SELF-IMPROVING ROBUST PREFERENCE OPTIMIZATION

Anonymous authors
Paper under double-blind review

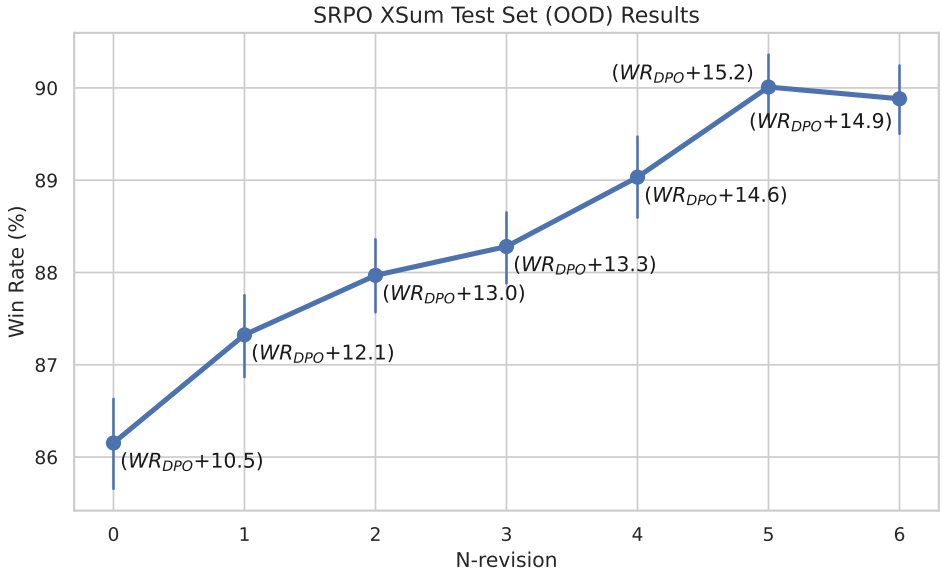


Figure 1: Self-improvement by SRPO in terms of win rates against human (*WR*). We demonstrate robustness by training on TL;DR and evaluating on XSum. Gains on Direct Preference Optimization (DPO) are reported in text captions.

ABSTRACT

Both online and offline RLHF methods such as PPO and DPO have been extremely successful in aligning AI with human preferences. Despite their success, the existing methods suffer from some fundamental limitations: prominent among those limitations are **(a)** models trained with RLHF can learn from mistakes or negative examples through RL mechanism or contrastive loss at the time of training. However at the time of inference they are not equipped with an innate mechanism to correct mistakes by self-improvement. **(b)** The optimal solution of existing methods is highly task-dependent and thus it is difficult for them to generalize to new tasks. Here we propose Self-Improving Robust Preference Optimization (SRPO), a practical and mathematically principled offline RLHF framework that address both these challenges. The key idea of SRPO is to cast the problem of learning from human preferences as a self-improvement process, which can be mathematically expressed in terms of a min-max objective that aims at joint optimization of self-improvement policy and the generative policy in an adversarial fashion. The solution for this optimization problem is independent of the training task and thus it is robust to its changes. We then show that this objective can be re-expressed in the form of a non-adversarial offline loss which can be optimized using standard supervised optimization techniques at scale. We show the effectiveness of SRPO in terms of AI Win-Rate (WR) against human (GOLD) completions. In particular, when SRPO is evaluated on the XSUM dataset, it outperforms the celebrated DPO by a clear margin of **15**% after 5 self-revisions, achieving WR of **90**%. Also when evaluated on the challenging Arena-Hard prompts, SRPO outperforms both DPO and IPO (by **4**% without revision and **6**% after just one revision), achieving a WR of **56**% against `Llama-3.1-8B-Instruct`.

1 INTRODUCTION

Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017) has rapidly become a standard method to align Large Language Models (LLMs). The main advantage of RLHF is that it allows the model to learn from its mistakes. So RLHF-tuned model not only learns from *good* experiences but also learns from *bad* experiences and mistakes. Despite being able to learn from mistakes, the standard RLHF-tuned models are not equipped with an innate mechanism to self-improve and correct their mistakes at the time of inference (Kumar et al., 2024). So if the model makes a critical mistake or miss a crucial fact at the time of inference It would be very difficult for the model to recover from those mistakes. Another practical issue that all the prominent existing RLHF methods (offline or online) (Ouyang et al., 2022; Rafailov et al., 2023; Azar et al., 2023; Zhao et al., 2023b; Ahmadian et al., 2024) encounter is that their optimal solution heavily depends on the training task in terms of the distribution used to generate the preference data (*behavior policy*) (Munos et al., 2023; Azar et al., 2023). This makes the existing RLHF methods vulnerable to the cases in which the evaluation distribution is significantly different from that of the behavior policy (Li et al., 2024b; Kirk et al., 2024).

To address these challenges, we introduce an alternative approach for aligning LLMs from human preferences based on more principled and robust foundations. Our goal is to find a solution that is robust to the changes in the preference dataset, meaning that changes in the distribution from which the completions are sampled do not affect the final outcome of learning significantly. To achieve this goal, we exploit the concept of self-improving (Huang et al., 2022; Bai et al., 2022) language models. By self-improving LLM we refer to a model capable of enhancing its outputs recursively with each inference iteration. Our **S**elf-Improving **R**obust **P**reference **O**ptimization (SRPO) consists of two back-to-back optimization processes:

(Step 1) In-Context Self-Improving Preference Optimization: The core idea is to learn an in-context self-improving model $\pi_\dagger$:[1] given an *in-context* completion y and a context x, the self-improvement model, $\pi_\dagger$, outputs an improved completion y' with probability $\pi_\dagger(y'|y, x)$ from which sampled completions are most preferred to completion y according to the human preference model p. As explained later, it turns out that this problem, in its KL-regularized form, can be expressed as a well-defined preference optimization problem and solved analytically. Furthermore, the solution can be estimated through a supervised direct preference optimization scheme similar to the approach used by Rafailov et al. (2023) and Azar et al. (2023).

(Step 2) Robust Preference Optimization of Generative Model: The next step is to exploit the self-improvement policy learned in the previous step to learn a generative LLM, π. The key idea here is that the best generative policy can be identified as a policy that generates completions requiring minimal improvement using the optimal self-improvement policy $\pi_\dagger$ derived in step 1. This goal can be achieved by minimizing the objective of step 1 with respect to the generative policy for *in-context* completions, y. Similar to step 1, this problem, in its KL-regularized form, can also be solved analytically in terms of the optimal improvement policy $\pi_\dagger$ and the optimal generative policy π. More significantly, we show that the solution for steps 1 and 2 can be estimated jointly through a single supervised direct preference optimization scheme using only a dataset of annotated pair-wise completions. Thus, one can solve both for the self-improvement policy $\pi_\dagger$ and π by minimizing the supervised learning objective of SRPO. Unlike existing RLHF methods, this solution is independent of the behavior policy and is therefore robust to its changes.

As using the self-improvement model in SRPO is a significant departure from the existing paradigm for RLHF, we provide a high-level motivation for it in Sec. 2. We then formalize our objective for SRPO in Sec. 3, allowing for the joint optimization of both $\pi_\dagger$ and π by optimizing an adversarial min-max objective. In Sec. 4 we present our main algorithmic/mathematical contribution: we prove that the preference probability p can be expressed in terms of the log-likelihoods of the optimal self-improvement policy $\pi^*_\dagger$ and the log-likelihoods of the optimal robust generative policy π^*. This theoretical finding is the key result for SRPO: solving this system of equations through least-squares regression provides us with the practical supervised SRPO objective that solves for both policy and robust generative policy through a single supervised objective without any need for reward model or online inference. Our key theoretical finding is similar to the main result of DPO (Rafailov et al.,

[1] From now on, a generative LLM will be considered as equivalent to a distribution or policy π from which we can sample completions y with probability $\pi(y|x)$, where x is the context or prompt.

2023) in that both express preference probabilities in terms of the optimal policy. However, DPO result only holds when preference probabilities conform to Bradly-Terry model (Bradley & Terry, 1952), whereas our key result is general as it holds across all preference models. In Sec. 5 we further illustrate our argument on the robustness of SRPO by providing an in-depth analysis of the solution of SRPO and other direct preference optimization methods. We also showcase/analyze the robustness of SRPO on a simple synthetic example. Finally in Sec. 6 we conduct large-scale experiments on training LLMs with SRPO both on in-distribution and OOD summarization tasks and we compare the results with those of standard baselines.

2 Learning Self-Improvement Policy Through Preference Optimization

The goal of this section is to motivate why learning self-improvement models through preference optimization can be useful for training a GPT-style language model. One of the main challenges that affects the performance of sequence-to-sequence GPT-style models is that it is hard for them to recover from failure at the time of inference. This problem which is known as *exposure bias* in the literature of sequence-to-sequence prediction (Xu et al., 2020; Arora et al., 2023; Bengio et al., 2015), is due to the fact that at the time of inference GPT-style models generate tokens step by step based on their own previous outputs not the ground truth. This can lead to error accumulation: if the model generates an incorrect token early in the sequence (or a misleading token that may lead to an error later down the line), it may propagate errors since its subsequent predictions are based on the earlier, incorrect (or misleading) token (Arora et al., 2023).

