Preference-Strength-Aware Self-Improving Alignment with Generative Preference Models

Yuanzhao Zhai*
yuanzhaozhai@nudt.edu.cn
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Zhuo Zhang*
iezhuo17@gmail.com
Harbin Institute of Technology (Shenzhen); Peng Cheng Laboratory
Shenzhen, China

Cheng Yang
delpiero710@126.com
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Kele Xu
kele.xu@ieee.org
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Yue Yu
yuy@pcl.ac.cn
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Wei Li
jackleewei@sina.com
Independent Researcher
Guangzhou, China

Hui Wang
wangh06@pcl.ac.cn
Peng Cheng Laboratory
Shenzhen, China

Zenglin Xu†
zenglinxu@fudan.edu.cn
Peng Cheng Laboratory; Fudan University
Shenzhen, China

Dawei Feng
davyfeng.c@qq.com
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Bo Ding†
dingbo@nudt.edu.cn
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Huaimin Wang
hmwang@nudt.edu.cn
National University of Defense Technology; State Key Laboratory of Complex & Critical Software Environment
Changsha, China

Abstract

Self-improving alignment, which leverages large language models (LLMs) to generate synthetic preference data automatically, has garnered significant attention as a means of reducing reliance on human labelers. These methods typically employ *LLM-as-a-judge*, where the LLM generates responses and then employs itself to judge which response best aligns with the given prompt for curating the binary self-preferred dataset. However, these methods encounter two major challenges: (1) LLM-as-a-judge often produces error-prone evaluations, resulting in low-quality preference annotation, and (2) their optimization strategies often overlook the strength of preferences within binary pairs, leading to overfitting. This paper proposes a novel method, Preference-Strength-aware Optimization (PSO), to address these issues. Specifically, PSO frames the preference annotation process as a judgment token prediction task using the *generative preference model* to produce reliable judgments. The predicted judgment token indicates the preferred response and its corresponding probability reflects the disparity between responses, referred to as *preference strength*. Based on this strength, we introduce a new preference-strength-aware loss to adaptively reweight the impact of different response pairs on optimization, concentrating the model's learning on high-quality response pairs. Our experiments demonstrate that PSO significantly improves performance in preference benchmarks, achieving stronger alignment with human preferences, reducing verbose responses, and mitigating overfitting. Furthermore, PSO exhibits robust generalization and sample efficiency, offering a scalable and promising solution for LLM alignment without human-annotated preferences.

CCS Concepts

• **Information systems** → **Language models**; • **Computing methodologies** → **Natural language processing**.

Keywords

Large Language Models, Self-Improving Alignment

*These authors contributed equally.
†Corresponding author.

SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730063

ACM Reference Format:
Yuanzhao Zhai, Zhuo Zhang, Cheng Yang, Kele Xu, Yue Yu, Wei Li, Hui Wang, Zenglin Xu, Dawei Feng, Bo Ding, and Huaimin Wang. 2025. Preference-Strength-Aware Self-Improving Alignment with Generative Preference Models. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3726302.3730063

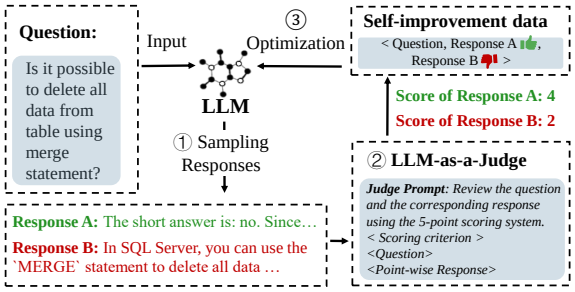
1 Introduction

The remarkable success of large language models (LLMs) in aligning human preference tasks such as instruction following opens up unprecedented opportunities to advance the research and application of information retrieval (IR) [7, 24, 29]. Human-annotated preference data is indispensable in developing LLMs that behave safely and act according to human values and intentions [4, 14]. Existing alignment methods, such as Reinforcement Learning from Human Feedback (RLHF) [21] and Direct Alignment Algorithms (DAAs) [1, 23], rely heavily on large volumes of high-quality preference data to achieve robust model performance [3]. However, curating such data demands significant human effort, time, and financial resources, making the acquisition of preference datasets for target applications both resource- and labor-intensive [5].

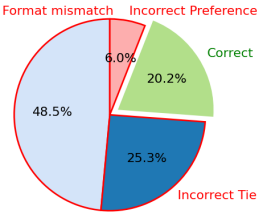
Self-improvement has emerged as a promising strategy to address the scarcity of human-curated preference datasets using self-generated data [27, 28]. As illustrated in Figure 1(a), an LLM can improve itself by serving two phases iteratively: i) following instructions in given prompts to generate appropriate responses, and ii) evaluating new responses as preference data by using *LLM-as-a-judge* to augment its training dataset for alignment [30]. Herein, LLM-as-a-judge assigns a score to a given response based on how well it aligns with the criteria specified in the given prompt, a binary preference pair is curated by comparing the corresponding scores. The approaches implementing this strategy have quickly gained attention in both academia and industry due to their minimal reliance on human-curated preferences [15, 16, 27, 28].

While effective, our study reveals two key limitations of the prior self-improvement approaches. *Firstly*, the scores produced by LLM-as-a-Judge are prone to errors, leading to preference judgments with an error rate as high as 79.8%, as illustrated in the error analysis in Figure 1(b). Notably, 48.5% of the reported errors are caused by format mismatch, while 25.3% of the errors involve LLMs inaccurately assessing preferences as ties. *Secondly*, existing self-improving methods employ DAAs optimize LLMs on self-generated response pairs with binary preferences. Such methods overlook the extent of the differences between two responses in a preference pair, which is prone to be overfitting [1, 22, 31]. In contrast, our study noticed that the strength of the self-generated preferences is proportional to their alignment with human-curated preferences, as shown in Figure 1(c).

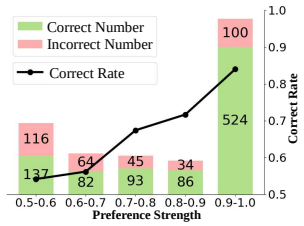
To overcome the above limitations, we frame the self-improvement of LLM alignment as a judgment token prediction problem and propose the Preference-Strength-aware Optimization, abbreviated as PSO. To alleviate the format errors and inaccurate ties caused by LLM-as-a-Judge, we take as input a pair of self-generated response candidates ($A = y^h, B = y^l$) and predict a judgment token indicating which candidate is preferred (A or B). The corresponding



(a) The Pipeline of Self-Improving Alignment.



(b) Evaluation of LLM-as-a-judge.



(c) Evaluation of generative PMs.

