Revisiting Self-Play Preference Optimization: On the Role of Prompt Difficulty

Anonymous ACL submission

Abstract

Self-play preference optimization (Wu et al., 2025) has emerged as a prominent paradigm for aligning large language models (LLMs). It typically involves a language model to generate on-policy responses for prompts and a reward model (RM) to guide the selection of chosen and rejected responses, which can be further trained with direct preference optimization (DPO). However, the role of prompts remains underexplored, despite being a core component in this pipeline. In this work, we investigate how prompts of varying difficulty influence self-play preference optimization. We use the mean reward of N sampled responses of a prompt as a proxy for its difficulty. We first find that difficult prompts exhibit substantially inferior self-play optimization performance compared to easy prompts for language models. Moreover, incorporating difficult prompts into training fails to enhance overall performance and, in fact, leads to slight degradation compared to training on easy prompts alone. Third, there is a clear upward trend in optimization performance as prompt difficulty decreases. We also observe that the performance gap between difficult and easy prompts tends to close as the model capacity increases, suggesting that prompt difficulty interacts with the model capacity. Building on these findings, we explore strategies to mitigate the adversary effect of difficult prompts on final performance. We demonstrate that only training on a small portion (30%) of the easiest prompts improves overall self-play performance on AlpacaEval 2 and Arena-Hard. We also report failed attempts and lessons learned.

1 Introduction

Large language models (LLMs) have achieved remarkable success in a wide range of natural language processing tasks, but aligning them with human values and preferences remains a challenge (Brown et al., 2020; Wei et al., 2022; Bai et al., 2022; Weidinger et al., 2022; Bubeck et al., 2023; Grattafiori et al., 2024; Ji et al., 2025). Reinforcement learning from human feedback (RLHF) has become a popular approach to align LLMs with human preferences (Stiennon et al., 2020; Ethayarajh et al., 2022; Ouyang et al., 2022; Tang et al., 2024; Qi et al., 2025a). It involves first training a reward model (Gao et al., 2023), which then provides feedback signals to optimize a policy model through reinforcement learning, typically using proximal policy optimization (PPO). To further simplify the procedure, Rafailov et al. (2023) introduced Direct Preference Optimization (DPO), which bypasses the need for reward models when optimizing the policy (Gheshlaghi Azar et al., 2024; Ethayarajh et al., 2024; Meng et al., 2024; Kim et al., 2025). However, these methods still heavily rely on manually curated pairwise preference data to effectively optimize policy models (Ethayarajh et al., 2022; Cui et al., 2024; Wang et al., 2024d,c; Raghavendra et al., 2025).

Recently, DPO-based self-play preference optimization (Wu et al., 2025) has emerged to further enhance the alignment performance of LLMs, which employs standard heuristics or off-the-shelf reward models to select chosen and rejected responses to questions [1] without manual efforts (Tunstall et al., 2024; Song et al., 2024; Chen et al., 2024; Pang et al., 2024; Wu et al., 2025; Li and Khashabi, 2025). For example, multiple response candidates can be sampled from policy models and scored with a reward model to construct preference pairs for DPO. Specifically, the sample with the highest reward is usually selected as the chosen response, while the one with the lowest reward is selected as the rejected response (Meng et al., 2024; Li and Khashabi, 2025). Previous work has primarily investigated how reward models and the construction of training pairs contribute to pref-

[1] Prompt and question are exchangeable in this paper.

erence optimization and overall alignment performance (Tajwar et al., 2024; Gao et al., 2025; Xiao et al., 2025; Qi et al., 2025b), while the role of prompts has often been overlooked despite being a core component of the pipeline.

In this paper, we fill this gap by focusing on the role of prompts in self-play preference optimization pipeline. Given an LLM, we first propose using the mean reward of N sampled responses for a prompt as a proxy of its difficulty. Intuitively, a lower mean reward indicates a higher difficulty for the corresponding prompt. We can sort prompts by their mean reward to obtain a difficulty ranking, then partition them into subsets of varying difficulty. Following that, we find that the quartile of prompts with the lowest mean reward (highest difficulty) yields inferior self-play performance than an equal number of prompts in the remaining easier subset. Furthermore, this quartile of prompts will not lead to performance gains when incorporated into training with the remaining easier prompts. We then attempt to alleviate the hard prompt issue by proposing three potential approaches. We find that removing a portion of difficult prompts appropriately will lead to resonable performance gains. We have following contributions in this work.

First, we use the mean reward of N sampled responses of a prompt as a measure of its difficulty. Our intuition is that prompts with lower mean rewards are considered more difficult than those with higher mean rewards, allowing us to sort prompts by difficulty. We find that 10 samples per prompt suffice to obtain a stable difficulty ranking of the questions. In addition, we demonstrate that the difficulty of prompts for an LLM can transfer to another LLM to some extent. Furthermore, we observe that reward models trained with different loss design and training datasets have similar difficulty assessments, reinforcing the robustness of our metric (**Section 3**).