To address this shortcoming of GPT-style models one may fine-tune the model such that it can revise its incorrect generations through self-improvement/self-refinement process (Madaan et al., 2024; Hu et al., 2024; Huang et al., 2022; Bai et al., 2022). The simplest way to train such a self-improvement model is through supervised fine tuning (SFT) (Bai et al., 2022; Wei et al., 2023). To train self-improvement model through SFT one first need to build a supervised dataset of (x, y, y^*) in which y is some *initial* completion than can be correct or incorrect and y^* is the corrected version of y. The improvement model can then be fine-tuned using SFT to predict y^* from the pair (x, y).

However training a SFT self-improvement model is a challenging task for the following reasons.

1. Creating a supervised dataset of (x, y, y^*) is resource intensive and hard to scale (Huang et al., 2022). Also the existing standard SFT datasets are not created in this way.

2. Moreover SFT training pipeline for self-improvement can simply overfit to predict the correct answer y^* only from the context x and ignore the initial answer y altogether or over-fit to specific refinement pattern (Li et al., 2024a). Thus it may not learn properly how to improve an incorrect y to the correct y^*.

Alternatively one might ask whether one can generalize the direct preference optimization methods (e.g., Rafailov et al., 2023; Azar et al., 2023) for learning self-improvement models using the standard human preference datasets. To answer this question we first try to answer a more fundamental question:

What is the best use of human preference data?

To address this question, we observe that human preferences inherently encode valuable information about the relationship between more-preferred and less-preferred completions. This relational information can be leveraged to refine less-preferred completions, guiding them closer to those that better align with human preferences. In essence, this process involves learning a model of alignment mechanics—principles or "rules" that dictate how completions can be iteratively improved in terms of satisfying human preferences. This approach represents a more natural and intuitive learning task compared to directly predicting the highest-preferred completion, which is the focus of standard RLHF methods. Directly learning the optimal completion can be a challenging task, particularly when the completion space encompasses the full complexity of human language. Furthermore, the highest-preferred answer is unlikely to exist in the dataset, especially when completions are generated by large language models (LLMs) that inherently fall short of human-level quality.

Instead, it is more effective to model the improvement process itself: given a query x and an initial completion y, the goal is to predict an improved completion y'. This shift in perspective transforms the task into learning how to enhance subpar outputs from an LLM, rather than aiming to directly generate perfect responses. By capturing the underlying principles of human preference, the model can iteratively refine suboptimal completions y, progressively steering them towards the ideal outcome y^*. This framework not only aligns with the iterative nature of human reasoning but also facilitates continuous self-improvement, making it a robust and scalable approach for enhancing the performance of LLMs. In the following, we show how the self-improvement policy, $\pi_\dagger$, is trained alongside the generative policy, π, using pair-wise preferences.

3 SRPO OBJECTIVE

We start by introducing some notations required for establishing our theoretical results. Let x and y denote a context and a completion drawn from the space of all possible contexts $\mathcal{X}$ and all possible completions $\mathcal{Y}$, respectively. The large language model (LLM) is represented by the probability distribution (policy) π where $\pi(y|x)$ denotes the probability of the completion y given the context x. In the remainder of this article, we consider three variants of this base LLM, the trainable model π_{train} (for which we use the short-hand π), the reference model π_{ref} and the behavior model μ from which the completions in pair-wise preference dataset is sampled.

We also introduce the self-improvement $\pi(y'|y, x)$ as a model that using a context x and in-context completion (thought) y aims at improving y to a better completion y'. Similar to base LLM, we can define a reference model $\pi_{\text{ref}}(y'|y, x)$ also for the self-improvement model. Let $\mathcal{D} = \{x, y_1, y_2\}$ be a dataset of contexts and completions where y_1 and y_2 are drawn independently from $\mu(\cdot|x)$. We then present every pair y_1, y_2 to human annotators who express preferences for one of the completions, denoted as $y_w \succ y_l$ where y_w and y_l denote the preferred and dis-preferred actions amongst $\{y_1, y_2\}$ respectively. We then write true human preference $p(y_1 \succ y_2|x)$ the probability of y_1 being preferred to y_2 knowing the context x. The probability comes from the randomness of the choice of the human we ask for their preference. So $p(y_1 \succ y_2|x) = \mathbb{E}_h[\mathbf{1}\{h \text{ prefers } y_1 \text{ to } y_2 \text{ given } x\}]$, where the expectation is over humans h.

Consider a reference policy π_{ref}, and a real positive regularization parameter $\beta \in \mathbb{R}_+^*$. Then, we define the Self-Improving Robust Preference Optimization objective (SRPO) for every context x as

$$J^*(x) = \min_{\pi} \max_{\pi_\dagger} \mathop{\mathbb{E}}_{\substack{y_1 \sim \pi(.|x) \\ y_2 \sim \pi_\dagger(\cdot|y_1, x)}} \left[p(y_2 \succ y_1|x) - \beta D_{\text{KL}}(\pi_\dagger || \pi_{\text{ref}}|y_1, x) + \beta D_{\text{KL}}(\pi || \pi_{\text{ref}}|x) \right], \quad (1)$$

with the KL-regularization terms are defined as: $D_{\text{KL}}(\pi_\dagger || \pi_{\text{ref}}|y_1, x) = \text{KL}(\pi_\dagger(\cdot|y_1, x)||\pi_{\text{ref}}(\cdot|y_1, x))$ and $D_{\text{KL}}(\pi||\pi_{\text{ref}}|x) = \text{KL}(\pi(\cdot|x)||\pi_{\text{ref}}(\cdot|x))$.

In nutshell, this objective aims at **(i)** finding the best self-improvement policy $\pi_\dagger^*$ that improves every $y_1 \sim \pi$ optimally w.r.t. the preference distribution p, i.e., the improved policy is most preferred to y_1, while keeping $\pi_\dagger^*$ close to the reference policy π_{ref}, **(ii)** minimizing the same objective to find the best (robust) policy π^* for which the generated completions can be only minimally improved by the optimal self-improvement model $\pi_\dagger^*$. The min-max nature of this objective makes self-improvement effective for all policies close to π_{ref} as we are optimizing $\pi_\dagger$ in the worst-case scenario.

4 OFFLINE SOLUTION FOR OPTIMIZING SRPO OBJECTIVE

In this section we show that the min-max objective of Eq. (1) can be transformed to a non-adversarial offline supervised loss. Thus it can be optimized at scale using standard optimization techniques.

4.1 MAIN RESULT

The optimization problem of Eq. (1) is a non-trivial optimization problem that often requires solving a two-stage adversarial optimization problem through the game-theoretic approaches, which are often challenging and difficult to scale up, (see e.g., Munos et al., 2023; Rosset et al., 2024; Calandriello et al., 2024, for how we can use game-theoretic approaches/objectives to train LLMs). Here, inspired

by Rafailov et al. (2023); Azar et al. (2023), we aim at casting this complex optimization objective as a standard supervised learning problem that can be solved at scale given an offline pairwise preference dataset. The main theoretical result of our work is then as follows

Theorem 1. *Given a context x and the behavior policy $\mu(\cdot|x)$ let $\mu(y|x)$ and scalars $\alpha \in [0,1]$, $\beta > 0$ we have that the solution of the min-max objective of Eq. (1) is obtained by minimizing the following loss*

$$L_\alpha(\pi, \pi_\dagger) = (1-\alpha)L(\pi, \pi_\dagger) + \alpha L_\dagger(\pi_\dagger), \tag{2}$$

where $L(\pi, \pi_\dagger)$ and $L_\dagger(\pi_\dagger)$ are defined respectively as follows

$$L_\dagger(\pi_\dagger) = \mathop{\mathbb{E}}_{\substack{y_1, y_2 \sim \mu(\cdot|x) \\ x \sim \rho}} \left[p(y_2 \succ y_1|x) - \frac{1}{2} - \beta \left[\log\left(\frac{\pi_\dagger(y_2|y_1,x)}{\pi_{ref}(y_2|y_1,x)}\right) - \log\left(\frac{\pi_\dagger(y_1|y_1,x)}{\pi_{ref}(y_1|y_1,x)}\right)\right]\right]^2. \tag{3}$$

and

$$L(\pi, \pi_\dagger) = \mathop{\mathbb{E}}_{\substack{y_1, y_2 \sim \mu(\cdot|x) \\ x \sim \rho}} \left[p(y_2 \succ y_1|x) - \frac{1}{2} - \frac{\beta}{2} \left[\log\left(\frac{\pi_\dagger(y_2|y_1,x)}{\pi_{ref}(y_2|y_1,x)}\right) + \log\left(\frac{\pi(y_1|x)}{\pi_{ref}(y_1|x)}\right) \right.\right. \tag{4}$$
$$\left.\left. - \left(\log\left(\frac{\pi_\dagger(y_1|y_2,x)}{\pi_{ref}(y_1|y_2,x)}\right) + \log\left(\frac{\pi(y_2|x)}{\pi_{ref}(y_2|x)}\right)\right)\right]\right]^2.$$

