Extended Abstract

If your preference is wrong, deconsolidate your belief and forget what you are confident in.

Motivation Aligning large language models (LLMs) with human preferences is typically done via reinforcement learning from human feedback (RLHF), but this process involves a complex, multi-stage training pipeline and can be unstable in practice. Direct Preference Optimization (DPO) was proposed as a simpler alternative that reframes preference alignment as a supervised learning task on human feedback pairs, eliminating the need for a reward model or on-policy reinforcement updates. While DPO is more stable and efficient than RLHF, it still has limitations: the objective rewards or penalizes outputs provided in the off-policy dataset, rather than the model's own responses. As a result, the model may not be explicitly discouraged from generating its own high-probability undesirable outputs, leaving room for improvement in alignment effectiveness.

Method We introduce Direct Preference and Penalization Optimization (DPPO), a hybrid framework that augments DPO with direct on-policy penalization. In DPPO, if the model votes for an losing sequence, the method identifies the model's *own* highest-probability response and explicitly penalizes it, rather than only penalizing the losing response from the dataset. This approach effectively teaches the model what *not* to say by directly reducing the likelihood of its most likely mistakes. DPPO implements this via three complementary training objectives: (1) a *minimum likelihood* loss that adds a penalty term to minimize the model's predicted probability of the sampled undesirable output; (2) an *unlikelihood* loss that applies a negative log-likelihood to known undesired response sequences; and (3) a *contrastive likelihood* loss that treats the reference likelihood as a baseline to be subtracted. These penalization losses are integrated with the standard DPO loss during fine-tuning. By providing direct on-policy negative feedback signals alongside off-policy preference reinforcement, DPPO addresses the mismatch of handling negative examples in vanilla preference optimization, giving the model both the "carrot" of reward and the "stick" of a much more precise and customized penalization.

Implementation First, a base model is obtained by supervised fine-tuning on the SmolTalk dataset. Next, we apply DPPO using UltraFeedback, a preference dataset with AI-generated scores, to further align the model with the desired outputs. During DPPO training, the model learns from the off-policy preference pairs (as in DPO) while also receiving on-policy penalties for its own most likely undesired rollouts. For evaluation, we compare the model's responses against those of the base model on test prompts; the reward model (Llama-3.1 Nemotron 70B) judges which response is better, and the percentage of prompts where the DPPO's output is preferred (win rate) is reported as the metric.

Results Our experiments demonstrate that DPPO achieves superior alignment performance with significantly improved data efficiency. Penalizing the model's own high-probability mistakes leads to large gains: notably, a model trained with the *unlikelihood* objective on only **8k** preference samples outperformed a standard DPO model trained on **60k** samples. Furthermore, a combined strategy—applying unlikelihood training followed by a DPO post-tuning yielded a win rate of **97.7%**, meaning the DPPO-aligned model won almost all comparisons against the baseline.

Discussion The result represents a substantial improvement in aligning the LLM's behavior with the preferred responses, achieved with far fewer preference data than baseline approaches. The significant gains from DPPO underscore the importance of actively penalizing a model's own likely mistakes as part of the alignment process. Our findings here resonate with the idea of the mixed use of off-policy data and on-policy rollouts. And the pre-collected responses for penalization avoided a huge computation burden.

Conclusion In summary, DPPO combines direct preference learning with targeted on-policy penalization. This hybrid approach yields a highly-aligned language model that attains nearly perfect preference compliance (a 97.7% win rate against the baseline) with substantially fewer training samples than prior methods. Our work demonstrates a promising direction for LLM alignment, moving beyond purely comparative feedback schemes by incorporating explicit on-policy negative examples to greatly improve data efficiency and outcome quality. Future extensions of DPPO could explore *self-curriculum learning*, wherein the model's own mistakes are iteratively collected and used as new penalization data in successive training rounds. Likewise, one could employ *dynamic penalization schedules* that adjust the strength of the penalty term over the course of training.