Figure 1: In self-improving alignment, LLM-as-a-judge is widely adopted for generating self-improvement data with preference labels. We evaluate the correct proportion of LLM-as-a-judge and generative PMs in human-labeled preferences within MT-Bench [30]. Please refer to Section 5.2 for evaluation results with more advanced LLMs.

probability $p(y^h \succ y^l)$ measures the disparity between y^h and y^l, referred to as *preference strength*. As shown in Figure 1(c), higher preference strengths align more closely with human preferences. Based on this observation, we propose to optimize an LLM on self-generated preferences by introducing a novel loss function for PSO. We leverage the preference strength information to reweight the impact of different response pairs on optimization adaptively. With PSO, response pairs with small preference strengths (i.e., more likely to be inconsistent with human preferences) contribute little to the overall preference optimization. In this way, the optimization can concentrate on response pairs with high quality without any explicit filtering of self-generated samples, thereby enhancing self-improvement performance.

Experimental results demonstrate that PSO achieves significant superiority over existing advanced self-improvement methods on preference benchmarks. Specifically, the PSO using Llama3-8B-Instruct shows significant improvements in AlpacaEval-2 [8], where the win rates rise from 22.92% to 43.66%, and in Arena-Hard [17] increasing from 22.92% to 29.2%. Compared with previous methods, our proposed preference-strength-aware loss function effectively mitigates the generation of verbose responses while reducing the risk of overfitting. Further analysis reveals that the generative PM possesses strong out-of-distribution generalization and sample-efficient capabilities, highlighting its significant potential for real-world applications without reliance on human-annotated preference data. In summary, our main contributions are as follows:

- We revisit the challenge of employing LLM-as-a-judges in self-improving alignment and introduce preference strength,

derived from the generative PM, as a robust indicator of alignment with human preferences.
- Experimental and theoretical analyses demonstrate that our preference-strength-aware loss function effectively alleviates verbose response generation and prevents overfitting.
- Experimental results show PSO achieves new state-of-the-art performance compared to advanced self-improvement methods. Its robust generalization and sample efficiency highlight its compelling potential in practical scenarios without human-annotated preference data.

2 Related Work and Preliminaries

Aligning large language models (LLMs) with human preference data has shown significant promise in improving their performance on downstream natural language processing tasks [4, 14]. Alignment approaches fall into two broad categories: online and offline methods. Online methods involve training a reward model from human preference data, which is then used in reinforcement learning (e.g., PPO [25]) to guide the LLM. Offline methods, such as Direct Preference Optimization (DPO [23]), bypass the reward model and directly use human preference data to optimize the LLM. However, both approaches rely heavily on high-quality, large-scale human preference data, which is expensive and time-consuming to acquire.

Self-improving alignment [6, 13, 28] represents a promising direction to alleviate the dependency on human-labeled preference data. These methods allow LLMs to autonomously generate their preference data autonomously, enabling the model to improve iteratively. Unlike traditional alignment methods, self-improvement techniques enable an LLM to serve two roles iteratively: one as a policy model and the other as a preference model. Specifically, the LLM first generates multiple candidate responses for a given prompt. Then, acting as a preference model, the LLM evaluates these responses and assigns preference scores, typically using an LLM-as-a-judge mechanism [30]. The model is then trained on self-preferred data using offline alignment techniques (i.e., DPO) to improve its performance. Building upon previous work [28], the process of self-improving alignment can be divided into two main steps: self-preferred data construction and preference optimization.

Step 1: Self-preferred data construction. Given a set of prompts and an initial LLM π_{init}. For each prompt $\boldsymbol{x}$, we generate K distinct responses $\{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_K\}$ by high-temperature sampling. Then the judge model maps the number of satisfied rules to specific numerical scores, scoring candidate responses in a point-wise LLM-as-a-judge manner. For example, with the LLM-as-a-judge instruction given in [28], the valid score ranges from 0 to 5. Based on the scores, the self-preference pair $\{\boldsymbol{x}, \boldsymbol{y}^h, \boldsymbol{y}^l\}$ can be determined, where $\boldsymbol{y}^h$ denotes highly preferred response and $\boldsymbol{y}^l$ denotes less preferred response.

Step 2: Preference optimization. The constructed preference data can be utilized to train the LLM using DAAs. For instance, the widely-used DPO aligns the policy model π_θ by solving a binary classification problem:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{init}}) = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}^h,\boldsymbol{y}^l}\Big[\log\sigma\Big(\beta\log\Big(\frac{\pi_\theta(\boldsymbol{y}^h|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^h|\boldsymbol{x})}\Big) - \beta\log\Big(\frac{\pi_\theta(\boldsymbol{y}^l|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^l|\boldsymbol{x})}\Big)\Big)\Big], \tag{1}$$

where the initial LLM π_{init} is usually served by LLM after supervised fine-tuning (SFT) and can be used to normalize the logits. The judge model and policy model are derived from the same LLM π_{init}, thereby achieving self-improving alignment. Existing self-improving alignment methods require multiple iterations of the above two steps to achieve satisfactory self-improvement performances.

While self-improvement holds promise, the LLM-as-a-judge mechanism often produces unreliable scoring results, leading to degraded performance and rapid saturation during iterative training. To address this, Self-Rewarding [28] carefully curates seed data to prime the judgment capabilities of π_{init}. Building on Self-Rewarding, Meta-Rewarding [27] allows the LLM to judge its own judgments and uses that feedback to refine its judgment skills to improve judgment capabilities. However, the problem of unreliable scoring remains. Self-Judge [15] refines the LLM-as-a-judge mechanism by formulating a single-token instruction-following task to identify and select superior responses. Besides, designed for offline datasets with binary preference labels, existing DAAs overlook the preference strength information within the response pairs, which can lead to overfitting and suboptimal alignment.

3 Methodology

To address the limitations of self-improving methods, we introduce a novel approach PSO, which substitutes the LLM-as-a-judge mechanism with a generative preference model and leverages detailed preference strength to enhance alignment efficacy. Following the self-improvement pipeline, PSO exploits the policy model to generate K candidate responses for each prompt and pairs these responses in the K-fold manner. The same LLM then functions as a generative PM to provide the preference strength of each response pair (Section 3.1) to construct self-generated preference pairs embedded with preference strength. Subsequently, we train the policy model with a novel preference-strength-aware loss on the self-preferred data (Section 3.2). Section 3.3 presents a gradient analysis, demonstrating how PSO leverages preference strength to mitigate overfitting.