4.2 Proof of Main Result

To prove the main result we first notice that the inner-maximization in the objective function of Eq. (1) is an instantiation of KL-regularized RL (Todorov, 2006). Thus it can be solved in analytical form and its solution is given by

$$\pi^*_\dagger(y_2|y_1,x) = \frac{\exp\left(\frac{p(y_2 \succ y_1|x)}{\beta}\right)\pi_{\text{ref}}(y_2|y_1,x)}{Z^*(y_1,x)}, \tag{5}$$

where $Z^*(y_1,x)$ is the normalization factor. One can easily show that by plugging $\pi^*_\dagger$ in the objective function of Eq. (1) we obtain:

$$J^*(x) = \min_\pi \mathbb{E}_{y_1 \sim \pi(.|x)} \left[\beta(\log(Z^*(y_1,x)) + D_{\text{KL}}(\pi||\pi_{\text{ref}}|x))\right]. \tag{6}$$

Now by solving Eq. (5) with respect to $p(y_2 \succ y_1|x)$ we obtain

$$p(y_2 \succ y_1|x) = \beta(\log(\pi^*_\dagger(y_2|y_1,x)) - \log(\pi_{\text{ref}}(y_2|y_1,x)) + \beta\log(Z^*(y_1,x))). \tag{7}$$

4.2.1 Optimizing the Self-Improvement Policy $\pi_\dagger$

We now prove that minimizing the objective of Eq. (3) gives us the optimal self-improvement policy (solution to maximization in the objective of Eq. (1)). We first notice that using the convention $p(y_1 \succ y_1|x) = \frac{1}{2}$ Eq. (7) implies

$$\frac{1}{2} = \beta(\log(\pi^*_\dagger(y_1|y_1,x)) - \log(\pi_{\text{ref}}(y_1|y_1,x))) + \beta\log(Z^*(y_1,x)). \tag{8}$$

Now by subtracting Eq. (7) from Eq. (8) we derive

$$p(y_2 \succ y_1|x) = \frac{1}{2} + \beta\left[\log\left(\frac{\pi^*_\dagger(y_2|y_1,x)}{\pi_{\text{ref}}(y_2|y_1,x)}\right) - \log\left(\frac{\pi^*_\dagger(y_1|y_1,x)}{\pi_{\text{ref}}(y_1|y_1,x)}\right)\right]. \tag{9}$$

This is our first key result that express preference $p(y_2 \succ y_1|x)$ in terms of the optimal self-improvement policy $\pi^*_\dagger$. So we enforce this equation for all y_1 and y_2 through minimizing ℓ_2 loss of Eq. (3) which concludes the proof for the self-improvement policy.

4.2.2 JOINT OPTIMIZATION OF THE ROBUST GENERATIVE POLICY π AND THE IMPROVEMENT POLICY $\pi_\dagger$

In this section, we show that to optimize the min-max objective of Eq. (1) (i.e., solving for π and $\pi_\dagger$) one can optimize the loss of Eq. (4). We start by collecting terms in Eq. (8) which implies $\beta \log(Z^*(y_1, x)) = \beta(\log(\pi_{\text{ref}}(y_1|y_1, x)) - \log(\pi^*_\dagger(y_1|y_1, x))) - \frac{1}{2}$. Thus, the objective of Eq. (6) can be expressed in terms of $\log(\pi^*_\dagger(y_1|y_1, x))$ (up to an additive and multiplicative constant) as follows:

$$J^*(x) \propto \min_\pi \mathbb{E}_{y_1 \sim \pi(.|x)} \left[\log \left(\frac{\pi_{\text{ref}}(y_1|y_1, x)}{\pi^*_\dagger(y_1|y_1, x)} \right) + D_{\text{KL}}(\pi || \pi_{\text{ref}}|x)) \right]. \tag{10}$$

Solving this objective with respect to π we obtain:

$$\pi^*(y|x) = \frac{\frac{\pi_{\text{ref}}(y|x)}{\pi_{\text{ref}}(y|y,x)} \pi^*_\dagger(y|y, x)}{Z^*(x)} \tag{11}$$

where $Z^*(x)$ is the normalization factor. Again by taking the logarithm from both side we obtain $\log(\pi^*(y|x)) = \log \left(\frac{\pi_{\text{ref}}(y|x)}{\pi_{\text{ref}}(y|y,x)} \pi^*_\dagger(y|y, x) \right) - \log(Z^*(x))$. Now by collecting terms in Eq. (7) and solving for $\log(\pi^*_\dagger(y_2|y_1, x))$ we obtain

$$\log(\pi^*_\dagger(y_2|y_1, x)) = \frac{p(y_2 \succ y_1|x)}{\beta} - \log(Z^*(y_1, x)) - \log(\pi_{\text{ref}}(y_2|y_1, x)) \tag{12}$$

Now by plugging Eq. (5) into Eq. (11) we deduce $\pi^*(y|x) = \frac{\exp(-\log(Z^*(y,x)))\pi_{\text{ref}}(y|x)}{Z^*(x)}$. Solving this equation with respect to $\log(Z^*(y, x))$ implies

$$\log(Z^*(y, x)) = \log(\pi_{\text{ref}}(y|x)) - \log(\pi^*(y|x)) - \log(Z^*(x)). \tag{13}$$

Combining Eq. (12) and Eq. (13) for any y_1 and y_2 we have $\frac{p(y_2 \succ y_1|x)}{\beta} - \log \left(\frac{\pi^*_\dagger(y_2|y_1,x)}{\pi_{\text{ref}}(y_2|y_1,x)} \right) = \log \left(\frac{\pi_{\text{ref}}(y_1|x)}{\pi^*(y_1|x)} \right) - \log(Z^*(x))$, and also $\frac{p(y_1 \succ y_2|x)}{\beta} - \log \left(\frac{\pi^*_\dagger(y_1|y_2,x)}{\pi_{\text{ref}}(y_1|y_2,x)} \right) = \log \left(\frac{\pi_{\text{ref}}(y_2|x)}{\pi^*(y_2|x)} \right) - \log(Z^*(x))$. Subtracting these two Equations and collecting terms leads to our *key result* in which we express the preference p in terms of the self-improvement policy $\pi^*_\dagger$ and the robust policy π^*.

$$\begin{aligned} p(y_2 \succ y_1|x) = & \frac{1}{2} + \frac{\beta}{2} \Bigg[\log \left(\frac{\pi^*_\dagger(y_2|y_1, x)}{\pi_{\text{ref}}(y_2|y_1, x)} \right) + \log \left(\frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)} \right) \\ & - \left(\log \left(\frac{\pi^*_\dagger(y_1|y_2, x)}{\pi_{\text{ref}}(y_1|y_2, x)} \right) + \log \left(\frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} \right) \right) \Bigg]. \end{aligned} \tag{14}$$

Remark 2. *One may notice the similarity of this result and Eq. 6 of* DPO *paper (Rafailov et al., 2023). Both results express $p(y_2 \succ y_1|x)$ in terms of the optimal policy π^*. However the result of* DPO *only holds under the assumption that p conforms to the Bradly-Terry model, whereas our result is general and holds for all p.*

To optimize for π and $\pi_\dagger$ using Eq. (14) we enforce this equation for all y_1 and y_2 through minimizing the expected ℓ_2 loss of Eq. (4) which concludes the proof on joint optimization of the generative policy π and the self-improvement policy through Eq. (4).

4.2.3 PUTTING ALL TOGETHER IN A SINGLE COMBINATION LOSS

We observe that the losses defined in Eq. (4) and Eq. (3) are inherently aligned, as both aim to optimize the same overall objective given in Eq. (1). This alignment allows us to construct a combined loss function for SRPO by using a convex combination of these two losses. Specifically, for any $\alpha \in [0, 1]$, minimizing the convex combination of the losses in Eq. (4) and Eq. (3) is equivalent to directly optimizing the objective in Eq. (1). Mathematically, the full loss of SRPO can be expressed as:

$$\mathcal{L}_{\text{SRPO}} = \alpha \cdot \mathcal{L}_{\text{L2}} + (1 - \alpha) \cdot \mathcal{L}_{\text{self-improve}},$$

where $\mathcal{L}_{\text{L2}}$ corresponds to the loss in Eq. (4) and $\mathcal{L}_{\text{self-improve}}$ corresponds to the loss in Eq. (3). By minimizing this combined loss, we ensure that the optimization process aligns with the original objective in Eq. (1) regardless of the value of α.

This formulation not only unifies the two loss functions under a single framework but also offers flexibility in adjusting the relative contributions of the two losses during training. This completes the proof.