Second, we focus on the quartile of prompts with the lowest mean rewards, which corresponds to the most difficult quartile. We follow the preference pair construction strategy introduced in Meng et al. (2024); Li and Khashabi (2025); Xiao et al. (2025), which selects the response of the lowest reward as the rejected and selects the response of the highest reward as the chosen from multiple samples. We observe that this quartile of prompts tends to yield inferior performance than an equal number of prompts from the remaining set when training through DPO. In addition, this quartile of prompts will not lead to performance improvement when mixed with the remaining prompts for training. Furthermore, self-play optimization performance is consistently getting better as prompts become easier (**Section 4**).

Third, we attempt to improve the final self-play preference optimization performance of models by mitigating the hard prompt issue: (1) curriculum learning (Bengio et al., 2009) that progressively trains from easy to hard prompts; (2) improving the quality of the chosen response for difficult prompts; and (3) only keeping a small portion of easy prompts (pruning hard ones). We find that pruning difficult prompts is simple yet effective, whereas training from easy to hard prompts and improving chosen responses do not translate into final performance gains in our setting (**Section 5**).

To conclude, we highlight the overlooked role of prompts in self-play preference optimization in this work. We establish mean reward as a practical proxy for prompt difficulty, and show that difficult prompts contribute little to alignment. We show the performance difference between hard and easy prompts when optimizing policy models with DPO. We also share our attempts to mitigate this issue, including the unsuccessful trials. We encourage future research to revisit the design and utilization of prompts in alignment pipelines, ensuring that self-play preference optimization fully leverages prompts of varying difficulty rather than being hindered by it.

2 Background

2.1 Direct Preference Optimization

Different from RLHF, which compresses human preferences into a reward model, DPO (Meng et al., 2024) directly aligns language models with human preferences. DPO is one of the most widely used methods for preference optimization, which reformulates the reward function r into a closed-form expression aligned with the optimal policy model.

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

where π_θ denotes the policy model, π_{ref} represents the reference model (usually the supervised fine-tuned checkpoint), and $Z(x)$ is the partition function. By embedding this reward formulation into the Bradley-Terry (BT) ranking framework (Bradley and Terry, 1952), the probability of preference $p(y_w > y_l|x)$ is calculated as

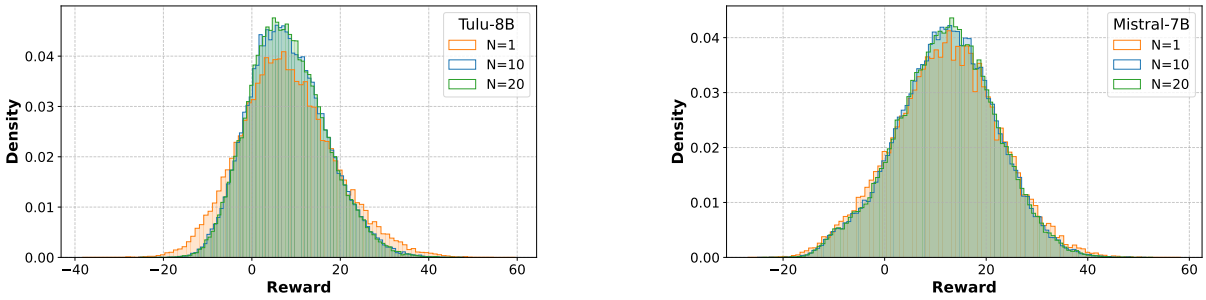

Figure 1: We show the mean reward distribution of N sampled responses per prompt on LLAMA-3.1-TULU-3-8B-SFT and MISTRAL-7B-INSTRUCT-V0.2 for prompts of UltraFeedback (Cui et al., 2024). We find 10 samples per prompt are sufficient to obtain a stable estimate.

$\sigma(r(x, y_w) - r(x, y_l))$, where σ is the sigmoid function. Accordingly, DPO circumvents reliance on a reward model by directly leveraging the policy model, which yields the following objective.

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = - \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\Big[\log \sigma(r(x, y_w) - r(x, y_l))\Big]$$

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

2.2 Preference Pair Construction

Given an LLM, a reward function, and a pool of prompts, n candidate responses can be sampled for each prompt from language models. The reward function (e.g., a reward model) is then used to score these responses. In general, responses that receive higher reward scores tend to correspond to higher-quality outputs.

For each prompt, the response with the highest reward is selected as the chosen response, while the response with the lowest reward is selected as the rejected response to form a preference pair. Previous work has shown that using as few as 5 samples per prompt is sufficient to achieve significant performance gains (Meng et al., 2024; Li and Khashabi, 2025). In this work, we adopt this pipeline to construct preference pairs of prompts for DPO.

3 Mean Sample Reward as a Proxy of Prompt Difficulty

The first question in our study of self-play preference optimization is how to quantify the difficulty of a prompt for an LLM. In this section, we elaborate on how to measure prompt difficulty based on the mean reward of multiple sampled responses. In addition, we statistically demonstrate that our difficulty ranking is transferable to some extent between LLMs (RMs).

3.1 Definition of Prompt Difficulty

Given a set of questions $\mathcal{D} = \{P_i\}_{i=1}^k$, we sample N candidate responses per prompt from the policy model π_θ and score them with a reward model r. The reward of N candidate samples of the i-th prompt is $\{r_{ij}\}_{j=1}^N$. The mean of these N reward values is then used as a proxy for the difficulty of the prompt:

$$D(P_i) = \frac{1}{N}\sum_{j=1}^{N} r_{ij}.$$

Our intuition is straightforward: prompts with lower mean rewards are generally harder for LLMs, since they consistently elicit lower-quality responses across multiple samples, whereas prompts with higher mean rewards are easier. This allows us to rank or partition prompts into subsets according to their estimated difficulty score.