3.1 Generative Preference Model

After obtaining the self-generated responses, the LLM subsequently serves as a preference model to judge these responses. To address the issue of unreliable scores in the LLM-as-a-judge mechanism, we employ a generative PM, which treats preference tasks as pairwise judgments between responses. As illustrated in Figure 2, the generative PM prompts the model to act as a selector, choosing the superior response based on the probability associated with the judge token ("A" or "B"). This approach naturally avoids issues such as incorrect scoring formats and tied cases.

For each response pair, the generative PM extracts the average probability of the judge token to determine the better response, utilizing a position-swapped judgment template to mitigate position bias. Formally, generative PM transfers the two generated response pairs $\{x, \boldsymbol{y}, \boldsymbol{y}'\}$ to preference pairs $\{x, \boldsymbol{y}^h, \boldsymbol{y}^l\}$ by:

$$\boldsymbol{y}^h = \begin{cases} \boldsymbol{y}, & \text{if } \frac{p_{\boldsymbol{y}}(A)+p_{\boldsymbol{y}}(B)}{2} \geq 0.5, \\ \boldsymbol{y}', & \text{if } \frac{p_{\boldsymbol{y}}(A)+p_{\boldsymbol{y}}(B)}{2} < 0.5. \end{cases} \tag{2}$$

Select the RESPONSE A or RESPONSE B that is better for the given instruction. Here are some rules of the evaluation:
(1) You should prioritize evaluating whether the response honestly/precisely/closely executes the instruction, then consider its helpfulness, accuracy, level of detail, harmlessness, etc.
(2) Responses should NOT contain more/less than what the instruction asks for, as such responses do NOT precisely execute the instruction.
(3) You should avoid any potential bias and your judgment should be as objective as possible. For example, the order in which the responses were presented should NOT affect your judgment, as RESPONSE A and RESPONSE B are **equally likely** to be the better.

Do NOT provide any explanation for your choice.
Do NOT say both / neither are good. You should answer using ONLY "RESPONSE A" or "RESPONSE B".
Do NOT output any other words.
<1-shot example>
Instruction: <User's Question>
RESPONSE A: <Response A>
RESPONSE B: <Response B>
Which is better, RESPONSE A or RESPONSE B? Your response should be either "RESPONSE A" or "RESPONSE B". RESPONSE <A/B>

Figure 2: The instruction of generative preference models for self-improving alignment.

where $p_{\boldsymbol{y}}(A)$ or $p_{\boldsymbol{y}}(B)$ denotes the probability of y being selected as the judge token (i.e., "A" and "B") when placed in `<Response A>` or `<Response B>`, respectively.

Although the binary self-preferring data can be directly utilized to optimize LLMs using DAAs, there is plenty of noise and misspecification during preference annotation, resulting in self-preferring data of varying quality and poor alignment performance (see Section 4.1). This issue motivates us to develop a method to assess the self-preferring provided by the generative PM. Intuitively, humans are more adept at discriminating between responses with a clear disparity while struggling with ambiguous responses. Our experiments also show that higher preference strength values align more closely with human preferences, as depicted in Figure 1(c). This result demonstrates that preference strength can contribute to constructing self-preferred data with richer preference information. Therefore, we propose leveraging the preference strength within a response pair as a measure of preference uncertainty in certain response pairs. Response pairs with strong preference strength correspond to low preference uncertainty. Given the self-preferred example $\{\boldsymbol{x}, \boldsymbol{y}^h, \boldsymbol{y}^l\}$ from Equation (2), we calculate the preference strength via the confidence of self-preferring:

$$p(y^h \succ y^l) = \frac{p_{y^h}(A) + p_{y^h}(B)}{2}. \tag{3}$$

A higher preference strength indicates a larger gap between the responses, making it easier for the LLM to discriminate accurately. In practice, we observe that generative PMs have the problem of excessive caution when providing preference strength (see Figure 6(b)), where the preference strengths for self-generated samples tend to be lower. To address the issue, we apply a simple linear calibration to ensure reasonable outputs, as follows:

$$p(\boldsymbol{y}^h \succ \boldsymbol{y}^l) \leftarrow \eta + \frac{(p(\boldsymbol{y}^h \succ \boldsymbol{y}^l) - 0.5) \cdot (1 - \eta)}{0.5}, \tag{4}$$

where η is the scaling factor and $\eta = 0.5$ means no calibrating. In practice, we can select η from $[0.5, 1]$ to address the excessive caution problem of generative PMs without warming up.

3.2 Preference-Strength aware Preference Optimization

Next, we demonstrate how to use the preference strength aware self-preference data to train the policy model. Generally, alignment methods aim to make the responses generated by the optimized LLM more human-preferred than those before optimization. Given prompt $\boldsymbol{x}$ and the initial LLM π_{init}, the optimization objective is:

$$\arg\max_{\pi_\theta} \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_\mathcal{X}} \mathbb{E}_{\boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x}),\boldsymbol{y}'\sim\pi_{\text{init}}(\cdot|\boldsymbol{x})} \big[p^*(\boldsymbol{y} \succ \boldsymbol{y}'|\boldsymbol{x})\big] - \beta \mathbb{D}_{\text{KL}}(\pi_\theta || \pi_{\text{init}}), \tag{5}$$

where π_θ denotes the policy model to be updated and $p^*(\boldsymbol{y} \succ \boldsymbol{y}'|\boldsymbol{x})$ denotes the optimal preference annotated by humans. According to [23], the closed-form solution to Equation (5) is provided by:

$$\pi^*(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(x)} \pi_{\text{init}}(\boldsymbol{y}|\boldsymbol{x}) \cdot \exp\left(\frac{1}{\beta} p^*(\boldsymbol{y} \succ \pi_{\text{init}}|\boldsymbol{x})\right), \tag{6}$$

where $Z(x)$ is the constant normalizing factor and $p^*(\boldsymbol{y} \succ \pi_{\text{init}}|\boldsymbol{x}) := \mathbb{E}_{\boldsymbol{y}'\sim\pi_{\text{init}}(\cdot|\boldsymbol{x}))} p^*(\boldsymbol{y} \succ \boldsymbol{y}'|\boldsymbol{x})$. By applying this solution to the self-preference data $\{\boldsymbol{x}, \boldsymbol{y}^h, \boldsymbol{y}^l\}$:

$$\sigma\big(\beta \log(\frac{\pi^*(\boldsymbol{y}^h|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^h|\boldsymbol{x})}) - \beta \log(\frac{\pi^*(\boldsymbol{y}^l|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^l|\boldsymbol{x})})\big) = \sigma\big(p^*(\boldsymbol{y}^h \succ \pi_{\text{init}}|\boldsymbol{x}) - p^*(\boldsymbol{y}^l \succ \pi_{\text{init}}|\boldsymbol{x})\big), \tag{7}$$

where $\sigma : \mathbb{R} \to [0, 1]$ is the sigmoid function.

