Self-Steering Optimization: Autonomous Preference Optimization for Large Language Models

Hao Xiang[1,3], **Bowen Yu**[2*], **Hongyu Lin**[1*], **Keming Lu**[2], **Yaojie Lu**[1], **Xianpei Han**[1], **Ben He**[1,3], **Le Sun**[1], **Jingren Zhou**[2], **Junyang Lin**[2]

[1]Chinese Information Processing Laboratory, Institute of Software, Chinese Academy of Sciences
[2]Alibaba Group [3]University of Chinese Academy of Sciences

{xianghao2022, hongyu, luyaojie, xianpei, sunle}@iscas.ac.cn benhe@ucas.edu.cn
{yubowen.ybw, lukeming.lkm, jingren.zhou, junyang.ljy}@alibaba-inc.com

Abstract

The key to effective alignment lies in high-quality preference data. Recent research has focused on automated alignment, which involves developing alignment systems with minimal human intervention. However, prior research has predominantly focused on developing data generation methods, while insufficient attention has been paid to quality control mechanisms, which often produce inaccurate and unhelpful data, leading to unpredictable benefits during iterative optimization. In this paper, we present Self-Steering Optimization (SSO), an algorithm that autonomously generates high-quality preference data, eliminating manual annotation requirements. SSO employs a specialized optimization objective to build a data generator from the policy model itself, which is used to produce accurate and on-policy data. We demonstrate SSO's effectiveness through comprehensive experiments on two series of models: Llama 3 and Qwen 2. Our evaluation across diverse benchmarks shows that SSO consistently outperforms baselines in human preference alignment and reward optimization. Further analysis validates SSO as a scalable framework for preference optimization, benefiting the advancement in automated alignment techniques.

1 Introduction

The field of Natural Language Processing has undergone revolutionary advancements driven by large language models (LLMs). After meticulous alignment processes, LLMs have demonstrated remarkable capabilities in following instructions and understanding human preferences. This leads to the development of widely acclaimed products like ChatGPT (OpenAI, 2023), which have captured significant public attention. However, aligning LLMs with human preferences is not trivial. Despite the existence of Proximal Policy Optimization (PPO) (Ouyang et al., 2022), an ideal alignment training process requires a robust reward model and a stable reinforcement learning process, encouraging researchers to develop offline preference optimization algorithms such as Direct Preference Optimization (DPO) (Rafailov et al., 2023). However, algorithms like DPO rely on a substantial amount of high-quality, annotated preference data, which is both resource-intensive and requires meticulous attention. In addition, the limited capabilities of human annotators cause inherent limitations in annotated data, making it challenging to achieve *superalignment* (Burns et al., 2023).

Consequently, recent researchers have shifted their focus towards automated alignment, with the intention of developing scalable, high-quality alignment systems with minimal human intervention. The cornerstone of this paradigm is the pursuit of scalable alignment signals that are capable of effectively replacing human-annotated preference data. Current popular strategies include self-judgement (Yuan et al., 2024; Wu et al., 2024), principle-based automated alignment (Yang et al., 2024b; Bai et al., 2022b), Constitutional AI (Bai et al., 2022b), and other methods (Fränken et al., 2024; Kumar et al., 2024).

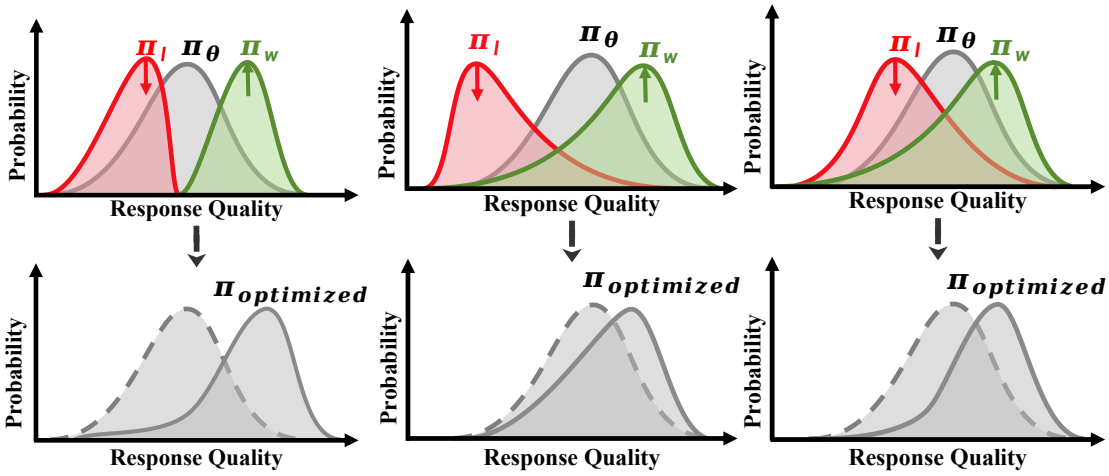
However, these methods do not pay enough attention to quality control mechanisms. The ideal preference data, as defined in RLHF (Ouyang et al., 2022), is constructed through human annotation, where the responses are sampled from the policy model and ranked according to their qualities. As shown in Figure 1(a), this type of preference data demonstrates high accuracy[1] and on-policy nature[2], leading to ideal alignment optimization (Tajwar

* Corresponding authors.

[1]**Accuracy of preference data**: the rate of the response pairs where the chosen response has higher quality than the rejected response.