4.3 Deriving the Sample Loss for SRPO

Calculating/optimizing the expected loss of Eq. (17) is often not practical as we often have no direct access to the behavior policy μ. Moreover the space of completions y is often very large (even infinite). So calculating the full expectation is often impractical. Instead one can estimate the expected loss L_α of Eq. (17) using a preference dataset $\mathcal{D}$ with a sample loss $\widehat{L}_\alpha$ and optimize this loss instead. We now focus on deriving the sample loss $\widehat{L}_\alpha$ for SRPO. Using the standard properties of ℓ_2-norm to replace $p(y_2 \succ y_1|x)$ with $\mathbf{1}(y_2 \succ y_1|x)$, as $p(y_2 \succ y_1|x) = \mathbb{E}[\mathbf{1}(y_2 \succ y_1|x)]$, in the objective of Eq. (**??**) allows us to derive the following sample loss for the improvement model:

$$\begin{aligned}\widehat{L}_\dagger(\pi_\dagger) =& \mathbb{E}_{(y_l,y_w,x)\sim\mathcal{D}}\left[\frac{1}{2} - \beta\left[\log\left(\frac{\pi_\dagger(y_w|y_l,x)}{\pi_{\text{ref}}(y_w|y_l,x)}\right) - \log\left(\frac{\pi_\dagger(y_l|y_l,x)}{\pi_{\text{ref}}(y_l|y_l,x)}\right)\right]\right]^2 \\ &+ \mathbb{E}_{(y_l,y_w,x)\sim\mathcal{D}}\left[\frac{1}{2} - \beta\left[\log\left(\frac{\pi_\dagger(y_w|y_w,x)}{\pi_{\text{ref}}(y_w|y_w,x)}\right) - \log\left(\frac{\pi_\dagger(y_l|y_w,x)}{\pi_{\text{ref}}(y_l|y_w,x)}\right)\right]\right]^2.\end{aligned} \tag{15}$$

Using the standard properties of ℓ_2-norm to replace $p(y_2 \succ y_1|x)$ with $\mathbf{1}(y_2 \succ y_1|x)$, as $p(y_2 \succ y_1|x) = \mathbb{E}[\mathbf{1}(y_2 \succ y_1|x)]$, in the loss of Eq. (4) allows us to derive the following sample loss for both the generative policy and the improvement model:

$$\begin{aligned}\widehat{L}(\pi,\pi_\dagger) =& \mathbb{E}_{(y_l,y_w,x)\sim\mathcal{D}}\Bigg[\beta\Bigg[\log\left(\frac{\pi_\dagger(y_w|y_l,x)}{\pi_{\text{ref}}(y_w|y_l,x)}\right) + \log\left(\frac{\pi(y_w|x)}{\pi_{\text{ref}}(y_w|x)}\right) \\ &- \left(\log\left(\frac{\pi_\dagger(y_l|y_w,x)}{\pi_{\text{ref}}(y_l|y_w,x)}\right) + \log\left(\frac{\pi(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right)\Bigg] - 1\Bigg]^2.\end{aligned} \tag{16}$$

looseness=-1 Furthermore one can use a single LLM (denoted by π) to represent both π and $\pi_\dagger$ by exploiting the in-context learning power of LLMs (Brown et al., 2020) such that $\pi_\dagger(y'|y,x) = \pi(y'|y,x)$. So for we define the full sample loss of SRPO as the convex combination of Eq. (16) and Eq. (15):

$$\widehat{L}_\alpha(\pi) = (1-\alpha)\widehat{L}(\pi,\pi_\dagger = \pi) + \alpha\widehat{L}_\dagger(\pi_\dagger = \pi). \tag{17}$$

The following pseudo-code can be used to train the LLM policy using SRPO objective:

Algorithm 1 Sampled SRPO

Require: Dataset $\mathcal{D}$ of prompts, preferred and dis-preferred generations x, y_w and y_l, respectively. A reference policy π_{ref} and a training policy π_θ, regularization coefficient β and combination coefficient α.

```
Initialize π_θ = π_ref
while true do
    Sample a minibatch B ∈ D
    Estimate ∇_θ L̂_α(π_θ) from Eq. (17) using minibatch B as the dataset
    Update π_θ using ∇_θ L̂_α(π_θ) using a standard optimizer
end while
return π_θ
```

5 ROBUSTNESS OF SRPO

We provide a comparison between SRPO and prior work on direct preference optimization in terms of their robustness to the behavior policy μ. In particular we consider as a point of reference DPO (PPO[2]) and IPO for which we have a good understanding of the underlying mathematical foundation.

In the case of both IPO and DPO the analytical solution is already well-established and analyzed for both algorithms (Azar et al., 2023; Rafailov et al., 2023; Tang et al., 2024). In particular the optimal solution for both IPO and DPO can be expressed explicitly in terms of the soft-max of the expected preference as follows (Azar et al., 2023):

$$\pi^*(y|x) = \frac{\exp\left(\beta^{-1}\mathbb{E}_{y'\sim\mu(\cdot|x)}[\Psi(p(y \succ y'|x))]\right)\pi_{\text{ref}}(y|x)}{Z^*(x)}, \tag{18}$$

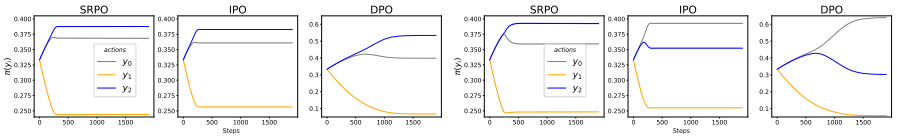
with the choice of $\Psi = I(\cdot)$ and $\Psi = \sigma^{-1}(\cdot)$ for IPO and DPO, respectively, where σ^{-1} denotes the inverse-sigmoid (logit function). Thus, based on Eq. (18), we can see that the solution for both IPO and DPO has strong dependency on μ in the form of expected preference under the distribution μ. Thus it may not be robust to changes in μ. This dependency on μ can be especially problematic when we evaluate the model on out-of-distribution tasks where the desired behavior is very different from μ and the expected preference under the distribution μ is not a good measure of performance. SRPO solution on the other hand has no dependency on the behavior policy μ: from Eq. (5) we observe that the optimal self-improvement policy $\pi^*_\dagger$ is independent of μ and, unlike DPO and IPO cases, is expressed in terms of softmax of $p(y_2 \succ y_1|x)$ for any pair of completions (y_1, y_2). Also the 0-revision policy π^* is also completely independent of μ as it is evident from Eq. (11) (i.e., it is proportional to $\pi^*(y|y, x)$ which itself is independent of μ). Thus, from a mathematical point of view, SRPO provides a robust solution for the problem of direct preference optimization that does not depend on the behavior policy μ.

To illustrate the differences between SRPO and DPO/IPO regarding robustness to μ, we consider a simple bandit example. For simplicity, we assume there is no context x. Consider the simple case where we have 3 completions y_0, y_1, and y_2, for which the preference model is

$$P = \begin{pmatrix} 0.5 & 0.99 & 0.3 \\ 0.01 & 0.5 & 0.25 \\ 0.7 & 0.75 & 0.5 \end{pmatrix}.$$

To test this hypothesis we consider two synthetic dataset of actions generated from distributions μ_0 and μ_1: We set μ_0 to be a uniform behavior policy ($\mu_0(y_0) = \mu_0(y_1) = \mu_0(y_2) = \frac{1}{3}$) and μ_1 skewed towards y_1 ($\mu_1(y_1) = 0.7, \mu_1(y_0) = \mu_1(y_2) = 0.15$). We then generate a dataset of 10000 pairs from μ_0 and μ_1 and rate them according to the preference model p (for any pair (y_1, y_2) we assign the preference by sampling from $p(y_1 \succ y_2)$, that is y_1 is preferred to y_2 with probability $p(y_1 \succ y_2)$). This provides us with two dataset of rated completions $\mathcal{D}_0$ and $\mathcal{D}_1$ for μ_0 and μ_1. We then use these two datasets to train the policy π using SRPO, DPO and IPO using a simple Adam optimizer. In the case of IPO and DPO we optimize only the 0-revision policy $\pi(y)$ where as for SRPO we also optimize the self-improvement policy $\pi(y|y')$ as well. We set the regularization constant β for all methods to 1. We consider a uniform distribution $\pi_{\text{ref}}(y) = 1/3$ for all algorithms and all ys. In the case of SRPO we set the self-improvement reference policy $\pi_{\text{ref}}(y|y') = 1/3$ for all y and y'. Also for SRPO we set the combination coefficient $\alpha = 0$ for simplicity.

We observe that in the case of using uniform μ_0 as a behavior policy all methods do the right thing and their policies converge to solutions in which y_2 dominates y_1 and y_0 (Fig. 2a). However, when we use the behavior policy μ_1 which is skewed towards y_1, both DPO and IPO converge to a solution in which y_0 dominates y_1 and y_2, while the policy of SRPO remains intact (Fig. 2b). Notice that the SRPO policy is slightly different in both cases. This is to be expected, we are in a finite data setting, and the sampling distribution will have some influence on the empirical preference model.