The left-hand side in Equation (7) is the implicit preference predicted by the optimal policy model π^* and normalized by the reference policy π_{init}. During optimization, it can be predicted with the current policy model π_θ:

$$\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta, \boldsymbol{x}) = \sigma\big(\beta \log(\frac{\pi_\theta(\boldsymbol{y}^h|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^h|\boldsymbol{x})}) - \beta \log(\frac{\pi_\theta(\boldsymbol{y}^l|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^l|\boldsymbol{x})})\big). \tag{8}$$

The right-hand side in Equation (7) denotes the expert-labeled preference $p^*(\boldsymbol{y}^h \succ \boldsymbol{y}^l)$, which is unavailable because the self-improving manner bypasses the human annotation. As discussed above, the calibrated preference strength defined in Equation (4) can reflect the consistency between self-preferences and human

preferences, and thus we exploit $p(\boldsymbol{y}^h \succ \boldsymbol{y}^l)$ to approximately estimate the target preference $p^*(\boldsymbol{y}^h \succ \boldsymbol{y}^l)$.

Since both implicit and calibrated preference strengths are probabilities, we use a soft version of the logarithmic/cross-entropy loss, which is commonly employed to measure prediction error [11] of $\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)$. We define the optimization objective with preference strength as:

$$\begin{aligned}\mathcal{L}_{\text{PSO}}(\pi_\theta) = -\mathbb{E}_{\boldsymbol{y}^h,\boldsymbol{y}^l}[p(\boldsymbol{y}^h \succ \boldsymbol{y}^l)\log\big(\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)\big)+ \\ (1-p(\boldsymbol{y}^h \succ \boldsymbol{y}^l))\log\big(1-\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)\big)],\end{aligned} \tag{9}$$

where we suppress the dependence on $\boldsymbol{x}$ in preference strength to denote the expectation of the prompt distribution. The PSO loss aims to make the distribution of implicit preference strength $\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)$ on response pairs $<\boldsymbol{y}^h, \boldsymbol{y}^l>$ as close as possible to the target preference strength $p(\boldsymbol{y}^h \succ \boldsymbol{y}^l)$ that is produced by the generative PMs. Unlike previous approaches, our proposed PSO effectively leverages rich preference information to adaptively optimize the policy model. This preference-strength-aware optimization is more accurate and robust in the absence of human-labeled preferences, as will be theoretically analyzed in the following section.

3.3 Gradient Analysis of PSO

We further provide an analysis that shows how PSO, from a gradient perspective, robustly optimizes the policy model by using preference strength, compared to traditional direct alignment methods (i.e., DPO and IPO). The gradient of our PSO loss is as follows:

$$\begin{aligned}\nabla_\theta\mathcal{L}_{\text{PSO}}(\pi_\theta) =&\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_\mathcal{X},\boldsymbol{y}^h,\boldsymbol{y}^l}[\beta\big(\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)\big) - p(\boldsymbol{y}^h \succ \boldsymbol{y}^l)\big) \\ &\big(\nabla_\theta\log\pi_\theta(\boldsymbol{y}^h|\boldsymbol{x}) - \nabla_\theta\log\pi_\theta(\boldsymbol{y}^l|\boldsymbol{x})\big)].\end{aligned} \tag{10}$$

The gradient of the PSO becomes zero when $\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)) = p(\boldsymbol{y}^h \succ \boldsymbol{y}^l)$, indicating that the model's predicted preference strength matches the target self-preference strength. The PSO loss is utilized to train the model whenever there is a discrepancy between them, enhancing the model's awareness of preference strength. For response pairs with minor preference strength, the corresponding gradient is also controlled within a reasonable magnitude. In comparison, the gradient of DPO is as follows:

$$\begin{aligned}\nabla_\theta\mathcal{L}_{\text{DPO}}(\pi_\theta) = -\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_\mathcal{X},\boldsymbol{y}^h,\boldsymbol{y}^l}[\beta\big(1-\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)\big) \\ \big(\nabla_\theta\log\pi_\theta(\boldsymbol{y}^h|\boldsymbol{x}) - \nabla_\theta\log\pi_\theta(\boldsymbol{y}^l|\boldsymbol{x})\big)].\end{aligned} \tag{11}$$

DPO trains the model until $\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\boldsymbol{x},\pi_\theta)$ reaches the maximum value 1, which necessitates $\log(\frac{\pi_\theta(\boldsymbol{y}^h|\boldsymbol{x})}{\pi_\theta(\boldsymbol{y}^l|\boldsymbol{x})}) \to \infty$. Consequently, DPO is susceptible to overfitting. To mitigate this drawback, IPO [1] incorporates a constant margin to mitigate overfitting on offline datasets.

$$\begin{aligned}\mathcal{L}_{\text{IPO}}(\pi_\theta;\pi_{\text{init}}) = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}^h,\boldsymbol{y}^l}\Big[\\ \big(\log(\frac{\pi_\theta(\boldsymbol{y}^h|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^h|\boldsymbol{x})}) - \log(\frac{\pi_\theta(\boldsymbol{y}^l|\boldsymbol{x})}{\pi_{\text{init}}(\boldsymbol{y}^l|\boldsymbol{x})}) - \frac{1}{2\beta}\big)^2\Big].\end{aligned} \tag{12}$$

IPO introduces a hard margin to enforce the gradient to approach zero when the preference strength reaches a certain threshold:

$$\begin{aligned}\nabla_\theta\mathcal{L}_{\text{IPO}}(\pi_\theta) = -\mathbb{E}_{\boldsymbol{y}^h,\boldsymbol{y}^l}[\big(\sigma^{-1}(\hat{p}(\boldsymbol{y}^h \succ \boldsymbol{y}^l|\pi_\theta)) - \frac{1}{2\beta}\big) \\ \big(\nabla_\theta\log\pi_\theta(\boldsymbol{y}^h) - \nabla_\theta\log\pi_\theta(\boldsymbol{y}^l)\big)].\end{aligned} \tag{13}$$

However, the fixed threshold can cause sub-optimal performance since different response pairs have different preference strengths. In contrast, our proposed PSO loss employs a dynamic margin to regulate the updates for different response pairs according to the preference strength information, thereby mitigating the overfitting issue.

4 Experiment

In this section, we evaluate the performance of LLMs trained with the PSO method in two axes: as a policy model (their ability to follow instructions, see Section 4.1), and as a judge model (their ability to judge responses, see Section 4.2). Our empirical analysis substantiates the following claims: PSO achieves new state-of-the-art performance compared to advanced self-improvement methods (Table 1), because (1) the generative preference model significantly improves the reliability of self-preference data (Table 2) and demonstrates strong generalization on out-of-distribution tasks (Figure 4); (2) the preference-strength-aware loss effectively prevents the generation of verbose responses (Figure 3) and reduces the risk of overfitting (Figure 5). Moreover, PSO is a parameter-efficient and sample-efficient online algorithm (Table 3), showcasing its immense potential in real-world resource-constrained scenarios (e.g., the absence of human preference annotations or the use of smaller LLM, Table 4).