DPPO: Direct Preference and Penalization Optimization

Pengwei Sun
Program of Biomedical Physics
Stanford University
pengwei@stanford.edu

Abstract

Large language models (LLMs) are typically aligned with human preferences using reinforcement learning from human feedback (RLHF), but this process can be complex and unstable. This report explores an alternative approach called Direct Preference and Penalization Optimization (DPPO), which combines Direct Preference Optimization (DPO) with novel direct on-policy penalization techniques. We fine-tune an LLM using a supervised dataset of high-quality chat responses (SmolTalk) and use DPPO to align the model with a preference dataset of AI-generated feedback (UltraFeedback). The DPPO method extends DPO by not only training with off-policy prefered and undesired responses but also explicitly penalizing most likely on-policy responses when the model fails to vote for the preferred sequence. We implemented three training paradigms including minimum likelihood, unlikelihood, and contrastive likelihood. The models are evaluated with the Llama-3.1 Nemotron-70B reward model, reporting the win rate against the supervised fine-tuned base model. Our experiments demonstrate that penalization dramatically improves data efficiency and alignment performance. Notably, an unlikelihood training on only 8k preference samples outperforms a standard DPO trained on 60k samples, and a combined strategy (unlikelihood training plus DPO post-tuning) achieves a win rate of 97.7%, indicating a substantial advancement in aligning LLM behavior with preferred outputs. These results suggest that DPPO's blend of preference optimization and direct on-policy penalization yields superior performance in LLM alignment, offering a promising direction for future research.

1 Introduction

Large language models (LLMs) nowadays have been widely used in various applications, such as text generation, machine translation, and question answering. Minaee et al. (2025) They are usually trained on large corpora of data to learn the statistical properties of language, and then fine-tuned on specific downstream tasks for generalization. Aligning LLMs with human preferences is a central challenge in modern AI. Several reinforcement learning approaches have been used to address this gap. Reinforcement learning from human feedback (RLHF) gather human feedback on the quality of model outputs and fine-tune the LMs to better match these preferences. Ouyang et al. (2022) It requires a separately trained reward model to reflect human preferences, and then the LM is fine-tuned to maximize this reward. Recently, Direct Preference Optimization (DPO) has been proposed as a simpler alternative for preference-based fine-tuning. Rafailov et al. (2024) DPO reframes the problem as a straightforward supervised learning task: given a pair of model responses with a preference label, the model is fine-tuned via a classification-style loss to prefer the higher-quality response . This method eliminates the need for on-policy sampling during training and has been shown to achieve alignment comparable to or better than RLHF while being more stable to train and easier to implement.

Stanford CS224R 2025 Final Report

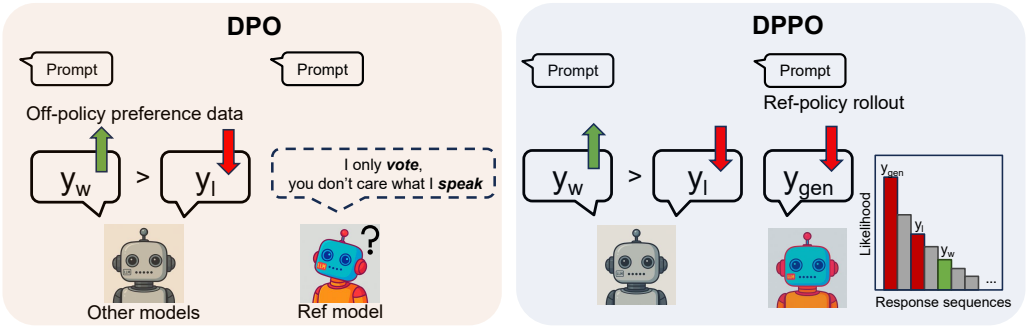

Figure 1: Method Overview. DPPO penalizes most likely generated response (y_{gen}) from the reference model to be optimized.

Despite DPO's advantages, there remains room for improvement in how preference data is used to steer LLMs. Notably, vanilla DPO focuses on rewarding preferred outputs and penalizing undesirable outputs. However, the winning and losing sequences in the dataset are often sampled from other external models. Although DPO votes for the winning or losing sequence when facing the preference data, the absolute likelihood to generate both sequences might not be as high. For example, when our model prefers the losing sequence rather the winning sequence, the preference indicates that our model does not behave well in response to the prompt. But penalizing the losing sequence sampled from other policies is not the most efficient and direct way to optimize our model. One possible solution to address this issue is to construct an on-policy preference dataset for each model to be optimized. However, collecting such high-quality human-preference data is expensive. Although integrated LLM scoring drastically decreases the cost for curating on-policy preference dataset and can be used to score the sampled rollouts from the model to be optimized, it cannot fully reach the level of human annotators. Barj and Sautory (2024) The question turns out to be, given an off-policy preference dataset, can we better leverage it to align our model with the human preference?