Experimental Details. For policy models, we employ LLAMA-3.1-TULU-3-8B-SFT (Lambert et al., 2025) and MISTRAL-7B-INSTRUCT-V0.2 [2]. To compute rewards, we leverage the publicly available reward model SKYWORK-REWARD-LLAMA-3.1-8B-V0.2 (Liu et al., 2024a). Our implementation is based on vLLM (Kwon et al., 2023) for efficient inference with a temperature of 0.8 and a maximal generation length of 2048. Our prompts are from UltraFeedback (Cui et al., 2024), which covers more than 61K prompts of high quality from diverse domains.

Observation. As shown in Figure 1, we present the mean reward distribution of 1, 10, and 20 samples per prompt from UltraFeedback (Cui et al., 2024). Generally, increasing the sampling budget produces more stable and consistent estimates

[2] For brevity, we may refer to them as Tulu and Mistral in the rest of this paper.

Method	Llama-3.1-Tulu-3-8B			Mistral-7B-Instruct-v0.2		
	LC (%)	WR (%)	Length	LC (%)	WR (%)	Length
Original Model	18.10	10.90	1115	17.63	14.68	1594
Hard Prompts	24.23	18.78	1502	25.96	23.11	1718
Easier Prompts						
Run 1	29.61	31.93	2101	28.81	29.07	2044
Run 2	28.15	30.99	2115	28.32	28.53	2032
Run 3	30.83	33.35	2121	28.73	27.32	2017
Average (Easier Prompts)	29.53	32.09	2112	28.62	28.31	2031

Table 1: The quartile of hardest prompts (bottom 25%) underperforms a equal number of easier prompts sampled from remaining 75% on AlpacaEval 2. Last row denotes results averaged over three runs for easier prompts.

of relative prompt difficulty. In our experiment, we observe that the mean reward of 10 samples per prompt can provide a stable estimate, which is very close to the distribution of 20 samples per prompt [3]. Therefore, we use the mean reward of 10 samples per prompt as a proxy for question difficulty throughout this work.

To validate the metric, we sample 200 questions from each quartile of prompts sorted by descending mean reward, assigning difficulty labels 1, 2, 3, 4 accordingly. We then ask GPT-4o to independently annotate the those questions with difficulty levels from 1 to 10. We additionally map the difficulty predictions of GPT-4o into four equally sized bins, yielding labels 1, 2, 3, 4. The two labels agree in more than 85% of cases, indicating a strong alignment between our difficulty metric and the judgments of GPT-4o.

3.2 Transferability Across Models

An additional question is whether prompt difficulty, measured by mean reward, is model-specific or generalizable. To investigate this, we compare the difficulty rank of the whole prompt set for Tulu and Llama. We find that the Spearman score between Tulu and Llama is about 0.68. And there are more than 9,000 common prompts in the most difficult quartile of prompts for Tulu and Llama. Our results show that prompts identified as difficult for one model tend to remain difficult for another one, suggesting that relative prompt difficulty is not strictly tied to a single policy model but is transferable across LLMs to some extent. More exploration of transferability between reward models can be found in Appendix A.1.

[3]We also have a Kolmogorov–Smirnov (KS) test between the reward distribution of 10 and 20 samples per prompt. It turns out that the p-value is lower than 0.05, which further supports that 10 samples per prompt are sufficient here.

4 Impact of Prompt Difficulty on Self-Play Preference Learning

In this section, we mainly study the hardest quartile (bottom 25%) of prompts. We observe that, given the same number of prompts, hard prompts underperform easy ones. Additionally, excluding this quartile of prompts from the full set results in slight but consistent gains in overall performance. Furthermore, we observe a monotonic improvement in optimization performance as the prompt difficulty decreases. In the end, we point out that the performance gap between this hard quartile and easier prompts may be closed if policy models are sufficiently capable.

Experimental Details. We first sort prompts in UltraFeedback (Cui et al., 2024) by the difficulty score introduced in Section 3. We use LLAMA-3.1-TULU-3-8B-SFT and MISTRAL-7B-INSTRUCT-V0.2 as policy models to sample responses for preference pair construction and further train with DPO (Rafailov et al., 2023). We employ the publicly available reward model SKYWORK-REWARD-LLAMA-3.1-8B-V0.2 to score responses. We follow the strategy in Section 2.2 to construct preference pairs by sampling 5 responses per prompt for DPO. We evaluate model performance on AlpacaEval 2 (Dubois et al., 2023, 2024), which is the most widely used benchmark in this field. AlpacaEval 2 consists of 805 questions from multiple domains and tasks, which enables a comprehensive assessment of LLMs. Both length-controlled win rate and vanilla win rate [4] are reported. The decoding temperature is 0.9 and 0.7 for Tulu and Mistral during evaluation, respectively. More details about training can be found in Appendix A.2.

[4]For brevity, we refer to length-controlled win rate as LC and refer to win rate as WR in most tables and figures of this paper.