4.1 Alignment Evaluation

4.1.1 Experimental setup. We validate the effectiveness of PSO in the self-improvement alignment scenario. The experimental setup is described in detail as follows:

Dataset and Model. Building on prior work [15], we primarily utilize UltraFeedback[1] for empirical investigations. UltraFeedback is a widely investigated alternative in scenarios where human-annotated feedback is unavailable. It comprises 64k prompts and focuses on the general human preference alignment. In our study, we only exploit the prompts from UltraFeedback to simulate self-improving scenarios. We employ Llama-3-8B-Instruct [2] as our initial model, widely used in recent LLM alignment studies [20, 27].

Baselines. Our baselines include three self-improvement methods: `Self-Rewarding` [28], `Meta Rewarding` [27], and `Self-Judge` [15] Self-Rewarding is the first study to leverage the LLM-as-a-judge prompt to provide rewards and employ iterative DPO to enhance alignment performance. Building upon Self-Rewarding, Meta-Rewarding further improves the judge capabilities of LLMs during the self-improvement process by generating additional judgment data. To address the unreliability of LLM-as-a-judge, both methods utilize seed data constructed with Llama-2-70B [26] to warm up the judge capabilities of the LLM before the self-improvement. Since this seed data is not publicly available, we directly report their best

[1] huggingface.co/datasets/openbmb/UltraFeedback.
[2] huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct.

Table 1: Results of different self-improvement methods evaluated on AlpacaEval-2 and Arena-Hard. Our method achieves state-of-the-art self-improvement performance along with superior length control.

Model	AlpacaEval-2			Arena-Hard		
	LC-WR	WR	Length	WR	95% CI	Length
Llama-3-8B-Instruct	22.92%	22.57%	1899	20.6%	(-2.0, 1.8)	2485
SFT on seed data	25.47%	25.10%	1943	24.2%	(-2.0, 1.8)	2444
Self-Judge [15]	36.56%	40.28%	2268	27.6%	(-2.3, 1.7)	2673
Self-Rewarding [28]						
Iteration 1	26.93%	27.12%	1983	23.2%	(-1.7, 1.9)	2438
Iteration 4	35.49%	35.37%	2005	27.3%	(-2.0, 2.2)	2448
Meta-Rewarding [27]						
Iteration 1	27.85%	27.62%	1949	25.1%	(-1.9, 1.8)	2395
Iteration 4	39.44%	39.45%	2003	29.1%	(-2.3, 2.1)	2422
PSO	**43.66%**	**43.30%**	1993	**29.2%**	(-2.2, 1.6)	2411

performance as stated in their experiments. Notably, our approach *does not* require seed data for warm-up. Self-Judge utilizes generative PMs to prefer self-generated responses, after which DPO is employed for optimization.

Evaluation Protocol. Following previous studies [27], our evaluation includes two widely used open-end benchmarks based on GPT4-as-a-Judge: AlpacaEval-2 [8] and Arena-Hard [17]. AlpacaEval-2 comprises 805 questions drawn from five datasets, while Arena-Hard extends MT-Bench [30] by including 500 additional technical problem-solving queries. For AlpacaEval-2, we provide both the raw win rate (WR) against the baseline GPT4 Turbo model and the length-controlled win rate (LC-WR) designed to mitigate the influence of model verbosity. For Arena-Hard, we report the WR and the 95% confidence interval (CI) judged by GPT4 relative to the baseline model.

Implement Details. In the self-preferred data curation, PSO samples five responses for each prompt from UltraFeedback with a sampling temperature of 0.8. We select the response pair with the highest preference strength to ensure high consistency with human preferences in the subsequent training phase. The training process employs a cosine learning rate schedule, incorporating a 10% warm-up phase across one epoch. The AdamW optimizer is utilized with an initial learning rate of 7×10^{-7}, a batch size of 128, and a maximum sequence length of 2048. All experiments are conducted on four Tesla A100 GPUs, each equipped with 80GB of memory.

4.1.2 Main results. The evaluation results across two benchmarks are presented in Table 1. Our findings are as follows:

(1) **PSO significantly improves benchmark performance, achieving state-of-the-art self-improvement alignment.** On the AlpacaEval-2 benchmark, performance notably increases from 22.92% to 43.66%, surpassing both private models (e.g., GPT-4 of 38.13% and Claude 3 Opus of 40.51%) and advanced self-improvement models. This improvement is particularly remarkable given that the model consists of only 8 billion parameters and does not utilize external supervision. Additionally, in the more challenging Arena-Hard benchmark, which diverges significantly from the distribution of UltraFeedback, we observe a substantial performance boost from 20.6% to 29.2%. These results underscore the substantial potential of self-improvement methods in scenarios without expert-labeled preference data, highlighting their significant practical value.

(2) **PSO demonstrates appealing computation and data resource management compared to prior self-improvement methods.** Unlike Self/Meta-Rewarding, which requires multiple iterative processes (e.g., Iteration 4) to reach the best performance, PSO achieves optimal performance with a single iteration. This makes PSO more practical and energy-efficient for real-world applications. **??** further analyzes the resource consumption of PSO. Moreover, Self/Meta-Rewarding relies on carefully constructing seed data to improve the LLM-as-a-judge ability before self-improvement. In comparison, PSO does not require any additional human-labeled data, further reducing the dependency on annotated preference data for self-improvement.

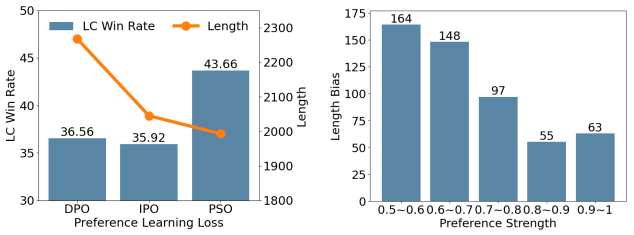
(3) **PSO effectively mitigates the longstanding issue of length explosion in self-improvement methods.** As illustrated in Table 1, although Self/Meta-Rewarding introduces a length-control mechanism to address the length explosion challenge explicitly, this issue persists. The length explosion problem arises from the judge model's preference for verbose responses. While it is theoretically possible that PSO might also exhibit length bias, we further find that incorporating the preference strength information with our proposed preference-aware loss functions effectively alleviates the length bias issue (see Section 4.1.3). As a result, our method produces more concise and high-quality responses.