(a) SRPO vs IPO and DPO for uniform μ_0. (b) SRPO vs. IPO and DPO for skewed μ_1.

Figure 2: Learned action probabilities for the synthetic example. SRPO always chooses the correct ~~arm regardless of skew in μ, while~~ both IPO and DPO are effected by the skew (Fig. (2b)).

[2]As it is shown by Azar et al. (2023) the optimal solutions of DPO and PPO are identical. So in the remainder of this section we focus on DPO.

6 EXPERIMENTS

Setup. In our experiments, we consider the offline direct preference optimization setup to learn from human preferences (Rafailov et al., 2023). In the offline setting, the goal is to train the LLM policy directly from a dataset $\mathcal{D}$ of pairwise completions (y_l, y_w) sampled from a behavior policy μ and annotated by human raters without using a reward model or online inference/RL. We empirically test the effectiveness of SRPO against two prominent offline preference learning methods, DPO (Rafailov et al., 2023) and IPO (Azar et al., 2023) as baselines. We make this choice since both these baselines, have been widely used in solving different language tasks (Tunstall et al., 2023; Wallace et al., 2023; Yuan et al., 2024a; Pang et al., 2024a; Lin et al., 2023).

Implementation details. SRPO trains simultaneously both the standard generative policy π and the self-improvement policy $\pi_\dagger$ used for revising the completions of models through a single optimization process. As explained earlier we only use a single LLM to represent both π and $\pi_\dagger$ (denoted simply by π). To get the best completions from SRPO we first generate completions in the 0-revision (0-rev.) model and then we improve these completions with the self-improvement model. We call the revised outputs 1-revision (1-rev.) completions. We can iterate on the improvement process N times to get N-revision (N-rev.) completions. We report results from 0-rev. to 5-rev. cases. For IPO and DPO we also report results on 0-rev. and 1-rev. For revising the completions we use IPO and DPO in in-context learning mode with the 0-rev. completions used as contexts. In the case of DPO we use the same loss and hyper-parameters used by (Rafailov et al., 2023). For IPO since the original paper hasn't provided the hyper-parameters we used a set of hyper-parameters (i.e., learning rate and regularization constant β) from the range of hyper-parameters that was working. Furthermore we noticed that the performance of IPO was not affected significantly by the choice of these hyper-parameters. So no significant gain is expected by hyper-parameter tuning.

Datasets. We use the Reddit TL;DR Summarization dataset (Stiennon et al., 2020) as the main dataset for our experiments[3]. For training, there are 116k human-written instruction following examples with reference completions (SFT split) while there are 93k human-annotated preference pairs (Preference split). We also use the XSum dataset test split[4] (Narayan et al., 2018), which contains 11.5k total test examples to measure Out-of-Distribution (OOD) generalization.

Model Setup. We use LLaMA-7B as base model (Touvron et al., 2023) and a single 8 $\times$ NVIDIA H100 node to conduct all LLaMA-based experiments. We first SFT the model on the SFT split of the TL;DR dataset, before preference training and use the same π_{ref} for all preference training experiment. Below are details on the recipe for the SFT and preference training stages.

Supervised-fine Tuning. In the SFT stage, we train for 2 epochs, using the AdamW optimizer (Loshchilov & Hutter, 2019), with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and 0.1 weight-decay. We use a cosine decay learning rate (Loshchilov & Hutter, 2017) with a peak value of 2×10^{-5} and 3% of all steps being warm-up steps. We use an effective batch-size of 64.

Preference Training. We use our SFT model as π_{ref} and we initialize π with π_{ref}. All models were trained for 5 epochs on the TL;DR preference split using the same optimization setting of the AdamW optimizer as in the SFT stage with 150 warmup steps, and an effective batch-size of 128. To fine-tune the models, we use the default PEFT settings in the TRL library[5], using LoRA (Hu et al., 2022) with a rank of 16 and an alpha of 32. For SRPO and IPO, we used $\beta = 0.01$ with a learning rate of 2×10^{-6}. For DPO following Rafailov et al. (2023), we used the common $\beta = 0.1$ with a learning rate of 1×10^{-6} and a constant learning rate schedule.

Evaluation. We use win rates as computed by gpt-4-0613 (OpenAI, 2023) using the Alpacafarm framework (Dubois et al., 2024), as the main means for evaluation (See Sec. C.3 for more details). We measure performance on both in-distribution and OOD examples at test time in the following manner: For the former, we compute win rates against gold reference completions from the test set of the TL;DR SFT split. For the latter, we measure win rates against gold completions from the test set of the XSum dataset. In both settings, we use the first 1,024 samples from each of the test sets. To estimate the win rate more accurately with confidence intervals, we bootstrap 20 times with replacement from the 1,024 samples, each time using a sample size of 512. To sample from the

[3] https://github.com/openai/summarize-from-feedback
[4] https://huggingface.co/datasets/csebuetnlp/xlsum
[5] https://github.com/huggingface/trl

self-improvement policy, we first sample y from $\pi(\cdot|x)$. Then, using the same policy, we condition on y to sample from the self-improvement policy, that is $y' \sim \pi(\cdot|y, x)$. We refer to generations from $\pi(\cdot|x)$ as 0-rev. (0-rev.) and generations from $y' \sim \pi(\cdot|y, x)$ as 1-rev. (1-rev.) (Bai et al., 2022). For N-revision, we apply the same procedure, conditioning on the sample from the N-1th-rev.

TL;DR Results. We test our models on the test set split of the TL;DR dataset in Fig 3 (left panel). For every model, we generate 0-rev. and then use these generations to revise our completions recursively from 1-rev. to 5-rev. using the self-improvement model, and measure the models' win rate against the human-written gold reference summaries.

We observe that in the case of in-distribution TL;DR SRPO 4-rev. generates high-quality summaries with the highest win rate against the gold summaries, compared to the win-rates of of the baseline methods, as well as other variants of SRPO 0-rev. Furthermore, we observe that SRPO self-improvement process manages to consistently improve upon SRPO 0-rev. . However, DPO and IPO fail to generate an improved sample through the self-improvement step.

Out-of-distribution (OOD) Results. To assess robustness in an OOD setting, we test SRPO models trained with TL;DR preference dataset on the XSum test split in Fig. 3 (right panel) (Narayan et al., 2018). As in the TL;DR case, we observe that self-improvement is effective in improving the performance of SRPO as SRPO 5-rev. generates the highest win rate against the gold summaries, compared to all revisions of the baseline methods, as well as prior revision of SRPO. We also observe that the gap in performance between SRPO and the baselines is significantly higher in OOD case.

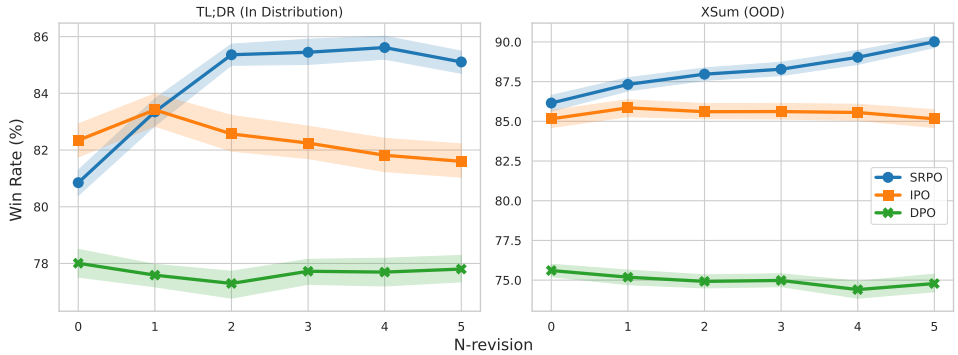


Figure 3: We present the win rates of SRPO, IPO, and DPO against human-written summaries (GOLD) as a function of N-revisions for both in-distribution (TL;DR) and out-of-distribution (XSum) settings. The curves represent the mean win rates, with shaded areas indicating the st.dev. across 20 bootstrap evaluations. Notably, DPO and IPO show no improvements in their generations, whereas SRPO shows significant improvements with each iteration.

7 Discussion and Limitations

In this paper we have developed Self-Improving Robust Preference Optimization (SRPO), a robust offline approach for learning from human preferences. We have proven mathematically and empirically, that unlike other offline methods like DPO and IPO, the solution of SRPO is completely independent of the behavior policy μ and thus SRPO is completely robust to changes in μ.

Summary of results. We have tested SRPO on standard summarization tasks both on in-distribution and out-of-distribution (OOD) regimes. We have observed that in the OOD case SRPO outperforms both IPO and particularly the celebrated DPO by a clear margin in terms of win-rate against gold completions, while in the in-distribution case there is less difference between SRPO and the baselines. This is an expected behavior since in-distribution case the robustness aspect of the algorithm matters less. We have observed that although 0-revision generation of SRPO performs well, we have observed a boost across the board by revising the generation through the self-improvement model.