To address this gap, we explore the idea that direct penalization of most likely inferior responses from the model to be optimized itself can complement DPO. By explicitly sampling the most likely response from the reference model and reducing the likelihood of the sampled on-policy response when the model votes for the undesired off-policy sequences, the model can learn to avoid pitfalls more effectively than with preference-guided reward signals alone. We hypothesize that integrating a penalization objective with DPO will lead to stronger alignment (fewer bad outputs) and better data utilization, as the model receives a clearer and stronger customized signal about what not to say.

We propose Direct Preference and Penalization Optimization (DPPO), a training framework for LLM alignment that augments DPO with direct penalization losses. Our contributions include: (1) introducing three penalization-based training objectives (minimum likelihood, unlikelihood, and contrastive likelihood) to directly handle negative most likely feedback, (2) demonstrating improved performance and data efficiency – for example, showing that using a fraction of the preference data with penalization can outperform using the full data with standard methods, and (3) analyzing the effects of key hyperparameters like penalization strength and expectile threshold on the alignment outcome. The results highlight that direct on-policy penalization is a powerful complement to preference optimization, yielding nearly a 98% win rate against the baseline model's responses. We also discuss how this approach affects the model's voting behavior (its tendency to pick the better response).

2 Related Work

Our work builds on the DPO algorithm proposed by Rafailov et al. (2024). DPO eliminates the need for traditional RL in alignment by leveraging a clever reparameterization of the reward model. It fine-tunes a language model using a simple preference classification loss: given a prompt and two responses (one preferred, one dispreferred), the model's parameters are adjusted so that the log-likelihood of the preferred response is higher than that of the dispreferred response by a certain margin (derived in closed form). Essentially, the language model itself is treated "as secretly a reward

model," directly producing higher likelihood for better responses. The DPO loss is defined as:

$$\mathcal{L}_{\text{DPO}} = -\log \sigma(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}) \tag{1}$$

where y_w is the winning sequence, y_l is the losing sequence, π_θ is the model's predicted probability, and π_{ref} is the predicted probability from the reference model (the supervised fine-tuned model used as initialization here). In our experiments, we implement vanilla DPO as a baseline.

SimPO (Simple Preference Optimization) is a recently proposed variant of DPO that simplifies the RLHF fine-tuning pipeline. Meng et al. (2024) Unlike DPO, which relies on a fixed reference model to calibrate rewards, SimPO uses a reference-free reward defined by the average log-likelihood of the model's output sequence. In particular, for a given prompt and candidate response y, SimPO defines the reward as

$$r_\theta(y) = \frac{1}{|y|} \sum_{t=1}^{|y|} \log \pi_\theta(y_t \mid x, y_{<t}) \tag{2}$$

a length-normalized log-probability that directly aligns with the model's generation process. This design eliminates the need for any reference policy and ensures the reward metric is consistent with the model's actual decoding objective, addressing a known discrepancy in DPO between the training reward and the generation metric. Furthermore, SimPO augments the standard Bradley–Terry pairwise preference loss by introducing a target reward margin γ, which requires the preferred (winning) response to score at least γ higher in reward than the dispreferred (losing) response. In practice, SimPO's objective maximizes the likelihood $\sigma(r_\theta(y_w) - r_\theta(y_l) - \gamma)$ for each human-labeled pair (y_w, y_l), encouraging a clear separation between winning and losing responses. By forgoing the reference model and using a generation-aligned reward, SimPO offers a simpler and more efficient training pipeline. Empirically, these changes lead to better reward alignment and significant performance gains: SimPO consistently outperforms DPO on alignment benchmarks, demonstrating the benefits of its streamlined approach. We implement SimPO as another stronger baseline to compare.