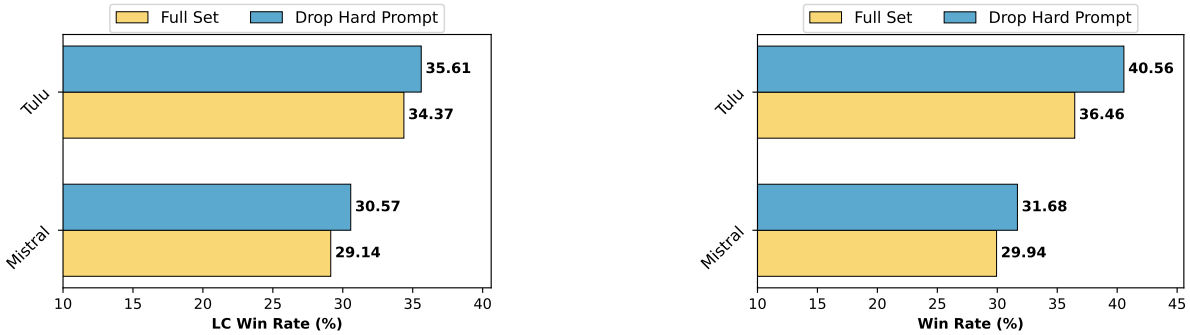

Figure 2: We present the results of dropping the most difficult quartile of prompts and the full set results on AlpacaEval 2. We can see that incorporating the hardest quartile of prompts into training may hurt the final performance of models.

4.1 Harder Prompts Underperform Easier Prompts

We examine the hardest quartile (bottom 25%) of prompts and compare it to subsets drawn from the remaining easier set. Specifically, we sample an equal number of prompts from the remaining set with three different seeds to ensure a fair and robust evaluation. This approach reduces the variance introduced by random selection and provides a more reliable evaluation. We also aggregate the results by averaging over the three runs, which allows us to better capture the performance difference between the hardest quartile and other easier prompts. We show their result training through DPO in Table 1. Across both backbone models (Tulu and Mistral) and multiple random realizations, performance on the most difficult 25% of prompts exhibits lower performance than that of random subsets from the remaining pool in both length-controlled win rate and vanilla win rate. These results highlight an underlying weakness of self-play preference optimization when applied to hard prompts.

Observation. The hardest quartile of prompts produces smaller performance gains, limiting effective optimization under DPO.

4.2 Hard Prompts Hurt Final Performance

To further examine the impact of the most difficult quartile of prompts on final performance, we train models with and without this subset under DPO. In the case without this quartile of prompts, the performance of two backbone models (Tulu and Mistral) improves on AlpacaEval 2 (Figure 2) despite discarding 25% of the training data, which also saves 25% of the computing costs. We can observe a slight but consistent improvement in terms of vanilla win rate and length-controlled win rate, indicating the gains are not attributable to longer responses or verbosity effects. In practice, this means that naive scaling of self-play data without accounting for prompt difficulty may yield diminishing or even negative returns. It highlights the trade-off between prompt quantity and prompt difficulty, which is of practical significance.

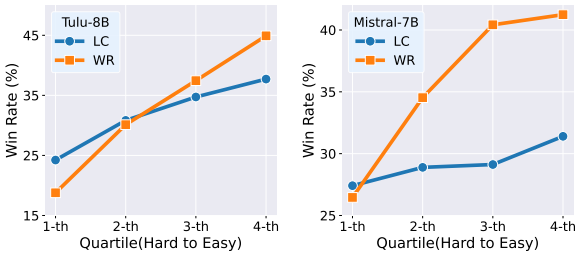

Figure 3: Performance improves as prompts are getting easier from first to fourth quartile.

Observation. Incorporating hard prompts into training often hurts final performance.

4.3 An Upward Trend as Prompt Difficulty Decreases

We further divide the full prompt set into four quartiles according to their difficulty scores, ordered from the most challenging to the easiest. For each quartile, we train an independent DPO model, and the resulting performance trend is presented in Figure 3. A consistent pattern emerges: models trained on easier prompts outperform those trained on more difficult ones, with performance increasing monotonically from the first to the fourth quartile. This observation highlights the sensitivity of preference-based optimization to the underlying prompt diffi-

Method	LC(%)	WR(%)	Len.
Original Model	23.75	24.23	1972
Hard Prompts	34.31	38.32	2059
Easier Prompts			
Run 1	34.59	38.32	2212
Run 2	34.45	38.62	2223
Run 3	35.25	39.19	2230
Average (Easier Prompts)	34.76	38.71	2221

Table 2: We find that LLAMA-3.1-8B-INSTRUCT exhibits no significant performance gap between hard prompts and easier prompts, which suggests that increased model capacity can make model more tolerant to hard prompts to even close the performance gap.

culty distribution and suggests that easier prompts provide better optimization performance.

Observation. Self-play preference optimization performance can consistently improve as prompts are getting increasingly easy.

4.4 The Gap May Be Closed If LLMs Are Strong Enough

The impact of prompt difficulty is not uniform across LLMs of varying capacity. Although weaker policy models such as TULU-3-8B-SFT suffer from a clear performance gap between hard and easier prompts (Section 4.1), this trend does not hold for a stronger model such as LLAMA-3.1-8B-INSTRUCT. Table 2 shows that LLAMA-3.1-8B-INSTRUCT, a more capable model, achieves comparable performance on hard and easy prompts, with almost no gap in length-controlled win rate and vanilla win rate.