Future work and Limitations. In our work we used standard and relatively simple language tasks. In the future we would like to apply SRPO to more challenging multi-task benchmarks in which the existing RLHF methods often specialize to a specific set of tasks more represented in the dataset, whereas SRPO should be more resilient due to its robustness to behavior policy μ.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to Basics: Revisiting REINFORCE Style Optimization for Learning from Human Feedback in LLMs, 2024.

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Chi Kit Cheung. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation, 2023. URL https://arxiv.org/abs/2204.01171.

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

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Daniele Calandriello, Daniel Guo, Remi Munos, Mark Rowland, Yunhao Tang, Bernardo Avila Pires, Pierre Harvey Richemond, Charline Le Lan, Michal Valko, Tianqi Liu, Rishabh Joshi, Zeyu Zheng, and Bilal Piot. Human alignment of large language models through online preference optimisation, 2024.

Yiding Chen, Yihe Dong, and Tengyu Ma. Provably mitigating overoptimization in rlhf: Your sft loss is secretly regularizing. *arXiv preprint arXiv:2405.16436*, 2024. URL https://arxiv.org/abs/2405.16436.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, 2017.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023. URL https://arxiv.org/abs/2310.01377.

Abhimanyu Dubey and 532 others. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback, 2024.

Nathan Grinsztajn, Yannis Flet-Berliac, Mohammad Gheshlaghi Azar, Florian Strub, Bill Wu, Eugene Choi, Chris Cremer, Arash Ahmadian, Yash Chandak, Olivier Pietquin, and Matthieu Geist. Averaging log-likelihoods in direct alignment. *arXiv preprint arXiv:2406.19188*, 2024. URL https://arxiv.org/abs/2406.19188.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. Direct language model alignment from online ai feedback, 2024. URL https://arxiv.org/abs/2402.04792.

Chi Hu et al. Teaching language models to self-improve by learning from language feedback. *arXiv preprint arXiv:2406.07168*, 2024.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve, 2022.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the effects of rlhf on llm generalisation and diversity, 2024.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning, 2024. URL https://arxiv.org/abs/2409.12917.

Zhuorong Li, Daiwei Yu, Lina Wei, Canghong Jin, Yun Zhang, and Sixian Chan. Soften to defend: Towards adversarial robustness via self-guided label refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24776–24785, 2024a.

Ziniu Li, Tian Xu, and Yang Yu. Policy optimization in rlhf: The impact of out-of-preference data, 2024b.

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

Hao Liu, Carmelo Sferrazza, and Pieter Abbeel. Chain of hindsight aligns language models with feedback, 2023.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024. URL https://arxiv.org/abs/2405.14734.

Rémi Munos, Michal Valko, Daniele Calandriello, Mohammad Gheshlaghi Azar, Mark Rowland, Zhaohan Daniel Guo, Yunhao Tang, Matthieu Geist, Thomas Mesnard, Andrea Michi, Marco Selvi, Sertan Girgin, Nikola Momchev, Olivier Bachem, Daniel J. Mankowitz, Doina Precup, and Bilal Piot. Nash learning from human feedback, 2023.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206.

OpenAI. Gpt-4 technical report, 2023.

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 amd Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization, 2024a.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization, 2024b. URL https://arxiv.org/abs/2404.19733.

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

Paul Roit, Johan Ferret, Lior Shani, Roee Aharoni, Geoffrey Cideron, Robert Dadashi, Matthieu Geist, Sertan Girgin, Léonard Hussenot, Orgad Keller, et al. Factually consistent summarization via reinforcement learning with textual entailment feedback. *arXiv preprint arXiv:2306.00186*, 2023.

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

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

Yuda Song, Gokul Swamy, Aarti Singh, J. Andrew Bagnell, and Wen Sun. The importance of online data: Understanding preference fine-tuning via coverage, 2024. URL https://arxiv.org/abs/2406.01462.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1f89885d556929e98d3ef9b86448f951-Paper.pdf.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of llms should leverage suboptimal, on-policy data, 2024. URL https://arxiv.org/abs/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. Generalized preference optimization: A unified approach to offline alignment, 2024.

Emanuel Todorov. Linearly-solvable markov decision problems. *Advances in neural information processing systems*, 19, 2006.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

Lewis Tunstall, Edward 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. Zephyr: Direct distillation of lm alignment, 2023.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. *arXiv preprint arXiv:2311.12908*, 2023.

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. *arXiv preprint arXiv:2309.16240*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study, 2024. URL https://arxiv.org/abs/2404.10719.

Yifan Xu, Kening Zhang, Haoyu Dong, Yuezhou Sun, Wenlong Zhao, and Zhuowen Tu. Rethinking exposure bias in language modeling, 2020. URL https://arxiv.org/abs/1910.11235.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models, 2024a.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models, 2024b. URL https://arxiv.org/abs/2401.10020.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. RRHF: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023. URL https://arxiv.org/abs/2304.05302.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. SLiC-HF: Sequence likelihood calibration with human feedback. *arXiv*, 2023a.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J. Liu. SLiC-HF: Sequence likelihood calibration with human feedback. *arXiv*, 2023b.

A RELATED WORKS

Our work lies in offline preference optimization, a vivid area of research since the introduction of DPO (Rafailov et al., 2023). Some of the core concepts of this research topic was generalized and formalized by Azar et al. (2023). In particular they characterized the underlying optimal solution for a generic preference optimization objective and introduced IPO for addressing some of the related shortcomings of DPO. SLiC-HF (Zhao et al., 2023a) was introduced around the same time, from a less RL-centric point of view. All these approaches have been abstracted later by Tang et al. (2024), the general recipe being to build a contrastive loss function from a convex classification function and to make use of the analytical solution of the RL problem to learn directly the policy. A common underlying assumption is that the related RL problem is KL-regularized. This has been generalized to more general f-divergences by Wang et al. (2023). These are just a few among many works on direct alignment (Tajwar et al., 2024; Xu et al., 2024; Guo et al., 2024; Yuan et al., 2024b; Song et al., 2024). However, they all share the fact of not considering self-improvement policies, contrary to SRPO. Concurrent to this work Kumar et al. (2024) use online RLHF for learning self-correcting model. However unlike our method which aims at maximizing the preference of improved completion with respect to the in-context completion their work focuses on maximizing the reward of final answer. The issue with this approach is that the model can simply ignore the in-context completion and try to optimize the final completion without paying attention to in-context completions.

Offline preference optimization was introduced as an alternative to more classic RLHF approaches, such as PPO (Schulman et al., 2017; Ouyang et al., 2022) or more generally policy-gradient-based approaches (Roit et al., 2023; Ahmadian et al., 2024). These methods require training a *reward model* on a preference dataset, usually with a Bradley-Terry model (Bradley & Terry, 1952). The reward model is then used to fine-tune the LLM via online RL, requiring many generations from the model. This reward model shares the common issue of DPO and other direct preference alignment methods, it is dependent on the sampling distribution μ used for constructing the preference dataset, contrary to SRPO. Moreover, classic RLHF is online, while SRPO is offline and thus more easily scalable.

Some similarities also exist between SRPO and Nash-MD (Munos et al., 2023). Indeed, if in 1 we replace the self-improvement policy $\pi_\dagger(\cdot|y, x)$ by a classic policy $\pi(\cdot|x)$, then we obtain the saddle-point optimization problem that Nash-MD solves. However, considering a self-improvement policy is a core contribution of our work, and it is not anecdotal. From a technical viewpoint, this is critical for simplifying the minimax problem of Eq. (1) and obtaining a simple offline optimization problem. NashMD on the other hand adapts algorithms from the game-theory literature and can only be solved online with all the stability issues of online methods and large inference costs. From practical point of view self-improvement provides a boost in performance by refining the original generations of LLM. The feature that Nash-MD is missing. Finally, even though the Nash equilibrium of Nash-MD does not depend on the sampling distribution μ, it relies on a learned reward function, with the possible associated caveats mentioned earlier, which is not the case of SRPO.

Our work is also obviously related to the concept of chain of thoughts (Wei et al., 2023; Yao et al., 2024), self-improvement (Huang et al., 2022) and self-refining LLMs (Madaan et al., 2024). However, it is very often used as a way of prompting a model to obtain better results, and less often as a component of a learning paradigm (Liu et al., 2023; Huang et al., 2022). A notable exception is the recent work of (Pang et al., 2024b) that generalizes DPO to incorporate chain of thoughts. However the main focus of this work is mostly on improving the chain of thoughts reasoning and not on self-improvement. To our best knowledge, we propose the first approach that combines training self-improvement LLMs and offline preference optimization through a single supervised objective, moreover in a theoretically grounded manner and showing the robustness to μ.