Unlikelihood training is a method to decrease the likelihood of certain words or tokens which are called as negative candidates. Welleck et al. (2019) It is originally proposed to relief repetitive and dull text while maintaining perplexity. It forces unlikely generations to be assigned lower probability by the model. Compared to directly minimizing the likelihood of negative candidates, unlikelihood training has lower risk of gradient explosion. The unlikelihood loss is defined as:

$$\mathcal{L}_{\text{unlikelihood}} = -\sum_{t=1}^{T} \log(1 - \pi_\theta(y_t|y_{<t}, x)) \tag{3}$$

3 Method

The vanilla DPO model is trained to maximize the probability of the winning sequence while minimizing the probability of the losing sequence. However, the binary preference dataset is usually off-policy, meaning that the losing sequence is not sampled from the model to be optimized, but rather from external models when constructing the dataset. The optimized model may not tends to generate the losing sequences with high probability. Thus penalizing the model's predicted probability on the losing sequence is not effective during training. To address this issue, we propose three methods to directly penalize the model's most likely output when the model loses (i.e. prefers the losing sequence). The intuition is that if the model does not match the preference in response to some prompt, it prefers the losing sequence over the winning sequence. But at the same time, it might prefer the most likely output over the losing sequence a lot. So the most efficient way to penalize the model is to directly penalize the model's most likely output, rather than the losing sequence output by other models. We propose three methods to directly penalize the model's most likely output when the model prefers the losing sequence.

3.1 Minimum likelihood training

First, we greedily sample the most likely output from the reference model given the prompt x:

$$y_{\text{gen}} = \arg\max_y \pi_{\text{ref}}(y|x) \tag{4}$$

Then we can directly penalize the model's predicted probability on this generated sequence y_{gen} when the model prefers the losing sequence.

$$\mathcal{L}_{\text{minll}} = \mathcal{L}_{\text{DPO}} + \alpha \cdot f(r) \log \pi_\theta(y_{\text{gen}}|x) \tag{5}$$

where α is a hyperparameter to control the strength of the penalty, r is the reward defined as the log ratio between the likelihood of the winning sequence over the losing one (6). $f(r)$ is a reward-weighted function that controls the strength of the penalty based on the reward r (7).

The reward r is calculated as below and then normalized to be in the range of $[-1, 1]$:

$$r = \log \frac{\pi_\theta(y_w|x)}{\pi_\theta(y_l|x)} \tag{6}$$

We use four forms of $f(r)$ in different training paradigms:

$$f(r) = \begin{cases} -\mathbb{1}\{r < 0\} r & \text{linear weight function} \\ \mathbb{1}\{r < 0\} r^2 & \text{quadratic weight function} \\ \mathbb{1}\{r < 0\} (-r)^{0.5} & \text{square root weight function} \\ (\mathbb{1}\{r < 0\} - \zeta) r^2 & \text{expectile weight function} \end{cases} \tag{7}$$

If $r > 0$, the model prefers the winning sequence over the losing sequence, and the training loss degenerates to be the vanilla DPO loss (except for the expectile weight function, which reserves to reward the model's most likely output). When $r < 0$, the model prefers the losing sequence over the winning sequence, and we penalize the model's predicted probability of the most likely output y_{gen}. The penalization loss is scaled with the reward r through the reward weight function.

3.2 Unlikelihood training

Inspired from the unlikelihood loss function Welleck et al. (2019), we propose the unlikelihood training loss:

$$\mathcal{L}_{\text{unll}} = \mathcal{L}_{\text{DPO}} - \alpha \cdot f(r) \log(1 - \pi_\theta(y_{\text{gen}}|x)) \tag{8}$$

Compared to directly minimizing the likelihood of y_{gen}, unlikelihood loss uses a simple math transformation to stabilize the training procedure.

3.3 Contrastive likelihood training

In the contrastive likelihood training, we modify the classification term in the DPO loss. While minimizing the likelihood of y_{gen}, we subtract the reference baseline $\pi_{\text{ref}}(y_{\text{gen}}|x)$, which constrains the model not diverging too far from the reference model while optimization. The new classification loss is defined as:

$$\mathcal{L}_{\text{conll}} = -\log \sigma(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} - \alpha\beta \cdot f(r) \log \frac{\pi_\theta(y_{\text{gen}}|x)}{\pi_{\text{ref}}(y_{\text{gen}}|x)}) \tag{9}$$

4 Experiment

Dataset For SmolTalk (Allal et al. (2025)), we use one-turn conversation between "user" and "assistant", truncate the context token length to 576, which is the 95th percentile of token lengths in the training set. For Ultrafeedback (Dubois et al. (2024)), we truncate the context token length to 989, the 95th percentile.