4.1.3 Further analysis. We first analyze how PSO works. We conduct ablation studies on the AlpacaEval-2 benchmark. `ScorePSO` uses the LLM-as-a-judge to annotate preference data, with the *normalized score difference* as the preference strength. `w/o PSO loss` removes PSO loss, optimizing with DPO. `w/o PS calibration` uses raw generative PM outputs as preference strength without adjustment via Equation (4). For a fair comparison, we also include Self-Rewarding *without seed data* as a baseline. Results are summarized in Table 2.

As shown in Table 2, we find that (1) generative PM is crucial for self-improvement performance. When included, it significantly improves alignment performance, with a 13.49 point gain compared to

Table 2: Ablation studies on AlpacaEval-2. From the results, we see that ① generative PM is crucial for self-improvement performance; ② PS calibration plays an essential role in our method; ③ Preference-strength-aware loss effectively alleviates the generation of verbosity response; ④ Introducing preference strength further improves self-improvement.

	Methods	LC-WR	Length	
①	Self-Rewarding w/o seed data	28.22%	2197	④
①	ScorePSO	30.05%	2158	④
②	w/o PSO loss	41.71%	2275	③
②	w/o PS calibration	35.66%	2050	③
②	PSO	43.66%	1993	③



(a) Performance comparison. (b) Correlation analysis.

Figure 3: The effect of preference strength in reducing length bias. Length bias is calculated by the average length difference between y^h and y^l. Preference strength is the critical factor in curbing verbose responses.

Self-Rewarding and `w/o PSO loss`, highlighting its role in enhancing self-alignment performance. (2) PS calibration plays an essential role in our method. We suspect that it may mitigate excessive caution in LLM judgment, contributing to more reliable outputs. (3) Preference-strength-aware loss effectively alleviates the generation of verbosity response. Removing PSO loss significantly increases response length (283 tokens on average), indicating that the PSO loss could help LLM output more concise responses. (4) Introducing preference strength further improves self-improvement. Despite using less reliable preference strength, `ScorePSO` outperforms Self-Rewarding (Iteration 4) after just one iteration, demonstrating preference strength's significant effect on self-improvement.

The PSO loss demonstrates significant potential in addressing length explosion, motivating us to investigate the underlying reasons. As shown in Figure 3(a), both IPO and PSO effectively reduce verbose responses (orange line), suggesting that anti-overfitting algorithms play a key role in mitigating length issues. However, IPO experiences significant performance degradation due to its use of a fixed margin across all response pairs, whereas our method employs a dynamic margin based on preference strength. We believe that **preference strength is the critical factor in curbing verbose responses.** Supporting evidence is provided in Figure 3(b), which shows that *larger preference strength is associated with reduced length bias.* Under Equation (10), preference strength adjusts the gradient

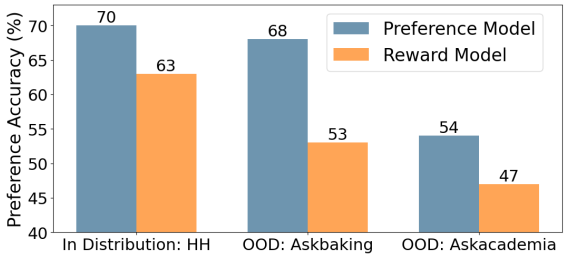


Figure 4: Reward modeling test of the generative PM and Reward Model on ID and OOD test set.

contributions, where pairs with smaller length biases contribute more significantly to model updates. This dynamic prioritization helps mitigate the generation of verbose responses.

4.2 Reward Modeling Evaluation

4.2.1 Experimental Setup. Our approach leverages the generative PM in self-improvement, effectively positioning the LLM as a judge or reward model. We next evaluate the reward modeling performance of PSO compared to the parameterized reward model. Parameterized reward models excel in in-distribution (ID) settings but struggle with out-of-distribution (OOD) generalization [19]. Therefore, our experiment considers two reward modeling settings: in-distribution (ID) performance and out-of-distribution (OOD) generalization.

Dataset and Model. We use the Anthropic Helpful (HH) dataset [2] for ID data, and "askbaking" and "askacademia" subsets from the SHP dataset [9] for OOD evaluation. We chose the TinyLlama-base model instead of an instruction-tuned variant for the following reasons: (1) We find that Llama3-8B-Instruct has data contamination [18] on downstream tasks; (2) we also test our method's scalability, particularly in resource-constrained scenarios where smaller models are necessary.

Evaluation Protocol. We evaluate the reward model and our generative PM in two main aspects, including (1) *rewarding test:* training on ID data with OOD testing, and (2) *downstream task test:* using the model as a reward model for self-improving downstream OOD tasks. For (1), we assess reward modeling performance on the test sets of three datasets and report accuracy. For (2), we use the SteamSHP-flan-t5-xl model [10] as a gold reference for scoring, as it shows high agreement with human preferences on these datasets. Additionally, we evaluate alignment by measuring the win rate against SFT models, using GPT-4 Turbo for judgment.

Baselines and Implement Details. We compare PSO against several baselines: `offline methods` including SFT, and DPO [23], IPO [1]; `online methods` including RLHF [21] and Self-Judge [15]. Self-Judge is chosen for its strong self-improvement capabilities, making it a robust comparison. For all methods, we assume access to human-labeled ID preference data and OOD data containing only instructions. We establish the reward model using TinyLlama by replacing the language head with a value head and use ID preference data to train the reward model. For generative PM, we fill the judgment prompt with paired responses to construct instructions labeled with one of the two judgment tokens. Specifically, the prompt for preference augmentation data is represented as $C(x, y^h, y^l)$, where C denotes the function for filling the judge prompt as shown

Table 3: Sample efficiency between reward model and generative PM. x% denotes we use x portion of ID data for training reward model. Generative PM is more sample-efficient than classic reward model.

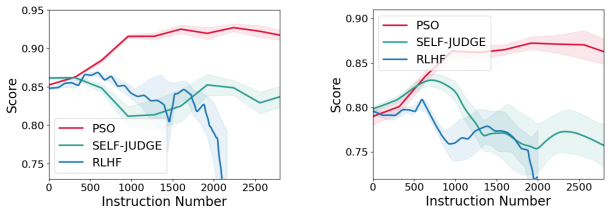
Data Portion	0%	2%	10%	50%	100%
Reward Model	41%	53%	56%	62%	62%
Generative PM	45%	57%	61%	66%	70%

in Figure 2. The corresponding response is either "A" or "B". Then we can conduct SFT to train generative PMs. After training the reward model, we evaluate offline methods directly on downstream OOD tasks. We apply a self-improvement strategy before testing downstream tasks for online methods.