B ABLATION: THE EFFECT OF COMBINATION COEFFICIENT α ON SRPO PERFORMANCE

SRPO loss of Eq. (17) is a convex combination of two losses $\widehat{L}$ and $\widehat{L}_\dagger$ via the combination coefficient α. To understand how both terms affects the loss we plot the win rates both in in-distribution case and OOD case as a function of α in Fig. 4. We observe that the term that contributes most to the performance of SRPO is $\widehat{L}_\dagger$ as in the case of $\alpha = 1$ when we only use the loss for improvement model $\widehat{L}_\dagger$ we almost match the best performance. On the other hand using only $\widehat{L}$ (i.e., $\alpha = 0$) is not

enough to achieve top performance. We also observe combining both losses seems to provide some boost in performance especially in OOD case.

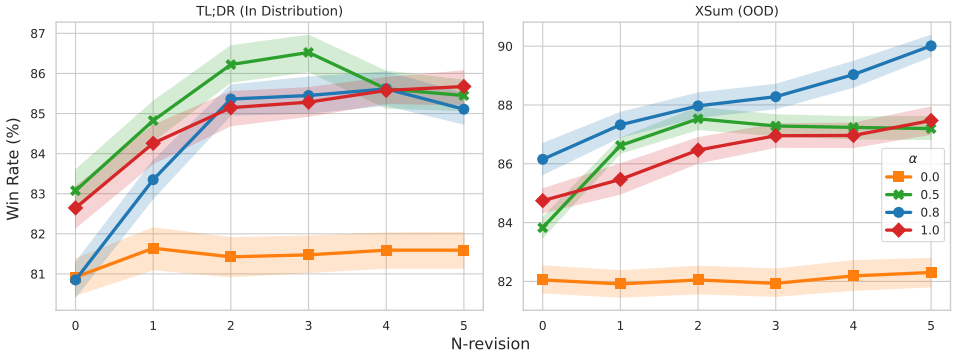


Figure 4: We present the win rates of SRPO against human-written summaries (GOLD) as a function of N-revision iterations at different α values. We report their mean (curve) $\pm$ st.dev. (shaded area), across 20 bootstrap evaluations, as described in the **Evaluation** section. We observe that SRPO achieves meaningful iterative improvements capability as the value of α increases.

C Experimental Details

We provide the prompt templates used for training and evaluations in section 6.

C.1 Prompt Templates

C.1.1 TL;DR

0-revision:

```
Below is a reddit POST and the corresponding SUBREDDIT and TITLE.
Write a both precise and concise summary of the contents of the
POST.

SUBREDDIT: ${subreddit}
TITLE: ${title}
POST: ${post}
TL;DR:
```

N-revision:

```
Below is a reddit POST and the corresponding SUBREDDIT, TITLE, and
 an EXAMPLE SUMMARY. Write a both precise and concise summary of
the contents of the POST.

SUBREDDIT: ${subreddit}
TITLE: ${title}
POST: ${post}
EXAMPLE SUMMARY: ${(N-1)th_example_summary}
TL;DR:
```

C.1.2 XSum

0-revision:

```
Below is a news ARTICLE and the corresponding ID and TITLE. Write
a both precise and concise summary of the contents of the ARTICLE.
```

```
ID: ${id}
TITLE: ${title}
ARTICLE: ${article}
TL;DR:
```

N-revision:

```
Below is a news ARTICLE and the corresponding ID, TITLE, and an
EXAMPLE SUMMARY. Write a both precise and concise summary of the
contents of the ARTICLE.

ID: ${id}
TITLE: ${title}
ARTICLE: ${article}
EXAMPLE SUMMARY: ${(N-1)th_example_summary}
TL;DR:
```

C.2 EXAMPLE SUMMARIES

C.2.1 TL;DR

Post	*I have a horrible caffeine addiction, and I don't like sacrificing any of my daily calories for coffee. I used to drink 5-6 Diet Dr. Peppers a day, but I have switched to almost exclusively drinking only water most days. I do have a Diet Dr. Pepper occasionally, and I know that technically it's okay to probably have one a day, but I honestly just feel better all around when I drink only water. But if I don't have some caffeine in the morning, I will be writhing in the floor in pain with a skull-pounding headache by mid-day. I know. Ridiculous, right?* *So, I went to the vitamin aisle the other day in search of a supplement with a little caffeine and the only thing I found was One A Day Adult Vitacraves Gummies with Energy Support. Here's the website with [nutritional info] One serving is 20 calories and has 90 mg of caffeine. There's also some vitamin B6, B12, and folic acid in there that supposedly helps turn food into energy. I have no idea if that's true. I've been taking them a couple days, and so far I do feel more energized and I haven't had any headaches. The issue is that one bottle is around $9 and there are only 60 in a bottle. You have to take 4 (4!) a day, which is very intriguing considering the brand name and all. So it'll cost me around $20 a month. Which is admittedly cheaper than what I used to spend on Diet Dr. Peppers, but still. I was just curious if anyone else had a better option for caffeine in a supplement or vitamin?*
GOLD	The vitamin with caffeine I'm taking is on the expensive side, and I'm a self-proclaimed tightwad. Anyone have better options?
`SRPO` 1-rev.	I have a caffeine addiction and need a little bit of caffeine in my day. I'm taking One A Day Adult Vitacraves Gummies with Energy Support, but it's expensive ($20/month). Any other suggestions for a caffeine supplement or vitamin?
`SRPO` 0-rev.	I have a terrible caffeine addiction and need a little caffeine in my day. I found a vitamin with caffeine in it, but it's pretty expensive. Any other suggestions for a caffeine supplement or vitamin?
`IPO` 1-rev.	I have a caffeine addiction and need some in my system every day to function, but I don't like sacrificing calories for coffee. I've found a supplement with caffeine, but it's expensive. Anyone have a better option?
`IPO` 0-rev.	I have a caffeine addiction and need some in my system every day to function, but I don't like sacrificing calories for coffee. I've found a supplement with caffeine, but it's expensive. Anyone have a better option?
`DPO` 1-rev.	I have a caffeine addiction and I need a supplement or vitamin that has caffeine in it to take a couple times a week so that I don't have a headache all day. Any recommendations?
`DPO` 0-rev.	I have a caffeine addiction and I need a supplement or vitamin that has caffeine in it to take a couple times a week so that I don't have a headache all day. Any recommendations?

C.2.2 XSUM

Article	*A paper detailing its proposals focuses on the need to avoid a hard border. The government stresses there should be no physical infrastructure, such as customs posts, at the border, which has almost 300 crossing points. Critics say the proposals lack credible detail, with Labour deriding the plans for the border as "a fantasy frontier". The government's paper does not envisage CCTV cameras or number plate recognition technology at the border, or set back from it. Instead, the government is arguing for a wide-ranging exemption under which small and medium-sized businesses will not have to comply with any new customs tariffs. Northern Ireland is the only part of the UK that will share a land border with an EU state post-Brexit. The future management of that border is a highly sensitive issue and is one of three main priorities in UK-EU Brexit negotiations. . . . What is the customs union? Countries in the customs union do not impose tariffs - taxes on imports - on each other's goods. Every country inside the union levies the same tariffs on imports from abroad. So, for example, a 10% tariff is imposed on some cars imported from outside the customs union, while 7.5% is imposed on roasted coffee. Other goods - such as soap or slate - have no tariffs. The UK has said it is leaving the EU's customs union because as a member it is unable to strike trade deals with other countries.*
GOLD	The government has said there must be an "unprecedented solution" for the border between Northern Ireland and the Republic after Brexit.
SRPO 1-rev.	The UK wants to avoid a hard border between Northern Ireland and the Republic of Ireland after Brexit by proposing two customs models to the EU. The UK also dismisses the idea of a customs border in the Irish Sea.
SRPO 0-rev.	Brexit Secretary David Davis wants a limited transition period to implement new customs arrangements, including consideration of the "unique circumstances" of Northern Ireland and the Republic of Ireland. The UK is trying to prevent the need for physical customs posts along the Irish border.
IPO 1-rev.	Theresa May wants a 'seamless' border between Northern Ireland and the Republic of Ireland after Brexit - but the EU may not accept that as viable. The UK government has published a position paper setting out its ideas on the issue.
IPO 0-rev.	Theresa May wants a 'seamless' border between the Republic of Ireland and Northern Ireland after Brexit - but the EU may not accept that as viable. The UK government has published a position paper setting out its ideas on the issue.
DPO 1-rev.	the UK is proposing a "new customs partnership" or a "highly streamlined customs arrangement" to prevent a hard border between Northern Ireland and the Republic of Ireland, in the event of no deal being reached over Brexit.
DPO 0-rev.	the UK is proposing a "new customs partnership" or a "highly streamlined customs arrangement" to prevent a hard border between Northern Ireland and the Republic of Ireland, in the event of no deal being reached over Brexit.