Evaluation We use a parametric reward model for scoring with the Llama 3.1 Nemotron 70B Reward Model. The prompts and responses are constructed as a chat template. Nemotron generates a reward score for both reference model (our SFT model) and the evaluated model. We report the win rate across the evaluation set where "win" means the reward of the evaluated model is higher than the reference model.

Training details To train the SFT model, we finetune the base model (Qwen 2.5 0.5B Base) for one epoch with a batch size of 4, gradient accumulation step of 8, and learning rate of 10^{-6}. To train the DPO model, we screen the β and determine that $\beta = 0.01$ has the best performance. We use the same hyperparameters for DPPO training. All trainings are finished on a 48 GB NVIDIA RTX 6000 Ada GPU.

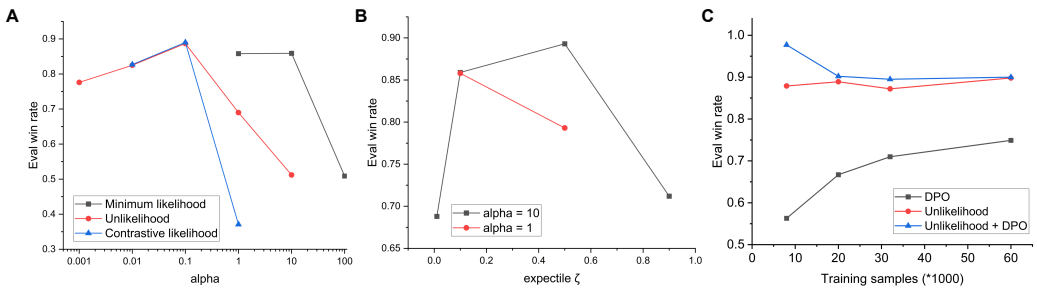

Figure 2: **A**. Model performance with different penalization strength α. **B**. Minimum likelihood model performance with different expectile ζ. **C**. Data efficiency of different training paradigms. For "Unlikelihood + DPO", the model is trained with unlikelihood loss with 60k samples, and then post-trained with different numbers of training samples using DPO loss.

5 Results

We first perform the hyperparameter search on the penalizing strength α and expectile ζ in different training paradigm (Fig. 2). α is a critical hyperparameter because it controls the strength of the penalization we apply on the model. Consistent with expectations, too low an α yields only marginal gains over DPO since the penalization is weak, while too high an α can lead to unstable training or declines in quality. The model might start chasing the reward model too aggressively, possibly overfitting to avoiding certain phrases at the expense of naturalness. The necessity of balancing these is why DPPO can be seen as a generalization of DPO: with $\alpha = 0$, DPPO degenerates to standard DPO, and as α increases, we gradually put more emphasis on "what not to say." We choose $\alpha = 10$ as the sweet spot for minimum likelihood method and 0.1 for unlikelihood and contrastive likelihood methods.

We test the expectile reward weight function in minimum likelihood training paradigm. $\zeta = 0.5$ has higher evaluation win rate when $\alpha = 10$ and $\zeta = 0.1$ is better when $\alpha = 1$. The expectile function is symmetric when $\zeta = 0.5$, leading to balanced penalization when the reward $r < 0$ and rewarding when $r > 0$. While the expectile function is skewed when $\zeta = 0.1$, prioritizing the penalization rather rewarding on y_{gen}.

The evaluation win rates of different models are shown in Fig. 3A. Compared to the SFT reference model, the DPO model achieves a win rate of 0.759, indicating that it generates responses that are more aligned with human preferences. Minimizing likelihood, unlikelihood, and contrastive likelihood training achieve win rates of 0.893, 0.898, 0.890, respectively. All our methods outperform the vanilla DPO model on the evaluation set.

Our methods have higher data efficiency than the vanilla DPO model, as shown in Fig. 2C. There are 60k samples in the training set, and we can achieve a win rate of 0.879 with only 8k samples using unlikelihood training, while DPO requires one full epoch to achieve a win rate of 0.749. The reason behind the gap is that the winning and losing sequences in the preference dataset are not directly sampled from the model to be optimized, but rather from external models. It indicates that the off-policy preference datasets are redundant and the training needs to be customized for each model.

