates reordering and † indicates deleting the response pairs containing responses with patterns.