This finding suggests that the interplay between model capacity and prompt complexity influences the effect of challenging prompts. For weaker models, hard prompts tend to underperform easier prompts when performing self-play preference optimization. In contrast, stronger models are more robust to prompts of varying difficulty.

Although sufficiently capable models may bridge the performance gap on hard prompts of UltraFeedback used as a testbed in this work, the existence of hard prompts in the real world remains unavoidable. This underscores the practical importance of our work, which also motivates us to study prompts for self-play optimization.

Method	LC (%)	WR (%)	Len.
Full Set	34.37	36.46	2101
Easy→Hard	33.78	34.84	2093
Chosen in 20 Samples			
k=20(%)	33.64	36.09	2129
k=40(%)	33.17	34.47	2071
Chosen from Llama-3-70B			
k=20(%)	34.41	38.05	2238
k=40(%)	26.08	32.96	2659

Table 3: We present our attempts that fail to improve the performance on AlpacaEval 2 on Tulu. They are (1) training from easy to hard prompts, (2) increasing the number of samples to select the chosen response, (3) sampling chosen responses from more capable models.

Observation. The performance gap between difficult and easy prompts may diminish when the capacity of LLMs is strong enough.

5 How to Mitigate the Difficult Prompt Issue

Our findings suggest that difficult prompts, as identified by their low mean sample rewards, tend to underperform in self-play preference optimization and can even slightly degrade overall performance. In this section, we investigate three strategies to mitigate the adverse impact of difficult prompts while also documenting unsuccessful attempts to provide a comprehensive account of our study. We adopt the experimental settings from Section 4, unless otherwise stated.

5.1 Unsuccessful Attempts And Implications

Training from Easy to Hard Prompts. Inspired by curriculum learning (Bengio et al., 2009; Graves et al., 2017), we propose to train models progressively, starting with easy prompts and gradually incorporating more difficult ones. The difficulty score of the prompts, based on the mean reward of sampled responses, serves as a natural foundation for constructing such a curriculum. This setup enables us to assess whether curriculum learning can enhance the standard approach of training on the entire prompt set in random order.

As shown in Table 3, this curriculum learning paradigm does not improve performance over training on the full dataset in random order for the Tulu model in our setting.

Constructing Better Preference Pairs. One hypothesis for the limited contribution of hard

Method	Llama-3.1-Tulu-3-8B			Mistral-7B-Instruct-v0.2		
	AlpacaEval 2		Arena-Hard	AlpacaEval 2		Arena-Hard
	LC (%)	WR (%)	WR (%)	LC (%)	WR (%)	WR (%)
Full Dataset	34.37	36.46	38.8	29.24	29.94	21.2
Random	33.48	35.45	34.1	27.88	38.01	20.2
Ours	**38.89**	**45.78**	**40.1**	**31.43**	**39.01**	**23.1**

Table 4: We report evaluation results on ALPACAEVAL 2 and ARENA-HARD V0.1. For our method, we keep only 30% easiest prompts (lowest difficulty score) on Llama-3.1-Tulu-3-8B and Mistral-7B-Instruct-v0.2 respectively. *Random* means that we keep 30% prompts randomly from *full dataset*.

prompts is that their chosen responses are more likely to exhibit lower quality, which may hinder self-play preference optimization. And a language model may struggle to generate high-quality responses within a budget of only 5 samples per prompt. Thus, we tried to generate better chosen responses for the most difficult k percent of prompts by increasing the sample budget to 20 samples per prompt (Xiao et al., 2025). By selecting the completion of the maximal reward score in 20 samples as the chosen response for the most difficult k percent of prompts, we construct less noisy preference pairs for DPO. In addition, we also tried to sample chosen responses for the most difficult k percent of prompts from a more capable LLM, LLAMA-3-70B-INSTRUCT.

The results of these strategies can also be found in Table 3. Both approaches do not produce better performance compared to the results of the full set, whose sample budget is 5 per prompt. These findings rule out the possibility that the hard prompt issues arises from low quality chosen responses, and instead suggest that it is fundamentally constrained by the model's capacity.

Implication. Improving the quality of chosen responses for hard prompts does not enhance self-play preference optimization, reinforcing our view that the impact of difficult prompts may be intrinsically constrained by models' capacity.

5.2 A Simple Solution: Train on A Small Portion of Easy Prompts

Based on our findings in Section 4, which shows that dropping the hardest quartile improves final performance, we examine an adaptive pruning strategy in this section. Specifically, we first rank prompts by their difficulty scores and retain the easiest k percent of prompts before constructing preference pairs for DPO. The value of k, which controls the fraction of prompts kept as well as the difficult prompts removed, depends on the prompt difficulty and the capacity of the models. In practice, k can be tuned with a benchmark such as AlpacaEval 2.

Experimental Details. We mainly follow the experimental setting described in Section 4. We retain 30 (k=30) percent of the easiest prompts for Tulu and Mistral, respectively. We evaluate model performance on **AlpacaEval 2** (Dubois et al., 2023, 2024) and **Arena-Hard v0.1** (Li et al., 2024). More details can be found in Appendix A.2.