Given partially use of training samples, we perform a vanilla DPO post-training after our training strategies. The evaluation win rates are reported as the dark green bar in Fig. 3A. DPO post-training improves the performance of unlikelihood training further, achieving 0.977 win rate over the SFT baseline.

We further investigate the voting behaviors of different models in Fig. 3B. DPO model successfully increases the number of positive reward, while contrained by the reference model, the distribution shift is limited. Our methods further decrease the number of negative rewards and slightly increase the positive rewards, resulting a skewed reward distribution and aligned preference.

We also implement SimPO Meng et al. (2024), a variant and extension of the vanilla DPO method as another baseline model for comparison. We screen the hyperparameters γ and β in the SimPO loss

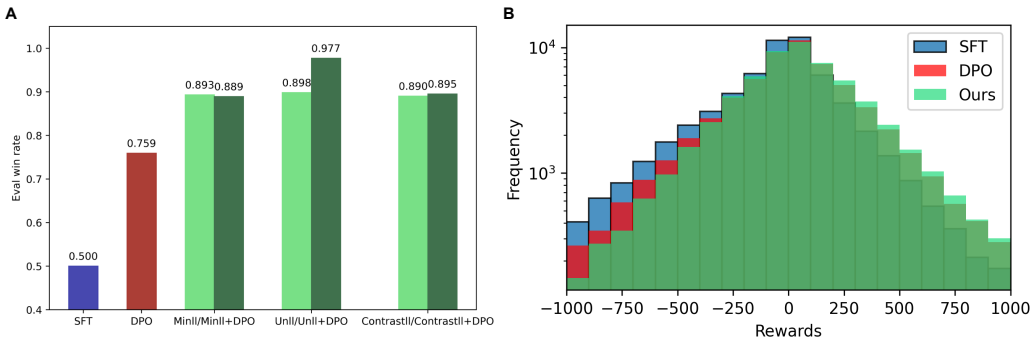

Figure 3: **A**. Model performance of different training paradigms. **B**. DPO reward distribution of different models. "Ours" refers to the unlikelihood training with DPO post-training.

Table 1: Evaluation win rate of SimPO

γ	β	Eval win rate
1	2.5	0.688
1	2	0.631
3	10	0.542
5	10	0.637

function, and report the evaluation win rate in Table 1. Notably, with a non-thorough hyperparameter tuning, SimPO is even performing worse than the vanilla DPO. On the one hand, we have limited resources in this course to tune the hyperparameters for SimPO. On the other hand, laborious searching process impedes the use and generalization of SimPO method when facing various models and datasets.

6 Discussion

The above results demonstrate that on-policy penalization is a powerful tool for preference optimization. One of the clearest effects of our penalization methods was the significant decrease in frequency of negative reward outcomes. In the context of alignment, a negative reward outcome means the fine-tuned model votes for an undesired response. These are the scenarios where alignment truly fails – the model might produce irrelevant, incorrect, or undesired content. By incorporating negative examples from the model itself in training (and not just off-policy comparisons), the model learns from its own mistakes. This is analogous to learning in humans: we often learn better from failures after we forget what we feel confident in (but it is actually wrong). The penalization loss provides that feedback by explicitly saying -

Your preference is wrong, deconsolidate your belief and make it less likely.

Consequently, the model's overall quality improved not just in average sense, but in a worst-case sense: the bad tails of the performance distribution were trimmed. This is important for user-facing AI systems where even a small probability of a very bad output can be unacceptable.

The result that 8k penalized examples beat 60k plain examples is telling. It suggests that the information content in a penalization signal is quite high. Each penalized example teaches a specific lesson, "avoid doing X in response to this prompt", which might be broadly applicable. In contrast, a preference comparison without penalization just says "Y is better than Z", which the model may internalize as "mimic Y and avoid Z", while Y and Z are not outputs from the model itself. Penalization makes the "avoid X" lesson explicit. Therefore, even with far fewer training samples, the model can generalize that knowledge to avoid a wide class of bad outputs. This partially explains the data efficiency gain – the model is not just memorizing which answer is better in each specific case, but learning a general rule to not produce certain patterns that are generally low-quality. In

reinforcement learning terms, penalization provides a form of negative reward for certain actions, which can accelerate learning of an optimal policy especially when those actions were previously tempting local optima. Our findings here resonate with the idea of the mixed use of off-policy data and on-policy rollouts – we took a fixed preference dataset and generated reference model outputs, and leveraged them to learn a better policy without needing to explore a lot more. From a practical standpoint, our use of pre-collected on-policy responses for penalization avoided a huge computation burden and still achieved great results. It opens up thinking about how future alignment training could first gather a pool of "likely mistakes" and then specifically train the model to avoid them, rather than relying on pure pairwise preference data.