4.2.2 Rewarding test results. Reward modeling performance on ID and OOD tasks is shown in Figure 4. First, the proposed generative PM performs better on ID tasks than the reward model. Generative PM formulates reward modeling as a next-token prediction task for the instruction-following, which better suits LLMs' autoregressive text generation characteristics than the reward model. Second, generative PM shows stronger generalization ability on OOD tasks, which highlights its role in enabling the self-improvement success of PSO. We also evaluate the sample efficiency of the generative PM and reward model. As shown in Table 3, without training, the generative PM and the reward model fail to distinguish $\boldsymbol{y}^h$ and $\boldsymbol{y}^l$. With various portions available, generative PM consistently outperforms the reward model across various dataset portions, suggesting that training generative PMs is more sample-efficient than training the reward model. These results demonstrate its practical potential for real-world downstream tasks.

4.2.3 Downstream task results. The experimental results on downstream OOD tasks, shown in Table 4, reveal the following findings: *First*, PSO significantly outperforms all baseline methods, achieving an average win rate improvement of 27 percentage points over the strong baseline Self-Judge. Notably, this result is obtained using the TinyLlama-base model, demonstrating the practical utility of PSO in real-world downstream tasks, especially in scenarios with constrained computational and annotation resources. *Second*, all online methods consistently outperform offline methods on OOD tasks. This aligns with the inherent advantage of online methods, which can generalize to OOD tasks without collecting new human preference data, leveraging the preference modeling capabilities of generative PMs or reward models. *Third*, compared to RLHF, PSO is more parameter-efficient by utilizing a single model for both the reward and policy functions, thus reducing resource requirements while delivering superior performance.

4.2.4 Further Analysis. To investigate the superior performance of PSO compared to other online learning approaches, we conducted an in-depth analysis and visualized the training process of these methods, as shown in Figures Figures 5(a) and 5(b). The results reveal that both RLHF and Self-Judge exhibit significant overfitting tendencies. For RLHF, this can be attributed to its reliance on a reward model with limited OOD generalization capabilities, leading to severe reward hacking issues [12]. Specifically, reward



(a) Generalize to OOD: Askbaking. (b) Generalize to OOD: Askacademia.

Figure 5: Training curves of online methods on OOD generalization tasks. The preference-strength-aware loss used in PSO effectively mitigates overfitting.

models may assign high rewards to certain low-quality OOD samples, thereby leading to the optimization of LLMs toward learning these undesirable behaviors. Although Self-Judge utilizes a generative PM with better OOD generalization, its subsequent DPO training procedure makes it more susceptible to overfitting (refer to Section 3.3 for gradient analysis). In contrast, **PSO leverages a generative PM with enhanced generalization capacity and incorporates a preference-strength-aware loss, which together provide robust performance and effectively mitigate overfitting challenges.**

5 Discussion

5.1 Scalability of Generative PM

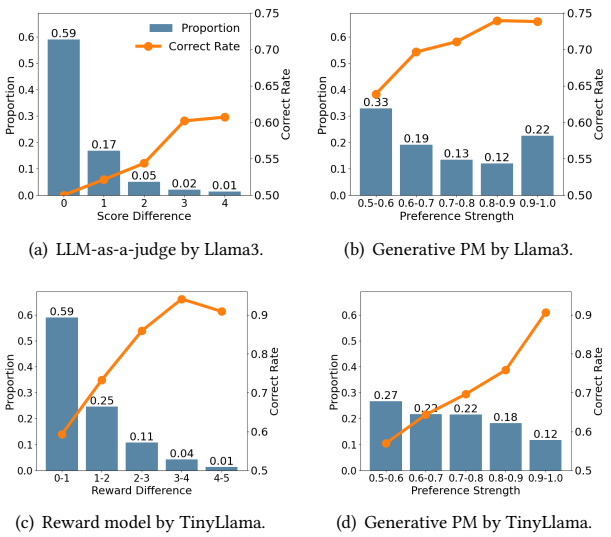
One of the core insights of our work is that the **preference strength derived from generative PM serves as a reliable indicator of alignment with human preferences.** We next investigate whether preference strength obtained from alternative judge models, such as the Reward Model or LLM-as-a-judge, exhibits similar behavior. As visualized in Figures 6(c) and 6(d), we examine the correlation between preference strength computed by these judge methods and the correct rate with human preferences. Our results reveal that irrespective of the type of judge model or the scale of the underlying model, the preference strength consistently aligns with our core insight. This observation underscores the scalability and robustness of the proposed preference strength, demonstrating its adaptability across various judge paradigms and model sizes. Comparing different judgment methods, we observe that preference strength derived from the Reward Model and LLM-as-a-judge tends to concentrate in the lower range for most response pairs. This suggests that these methods struggle to distinguish between self-generated responses effectively. In contrast, our approach leverages the finer-grained generative probabilities provided by generative PM to achieve more precise judgments. This highlights the superiority of our method in capturing nuanced differences between responses, leading to improved alignment with human preferences.

5.2 Is improving instruction-following ability enough for LLM-as-a-judge?

Our study identifies and systematically examines two fundamental limitations of employing LLM-as-a-Judge: insufficient instruction-following and limited judgment capabilities. These limitations manifest as a high incidence of format mismatches and frequent tie

Table 4: Downstream OOD task test. Experiments are repeated four times with different seeds, and we report the average scores and standard deviations. PSO significantly outperforms all online and offline methods.

Method	Online Method	Preference Type	In-distribution: HH		OOD: Askbaking		OOD: Askacademia	
			score	WR	score	WR	score	WR
SFT	No	N.A.	0.919±0.001	-	0.848±0.004	-	0.795±0.007	-
DPO	No	Binary	0.896±0.041	59%	0.733±0.035	31%	0.726±0.033	43%
IPO	No	Binary	0.925±0.004	67%	0.816±0.005	37%	0.793±0.016	51%
RLHF	Yes	Reward	0.932±0.007	57%	0.783±0.080	33%	0.718±0.085	46%
Self-Judge	Yes	Binary	0.923±0.133	63%	0.839±0.020	43%	0.780±0.069	47%
Ours	Yes	Strength	**0.958**±0.006	**73%**	**0.925**±0.005	**75%**	**0.886**±0.004	**70%**



(a) LLM-as-a-judge by Llama3.

(b) Generative PM by Llama3.

(c) Reward model by TinyLlama.

(d) Generative PM by TinyLlama.