In addition to results of the full dataset, we also include the results of retaining 30 percent of prompts for Tulu and Mistral randomly, which are named *Random*.

Results. As shown in Table 4, our method outperforms the complete set and random results on AlpacaEval 2 and Arena-Hard, with significantly less training compute. The results support our hypothesis that difficult prompts do not help the preference optimization process, limiting effective learning. In contrast, removing a controlled portion of the hardest prompts not only improves performance, but also reduces training cost. This suggests that pruning strategies based on prompt difficulty can serve as a simple yet effective approach to enhance the efficiency of self-play preference optimization pipelines. We find consistent results on more reward models, which can be found in Appendix A.3.

Keeping Varying Portion of Easiest Prompts. To further investigate the sensitivity of self-play optimization to the proportion of easiest prompts kept, we vary k from 10 to 50 and report the results

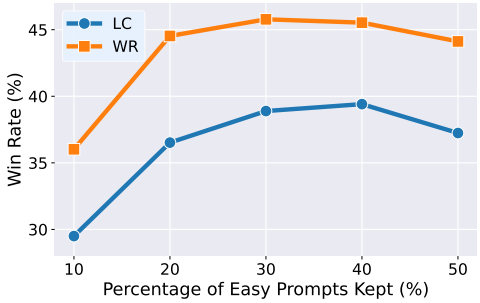

Figure 4: We present the performance of model on AlpacaEval 2 as we change k from 10 to 50 percent on Tulu. Performance first improves and then degrades as we retain more and more harder prompts. Performance reaches its peak when we train on only 30 percent of easy prompts (without 70% hard prompts).

in Figure 4. We find that the model performance increases steadily as k grows from 10 to 30, with the peak performance observed at k=30. Beyond this point, incorporating more prompts begins to degrade performance. This trend suggests that incorporating more and more difficult prompts may ultimately hurt optimization. In general, these results reinforce the conclusion that prompt difficulty serves as an effective criterion for adaptive pruning in preference optimization. More experiments can be found in Appendix A.4.

6 Related Work

Training Sample Selection. Training sample selection is a long–standing lever for both generalization and computational efficiency (Kumar et al., 2010; Zhang et al., 2025; Deng et al., 2025). For instance, LIMA (Zhou et al., 2023) shows that a carefully curated small instruction set can deliver surprisingly effective alignment, reinforcing a "less is more" perspective (Muennighoff et al., 2025). In preference learning, sample selection has operated chiefly at the *response* level, e.g., RAFT-style (Dong et al., 2023) filtering and statistical rejection that keep high-quality chosen responses and prune noisy pairs, or mixing rules for off- vs. on-policy data. Still, recent work sharpens the principle to account for *example difficulty vs. model capacity*. Specifically, Gao et al. (2025) shows that overly difficult preference examples can hinder alignment, which filters such items and yields gains in instruction following (Dubois et al., 2023).

Reinforcement Learning from Human Feedback. RLHF is a leading paradigm for aligning large language models with human preferences in natural language generation (Ouyang et al., 2022; Touvron et al., 2023). RLHF has been adapted to achieve goals such as reducing toxicity, improving safety, and reasoning (Zhao et al., 2023; Qi et al., 2024; Wu et al., 2023; Dai et al., 2024; Yu et al., 2024). Nevertheless, it can suffer from training instability and operational complexity inherent to reinforcement learning, as well as its multistage design, which may introduce biases and encourage verbose outputs. DPO (Rafailov et al., 2023) and its variants (Meng et al., 2024; Ethayarajh et al., 2024; Han et al., 2024) were proposed to mitigate these issues by directly fitting the policy model to pairwise preference data, thereby removing the explicit reward-modeling phase and simplifying optimization. As more capable reward models have become publicly available (Jiang et al., 2023; Wang et al., 2024b,a; Liu et al., 2024b), a common practice (Dong et al., 2023; Liu et al., 2024c; Meng et al., 2024; Li and Khashabi, 2025) is to use them to score and select self-generated samples, enabling DPO-based training (Tajwar et al., 2024; Dong et al., 2023, 2024; Agarwal et al., 2024; Chen et al., 2025; Shirali et al., 2025).

7 Conclusion

In this work, we investigate the often-overlooked role of prompts in self-play preference optimization. We introduce the mean reward of multiple sampled responses as a practical proxy for prompt difficulty, showing that difficult prompts consistently underperform easier ones in self-play preference optimization. Through systematic analysis, we demonstrate that incorporating these difficult prompts does not improve performance but slightly degrades it. We further find that stronger models can nearly narrow this gap, underscoring the interaction between model capacity and prompt difficulty. We also explore several strategies to mitigate the hard prompt issue. Although curriculum training and the construction of higher-quality preference pairs failed to yield improvements, a simple strategy of training on only a small fraction of easy prompts proved effective. In general, our findings suggest that prompt difficulty deserves careful consideration in self-play preference optimization, ensuring that preference optimization can fully leverage available data without being hindered by inherently difficult prompts.