Article	*It follows a row over the removal of personal items from graves in Torfaen which were sent to rubbish tips. Nearly 2,000 people signed a petition calling for legislation on the issue. Public Services Minister Leighton Andrews said he was sympathetic but believed it was a matter for each local authority to "develop and justify their own approaches". Torfaen council said in June 2014 it had organised the clear-up because the over-personalisation of graves in some cemeteries had prompted complaints and made maintenance difficult. The authority apologised for any upset, but said it had made efforts to inform people about the clear-up and had allowed them two months to collect any items they wanted to keep.*
GOLD	Calls for a Wales-wide law stating what tributes can be placed on children's graves have been rejected by ministers.
SRPO {1,..,5}-rev.	Public Services Minister Leighton Andrews has rejected a petition calling for legislation on the removal of personal items from graves, saying it was a matter for each local authority to "develop and justify their own approaches".
SRPO 0-rev.	Public Services Minister Leighton Andrews has rejected a petition calling for legislation on the removal of personal items from graves.
IPO {0,..,5}-rev.	Vicky Pryce wanted revenge on ex-MP Chris Huhne over him getting points on his licence, so she took the speeding points for him in 2003, a court heard.
DPO {0,..,5}-rev.	Vicky Pryce told court she signed speeding points form for her husband Chris Huhne in revenge for him threatening their marriage over his speeding points.

C.3 Evaluation details: calculating win-rate

Calculating the win-rate of AI models, especially in the context of comparisons against human performance or other models, involves several parameters and methodologies. In particular to calculate the win-rate of AI model vs. human, we follow these steps:

1. We run the trained model on the same input data as the test set of supervised dataset and we generate multiple completions.
2. We compare the completions using `gpt-4` with the corresponding ground truth completions of supervised dataset by assigning a point system based on outcomes (e.g., 1 for win, 0.5 for draw, 0 for loss).
3. We aggregate scores across rounds for each model and calculate the win rate.

In particular we use the following formula to calculate the win rate:

$$\text{Win Rate} = \frac{\text{Number of Wins by Model}}{\text{Total Rounds}}$$

We also report confidence intervals to indicate statistical reliability.

D Experiments on Ultra Feedback

Setup. In our experiments, we consider the offline direct preference optimization setup to learn from human preferences (Rafailov et al., 2023). We empirically test the effectiveness of SRPO against two prominent offline preference learning methods, DPO (Rafailov et al., 2023) and IPO (Azar et al., 2023) as baselines. We also consider two popular extensions of DPO, namely, SIMIPO (Meng et al., 2024) and RPO (Chen et al., 2024).

Implementation details. In the case of SRPO and all baselines we optimize the regularization coefficient β and the learning rate by hyper-parameter sweep over a range of possible parameters. (In the case of RPO we also sweep over the cross-entropy coefficient.) Also following on the recent line of work that shows using average log-likelihood is advantageous to using sum log-likelihood

in domains with varying completion length we use the average log-likelihood variant of IPO, DPO, RPO and SRPO (Grinsztajn et al., 2024; Yuan et al., 2023).

Datasets. We use the ShareGPT-Vicuna dataset[6] for SFT, a filtered version of the original dataset containing 53k prompt and completion pairs. For our preference training, we use the binarized version of the UltraFeedback (Cui et al., 2023)[7], which contains 64k pairwise preference data generated by various AI chatbots. For the win-rate evaluation, we use the Arena-Hard dataset[8], which contains 500 prompts, to test against `LLaMA-3.1-8B Instruct`(Dubey & others, 2024), an official post-trained version of `LLaMA-3.1-8B Base` model by the Meta LLaMA team. We generated completions using `LLaMA-3.1-8B Instruct`(Dubey & others, 2024) and Arena-Hard prompts, which we later used to compute win-rates against.

Model Setup. We use `LLaMA-3.1-8B Base` as base model (Dubey & others, 2024) and a Google cloud `v5litepod-256` TPUs to conduct all `LLaMA`-based training and evaluation. We first SFT the `LLaMA-3.1-8B Base` on the ShareGPT dataset. Then, we used the SFT checkpoint to initialize all preference training experiments. Below are details on the recipe for the SFT and preference training stages.

Supervised-fine Tuning. In the SFT stage, we train for a single epoch, using the Adam optimizer (Loshchilov & Hutter, 2019), with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a cosine decay learning rate scheduler(Loshchilov & Hutter, 2017) with a peak value of 2.5×10^{-5}, and decay it down to 1.25×10^{-5}. We use 10 warm-up steps and an effective batch-size of 64.

Preference Training. For a fair comparison across different methods, we initialize all models using the same SFT checkpoint and train them for 1 epoch using the same Adam optimizer setting: we used a learning rate of 1.25×10^{-6}, with a cosine learning rate scheduler to decay it down to 1.25×10^{-7}, with 128 warm-up steps, and an effective batch-size of 32. β_1 and β_2 values of the Adam optimizer are the same as the ones used for the SFT stage. We used the following values of β: $\beta_{\texttt{SRPO}} = 1.3$, $\beta_{\texttt{IPO}} = 1.0$, $\beta_{\texttt{DPO}} = 10.0$, $\beta_{\texttt{RPO}} = 10.0$, and $\beta_{\texttt{SIMIPO}} = 50.0$. For RPO, we used 1.0×10^{-5} as the weight of cross entropy loss on the preferred completions, which is then scaled by β.

To fine-tune the models, we use a training pipeline based on `OPTAX` and `FLAX` libraries of `JAX`.[9]

Evaluation. We use win rates as computed by `gpt-4o` (OpenAI, 2023), as the main means for evaluation (See Sec. C.3 for more details).

Arena-Hard Results. We test our models using the prompts from the Arena-Hard dataset in Fig 5. For every model, we generate 0-rev. and then use these generations to revise our completions recursively from 1-rev. to 5-rev. using the self-improvement model, and measure the models' win rate against the generations of `LLaMA-3.1-8B-Instruct` model for the same prompt.

We observe that srpo 1-rev. produces the highest-quality completions, outperforming both baseline methods and other revisions of SRPO in terms of win rates by a significant margin. Notably, we also observe a significant drop in the win rates for all models after just one revision.

[6] https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered

[7] https://huggingface.co/datasets/HuggingFaceH4/ultrafeedback_binarized

[8] https://lmsys.org/blog/2024-04-19-arena-hard/

[9] https://github.com/jax-ml/jax

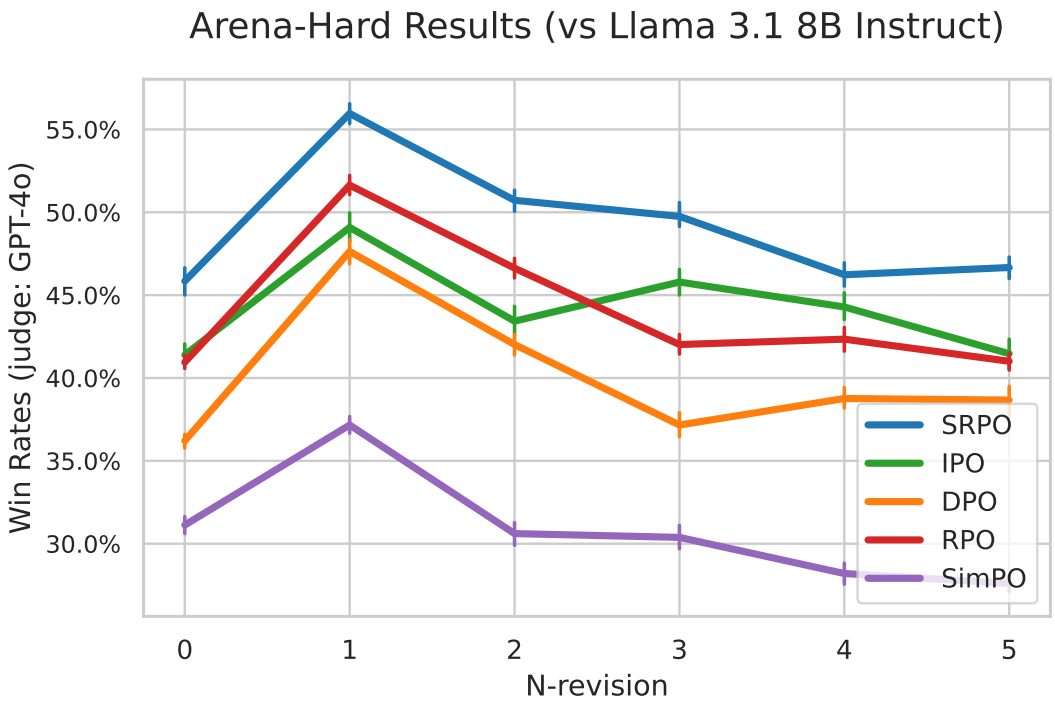


Figure 5: We present the win rates of `SRPO`, `IPO`, `DPO`, `SIMIPO` and `RPO` against `LLaMA-3.1-8B-Instruct` as a function of N-revisions for Arena-hard prompts setting. The curves represent the mean win rates, with shaded areas indicating the st.dev. across 20 bootstrap evaluations. Notably `SRPO` dominates the win-rates of all other methods across all revision steps.