Figure 6: Self-preferences of Llama3-8B-Instruct on UltraFeedback (top line) and TinyLlama on HH (bottom line). "Correct Rate" refers to the consistency between self-preferences and the external strong reward model. "Proportion" denotes the fraction of samples that fall into a specific bin. preference strength could serve as a reliable indicator of alignment with human preferences.

cases in paired responses, as depicted in Figure 1(b). Both issues significantly impede effective self-improvement. we next investigate whether using larger LLMs with stronger instruction-following abilities can yield satisfactory LLM-as-a-Judge results. The experimental outcomes are summarized in Table 5.

While larger LLMs with enhanced instruction-following capabilities, such as Mistral-7B-Instruct-v0.2, Llama-2-70B-Instruct (used in the Self/Meta-Rewarding baseline [28]), and Llama-3.1-70B, mitigate format mismatches, our experiments reveal that **LLM-as-a-Judge with more advanced models remain inadequate for resolving wrong tie cases effectively.** In contrast, generative PM that leverages the log probabilities of generating tokens "A" and "B" — instead of mapping the number of satisfied rules to discrete

Table 5: Evaluation results of various LLMs using LLM-as-a-judge on MT-Bench. Dark green text signifies correct judgments, whereas dark red text highlights errors. IP means "Incorrect Preference", and FM means "Format Mismatch".

Model	Correct	IP	Tie	FM
Llama-3-8B-Instruct	20.2%	6.0%	25.3%	48.5%
Mistral-7B-Instruct-v0.2	31.2%	11.4%	30.8%	26.6%
Llama-2-70B-Instruct	20.0%	9.27%	31.39%	39.33%
Llama-3.1-70B	55.69%	11.60%	25.47%	7.24%

numerical scores — successfully addresses these limitations. By eliminating format mismatches and reducing tie cases, the generative PM built with Llama-3-8B-Instruct achieves a markedly higher agreement with human judgments.

6 Conclusion

This study introduces a novel and strong self-improvement approach Preference-Strength-aware Optimization (PSO). By reframing the preference task as a judgment token prediction problem, PSO effectively mitigates the format errors and inaccurate ties inherent in LLM-as-a-judge. Through a novel loss function that adaptively weights optimization based on preference strength, PSO reduces overfitting risks and verbose response generation. Our experiments demonstrate that PSO achieves significant improvements over advanced self-improvement methods, with substantial gains in preference benchmarks such as AlpacaEval-2 and Arena-Hard. These results show that preference strength derived from generative PM serves as a reliable indicator of alignment with human preferences. Moreover, the generative preference model exhibits strong out-of-distribution generalization and sample efficiency, making it highly suitable for practical applications. Therefore, PSO represents a crucial step forward in advancing LLM alignment while reducing dependency on human-curated preference datasets. We hope our method will lead to further research in this promising direction.

ACKNOWLEDGMENTS

This work was partially supported by the Hunan Provincial Innovation-driven Development Plan Project (Grant NO. 2023RC1005) and was partially supported by the Open Fund of National Key Laboratory of Parallel and Distributed Computing (PDL) NO.2024-KJWPDL-02.

References

[1] Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036* (2023).

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

[3] Pietro Bernardelle and Gianluca Demartini. 2024. Optimizing LLMs with Direct Preferences: A Data Efficiency Perspective. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*. 236–240.

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

[5] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377* (2023).

[6] Mucong Ding, Souradip Chakraborty, Vibhu Agrawal, Zora Che, Alec Koppel, Mengdi Wang, Amrit Bedi, and Furong Huang. [n. d.]. SAIL: Self-improving Efficient Online Alignment of Large Language Models. In *ICML 2024 Workshop on Theoretical Foundations of Foundation Models*.

[7] Qian Dong, Yiding Liu, Qingyao Ai, Zhijing Wu, Haitao Li, Yiqun Liu, Shuaiqiang Wang, Dawei Yin, and Shaoping Ma. 2024. Unsupervised large language model alignment for information retrieval via contrastive feedback. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 48–58.

[8] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475* (2024).

[9] Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding Dataset Difficulty with $\mathcal{V}$-Usable Information. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 5988–6008. https://proceedings.mlr.press/v162/ethayarajh22a.html

[10] Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022. Understanding Dataset Difficulty with $\mathcal{V}$-Usable Information. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 5988–6008. https://proceedings.mlr.press/v162/ethayarajh22a.html

[11] Dylan J Foster and Akshay Krishnamurthy. 2021. Efficient first-order contextual bandits: Prediction, allocation, and triangular discrimination. *Advances in Neural Information Processing Systems* 34 (2021), 18907–18919.

[12] Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*. PMLR, 10835–10866.

[13] Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large Language Models Can Self-Improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 1051–1068.

[14] Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, et al. 2023. Ai alignment: A comprehensive survey. *arXiv preprint arXiv:2310.19852* (2023).

[15] Sangkyu Lee, Sungdong Kim, Ashkan Yousefpour, Minjoon Seo, Kang Min Yoo, and Youngjae Yu. 2024. Aligning Large Language Models by On-Policy Self-Judgment. *arXiv preprint arXiv:2402.11253* (2024).

[16] Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. 2024. LLMs-as-Judges: A Comprehensive Survey on LLM-based Evaluation Methods. *arXiv preprint arXiv:2412.05579* (2024).

[17] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024. From Crowdsourced Data to High-Quality Benchmarks: Arena-Hard and BenchBuilder Pipeline. *arXiv preprint arXiv:2406.11939* (2024).

[18] Inbal Magar and Roy Schwartz. 2022. Data contamination: From memorization to exploitation. *arXiv preprint arXiv:2203.08242* (2022).

[19] Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. Generative reward models. *arXiv preprint arXiv:2410.12832* (2024).

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

[21] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.

[22] Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, Bradley Knox, Chelsea Finn, and Scott Niekum. 2024. Scaling laws for reward model overoptimization in direct alignment algorithms. *arXiv preprint arXiv:2406.02900* (2024).

[23] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems* (2023).

[24] Yankun Ren, Zhongde Chen, Xinxing Yang, Longfei Li, Cong Jiang, Lei Cheng, Bo Zhang, Linjian Mo, and Jun Zhou. 2024. Enhancing sequential recommenders with augmented knowledge from aligned large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 345–354.

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

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

[27] Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason Weston, and Sainbayar Sukhbaatar. 2024. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge. *arXiv preprint arXiv:2407.19594* (2024).

[28] Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-Rewarding Language Models. *arXiv preprint arXiv:2401.10020* (2024).

[29] ChengXiang Zhai. 2024. Large language models and future of information retrieval: opportunities and challenges. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 481–490.

[30] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems* 36 (2023).

[31] Banghua Zhu, Jordan Michael, I., and Jiantao Jiao. 2024. Iterative Data Smoothing: Mitigating Reward Overfitting and Overoptimization in RLHF. *arXiv preprint arXiv:2401.16335* (2024).