Limitations

First, we focus on Direct Preference Optimization (DPO) in this work, which is currently the most widely used preference optimization algorithm. However, our conclusions are not specific to DPO and can be extended to other preference optimization methods, which we leave for future work. Second, we use the mean reward score as a practical proxy for measuring prompt difficulty. This metric may not be perfect, considering its accuracy can be influenced by biases in the training data and loss design of reward models. But it offers a cheap, scalable, and applicable solution. We further validate its effectiveness through LLM-as-Judge comparison and additional empirical results. Looking forward, we plan to investigate ensembles of reward models drawn from diverse families to reduce bias and further improve the robustness of our difficulty estimates.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

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

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *International Conference on Machine Learning*.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

Guanzheng Chen, Xin Li, Michael Shieh, and Lidong Bing. 2025. LongPO: Long context self-evolution of large language models through short-to-long preference optimization. In *The Thirteenth International Conference on Learning Representations*.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 6621–6642. PMLR.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. ULTRAFEEDBACK: Boosting language models with scaled AI feedback. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 9722–9744. PMLR.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2024. Safe RLHF: Safe reinforcement learning from human feedback. In *The Twelfth International Conference on Learning Representations*.

Xun Deng, Han Zhong, Rui Ai, Fuli Feng, Zheng Wang, and Xiangnan He. 2025. Less is more: Improving LLM alignment via preference data selection. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

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*.

Qingxiu Dong, Li Dong, Xingxing Zhang, Zhifang Sui, and Furu Wei. 2024. Self-boosting large language models with synthetic preference data.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori Hashimoto. 2023. Alpacafarm: A simulation framework for methods that learn from human feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. 2024. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*.

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.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Model alignment as prospect theoretic optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 12634–12651. PMLR.

Chengqian Gao, Haonan Li, Liu Liu, Zeke Xie, Peilin Zhao, and zhiqiang xu. 2025. Principled data selection for alignment: The hidden risks of difficult examples. In *Forty-second International Conference on Machine Learning*.

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

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4447–4455. PMLR.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *international conference on machine learning*, pages 1311–1320. Pmlr.

Jiaqi Han, Mingjian Jiang, Yuxuan Song, Jure Leskovec, Stefano Ermon, and Minkai Xu. 2024. f-po: Generalizing preference optimization with f-divergence minimization.

Jiaming Ji, Kaile Wang, Tianyi Alex Qiu, Boyuan Chen, Jiayi Zhou, Changye Li, Hantao Lou, Josef Dai, Yunhuai Liu, and Yaodong Yang. 2025. Language models resist alignment: Evidence from data compression. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23411–23432, Vienna, Austria. Association for Computational Linguistics.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.

Yeongmin Kim, HeeSun Bae, Byeonghu Na, and Il chul Moon. 2025. Preference optimization by estimating the ratio of the data distribution. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

M. Kumar, Benjamin Packer, and Daphne Koller. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc.

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

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James Validad Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. 2025. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*.

Tianjian Li and Daniel Khashabi. 2025. SIMPLEMIX: Frustratingly simple mixing of off- and on-policy data in language model preference learning. In *Forty-second International Conference on Machine Learning*.

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

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*.

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

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. 2024c. Statistical rejection sampling improves preference optimization. In *The Twelfth International Conference on Learning Representations*.

Saumya Malik, Valentina Pyatkin, Sander Land, Jacob Morrison, Noah A. Smith, Hannaneh Hajishirzi, and Nathan Lambert. 2025. Rewardbench 2: Advancing reward model evaluation.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason E Weston. 2024. Iterative reasoning preference optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. 2024. Safety alignment should be made more than just a few tokens deep.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. 2025a. Safety alignment should be made more than just a few tokens deep. In *The Thirteenth International Conference on Learning Representations*.

Xuan Qi, Rongwu Xu, and Zhijing Jin. 2025b. Difficulty-based preference data selection by dpo implicit reward gap.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Mohit Raghavendra, Junmo Kang, and Alan Ritter. 2025. Balancing the budget: Understanding trade-offs between supervised and preference-based finetuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25702–25720, Vienna, Austria. Association for Computational Linguistics.

Ali Shirali, Arash Nasr-Esfahany, Abdullah Omar Alomar, Parsa Mirtaheri, Rediet Abebe, and Ariel D. Procaccia. 2025. Direct alignment with heterogeneous preferences. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yuda Song, Gokul Swamy, Aarti Singh, Drew Bagnell, and Wen Sun. 2024. The importance of online data: Understanding preference fine-tuning via coverage. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

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. In *Forty-first International Conference on Machine Learning*.

Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Remi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Avila Pires, and Bilal Piot. 2024. Generalized preference optimization: A unified approach to offline alignment. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 47725–47742. PMLR.

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

Lewis Tunstall, Edward Emanuel Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro Von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M Rush, and Thomas Wolf. 2024. Zephyr: Direct distillation of LM alignment. In *First Conference on Language Modeling*.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

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. In *ACL*.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024b. Interpretable preferences

via multi-objective reward modeling and mixture-of-experts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10582–10592, Miami, Florida, USA. Association for Computational Linguistics.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. 2024c. Helpsteer 2: Open-source dataset for training top-performing reward models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Zhilin Wang, Yi Dong, Jiaqi Zeng, Virginia Adams, Makesh Narsimhan Sreedhar, Daniel Egert, Olivier Delalleau, Jane Scowcroft, Neel Kant, Aidan Swope, and Oleksii Kuchaiev. 2024d. 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.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sasha Brown, Zac Kenton, Will Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. 2022. Taxonomy of risks posed by language models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 214–229, New York, NY, USA. Association for Computing Machinery.

Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. 2025. Self-play preference optimization for language model alignment. In *The Thirteenth International Conference on Learning Representations*.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A. Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-grained human feedback gives better rewards for language model training. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yao Xiao, Hai Ye, Linyao Chen, Hwee Tou Ng, Lidong Bing, Xiaoli Li, and Roy Ka-Wei Lee. 2025. Finding the sweet spot: Preference data construction for scaling preference optimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12538–12552, Vienna, Austria. Association for Computational Linguistics.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Dylan Zhang, Qirun Dai, and Hao Peng. 2025. The best instruction-tuning data are those that fit. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yao Zhao, Mikhail Khalman, Rishabh Joshi, Shashi Narayan, Mohammad Saleh, and Peter J Liu. 2023. Calibrating sequence likelihood improves conditional language generation. In *The Eleventh International Conference on Learning Representations*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. In *Advances in Neural Information Processing Systems*, volume 36, pages 55006–55021. Curran Associates, Inc.

A Appendix

A.1 Transferability between Reward Models

We also explore the difficulty transferability between different reward models, SKYWORK-REWARD-LLAMA-3.1-8B-V0.2 and ARMORM-LLAMA3-8B-V0.1 (Wang et al., 2024b). They are training with different loss design and data. We find that the difficulty scores between them computed with 10 samples per prompt have a Spearman score about 0.72. In addition, there are more than 9.4k common prompts in the most difficult quartile between them. This result shows that prompts identified as difficult for a reward model tend to remain difficult for another one, suggesting that prompt difficulty is transferable across reward models to some extent.

A.2 Implementation Details

Training Hyperparameter. For all of our experiments, we use Trl (von Werra et al., 2020) for training. Initially, we performed hyperparameter sweeps for $\beta = \{0.01, 0.05, 0.1, 0.5\}$ and max learning rate $= \{3e{-}7, 5e{-}6, 1e{-}6\}$ as initial exploration. We train LLAMA-3.1-TULU-3-8B-SFT with $\beta = 0.01$ and max learning rate $= 3e{-}7$ and train MISTRAL-7B-INSTRUCT-V0.2 with $\beta = 0.05$ and max learning rate $= 3e{-}7$ We employ 10% of steps as training warmup.

Evaluation Hyperparameter. We maintain a maximal generation length, 2048, for both models. The temperature for MISTRAL-7B-INSTRUCT-V0.2 is 0.7 and is 0.9 for LLAMA-3.1-TULU-3-8B-SFT when evaluating AlpacaEval 2. For Arena-Hard, the maximal length is 4098 for both models and we use default temperature 0.

A.3 Results on More Reward Models

In this part, we demonstrate the effectiveness of removing the most difficult k percent of prompts for LLAMA-3.1-TULU-3-8B-SFT on more reward models, ARMORM-LLAMA3-8B-V0.1 (Wang et al., 2024b) and LLAMA-3.1-8B-INSTRUCT-RM-RB2 (Malik et al., 2025). Specifically, we remove the most difficult 70% of prompt and evaluate the model after DPO training with AlpacaEval 2. The results are shown in Table 5. Our conclusions hold across different reward models.

Method	ArmoRM-Llama3-8B-v0.1		
	LC	WR	Length
full	36.39	39.13	2313
random	35.12	34.67	1933
our	**39.45**	**41.34**	2032
Method	**Llama-3.1-8B-Instruct-RM-RB2**		
	LC	WR	Length
full	37.59	37.83	2045
random	35.03	34.26	2044
our	**40.09**	**41.67**	2096

Table 5: We demonstrate the validity of removing hard prompts on more reward models, ARMORM-LLAMA3-8B-V0.1 and LLAMA-3.1-8B-INSTRUCT-RM-RB2.

A.4 Keeping Varying Portion of Easy Prompts for Mistral

We vary k from 10% to 40% and report the results in Figure 5. There is an increasing trend as k grows from 10% to 30%, with the peak performance observed at $k = 30\%$. Beyond this point, incorporating more difficult prompts begins to degrade performance.

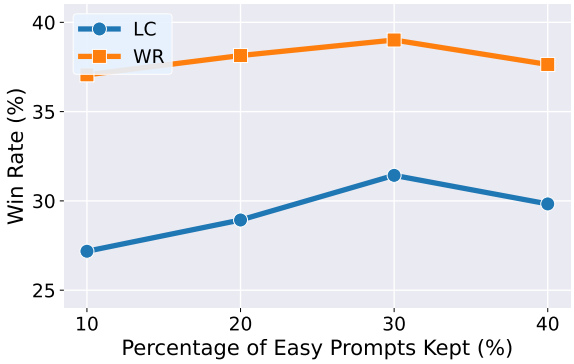

Figure 5: We present the performance of model on AlpacaEval 2 as we change k from 10 to 40 percent on Mistral. Performance first improve and then degrade as we incorporate more and more hard prompts. Performance reaches peak when we use only about 30% percent of easiest prompts.