Limitations The evaluation method is based on a parametric reward model, which may not accurately reflect real human preferences. There might be subtle areas where the DPPO model overfits to the reward model's preferences (for instance, maybe it learned to write in a style that Nemotron particularly likes, which might not always coincide with human preference). Some other evaluation metrics are called for, such as perplexity, diversity, and human evaluation to better understand the model's performance from different aspects.

7 Conclusion

We presented Direct Preference and Penalization Optimization (DPPO), a novel approach to fine-tune language models with preference data that leverages direct penalization of most likely outputs. By integrating penalization losses (minimum likelihood, unlikelihood, contrastive likelihood) with the standard DPO framework, DPPO addresses a key shortcoming of vanilla preference optimization – the mismatch of negative feedback. Our experiments, conducted on an open-source LLM with high-quality datasets (SmolTalk and UltraFeedback) and evaluated by a state-of-the-art reward model, demonstrated substantial gains. DPPO improved the alignment win rate to an impressive 0.977 under the best configuration with superior data efficiency. These results highlight that besides combining "push" and "pull" forces in training (rewarding good outputs and penalizing bad ones), a precise on-policy penalization can dramatically enhance learning from human or AI feedback. It provides a viable path to move beyond purely comparative training regimes and incorporate richer feedback signals.

There are several avenues for future work. First, one could explore automating the penalization data collection: for instance, using an iterative approach where the model's mistakes at one stage become the penalization targets for the next stage (a sort of self-curriculum). This might further improve performance or efficiency. Second, the balance between preference optimization and penalization could be dynamically adjusted during training (rather than a fixed α) – perhaps starting with a high penalization to quickly eliminate bad behaviors, then tapering it down as the model improves, to fine-tune the remaining subtle preferences. This schedule might achieve even better results. Another direction is to investigate DPPO on safety-critical alignment tasks. For example, models often exhibit biases or can generate toxic content. A penalization approach could be directly applied to known unsafe outputs (with human or model identification of such outputs) to train the model out of those failure modes. Similarly, penalization might help mitigate issues like model hallucinations by explicitly penalizing known incorrect statements if identified by a fact-checker model.

In summary, DPPO expands the toolkit for aligning language models by showing the value of direct penalization. It reinforces the intuitive idea that to teach a model, one should provide both the carrot and the stick. Our findings demonstrate that the judicious and precise use of the "stick" – in the form of penalizing low-reward, high-likely behaviors – can significantly accelerate and strengthen the alignment of LLMs. We hope this work inspires further research into hybrid optimization strategies that combine the strengths of supervised learning, preference learning, and negative example training to create AI systems that are not only smart and useful, but also safe and aligned with human values.

8 Team Contributions

This is a solo project.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. 2025. SmolLM2: When Smol Goes Big – Data-Centric Training of a Small Language Model. arXiv:2502.02737 [cs.CL] https://arxiv.org/abs/2502.02737

Houda Nait El Barj and Theophile Sautory. 2024. Reinforcement Learning from LLM Feedback to Counteract Goal Misgeneralization. arXiv:2401.07181 [cs.LG] https://arxiv.org/abs/2401.07181

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2024. AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback. arXiv:2305.14387 [cs.LG] https://arxiv.org/abs/2305.14387

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. SimPO: Simple Preference Optimization with a Reference-Free Reward. arXiv:2405.14734 [cs.CL] https://arxiv.org/abs/2405.14734

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2025. Large Language Models: A Survey. arXiv:2402.06196 [cs.CL] https://arxiv.org/abs/2402.06196

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. 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 Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. arXiv:2305.18290 [cs.LG] https://arxiv.org/abs/2305.18290

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural Text Generation with Unlikelihood Training. arXiv:1908.04319 [cs.LG] https://arxiv.org/abs/1908.04319