[2]**On-policy data**: the responses lie in the high-probability region of the policy model. The higher generation probability indicates better on-policy behavior.

Findings of the Association for Computational Linguistics: ACL 2025, pages 9073–9085



(a) Ideal preference data. (b) Accurate but off-policy data. (c) On-policy but inaccurate data.

Figure 1: Distribution changes of policy model π_θ after optimization with three types of preference data. During alignment, chosen responses π_w receive positive gradients to increase probability, while rejected responses π_l receive negative gradients to decrease probability. (a) Ideal data, with high accuracy (low overlap between π_w and π_l) and on-policy nature (π_l lies in high-probability region of π_θ), leading to right optimization direction and effective negative gradients optimization. (b) Sub-optimal data, with high accuracy (low overlap between π_w and π_l) and off-policy nature (π_l lies in low-probability region of π_θ), weakening negative gradients optimization. (c) Sub-optimal data, with low accuracy (high overlap between π_w and π_l) and on-policy nature (π_l lies in high-probability region of π_θ), interfering optimization direction.

et al., 2024; Kim et al., 2024). However, automated methods often fail to simultaneously guarantee accurate preferences and on-policy responses, instead producing suboptimal data, as shown in Figures 1(b) and 1(c), which impedes model alignment. For example, self-judgment is hampered by the inherent limitations of the model; this judging ability is restricted and difficult to improve, often resulting in hacked rewards and inaccurate preference data in Figure 1(c) (Wu et al., 2024). In other methods, incorporating additional input or processes may lead to off-policy responses and sub-optimal preference data, as shown in Figure 1(b).

We then recognized the need for a novel approach to generate high-quality preference data to address these limitations and advance automated alignment. One problem is to control the distribution of the chosen and rejected responses when constructing preference data. For most automated alignment methods, this seems an impossible task because chosen and rejected responses are typically obtained through complex pipelines. However, we found that principle-based methods (Yang et al., 2024b; Bai et al., 2022b) can achieve this goal because they construct preference data by directly sampling from the policy model based on good and bad principles. It can be approximated that the distributions of the chosen and rejected responses π_w and π_l are the distribution of the policy model π_θ with good and bad principles.

In this work, we introduce Self-Steering Optimization (SSO), a pioneering method that automatically generates accurate and near-on-policy preference data for the policy model. SSO optimizes a data generator with a special loss to control the distributions of chosen and rejected responses, and then uses the data produced by this generator to further optimize the policy model. Specifically, SSO first prompts the policy model with original queries x and a set of contrastive principles p^+ and p^- for the responses as training data and then optimizes the policy model based on two key objectives: a) Making rejected responses approximately on-policy to ensure the effectiveness of negative gradients in subsequent optimization. We only take care of the rejected responses, as the chosen responses usually have high generation probabilities. b) Maintaining a consistent gap between the chosen and rejected responses to ensure the accuracy of the preference data.

We demonstrate the effectiveness of Self-Steering Optimization on Qwen2 (Yang et al., 2024a) and Llama3 (Llama Team, 2024) backbones. Our experiments reveal SSO's ability to

generate accurate and on-policy preference data. As a result, improvements are observed on a wide range of benchmarks such as MATH (Hendrycks et al., 2021b), IFEval (Zhou et al., 2023), MT-Bench (Zheng et al., 2024b), and AlpacaEval 2.0 (Dubois et al., 2024). Furthermore, we conducted experiments through reward optimization, which also achieved satisfying results. Without human annotation or external models, SSO even outperforms baselines with annotated data (Cui et al., 2024), underscoring its potential as a scalable and efficient alignment approach.

2 Related Works

Preference Alignment Researchers have proposed various algorithms to align large language models (LLMs) with human preferences. Ziegler et al. (2020); Ouyang et al. (2022); Bai et al. (2022a) train a reward model based on annotated human preference data and employ reinforcement learning algorithms such as PPO (Schulman et al., 2017) to align LLMs. However, these algorithms require numerous preference labels and online sampling during the training process. To further reduce costs, direct preference optimization (DPO), sequence likelihood calibration (SLiC) (Zhao et al., 2023), identity preference optimization (IPO) (Azar et al., 2023), and Kahneman-Tversky optimization (KTO) (Ethayarajh et al., 2024) simplify the RLHF objective by directly increasing the margin between chosen and rejected responses.

Automated Alignment Previous alignment studies rely on manually annotated preference data and algorithms such as RLHF and DPO to conduct model alignment. Recently, numerous studies have found that LLM-generated data can reach the quality of ordinary manual annotations (Zheng et al., 2024b). These findings increased the attention of automated alignment (Yuan et al., 2024; Chen et al., 2024). Automated alignment aims to minimize human intervention by addressing the prohibitively expensive cost of human annotation. Current methods can be divided into four types based on the source of alignment signals (Cao et al., 2024): 1) Inductive Bias, from introducing appropriate assumptions and constraints (Huang et al., 2023a; Bai et al., 2022b; Yang et al., 2024b; Yuan et al., 2024; Chen et al., 2024). 2) Behavioral Imitation, another aligned model (Peng et al., 2023; Tunstall et al., 2023; Burns et al., 2023). 3) Model Feedback, feedbacks from other models (Lee et al., 2023; Hosseini et al., 2024). 4) Environmental Feedback, environmental interaction (Liu et al., 2023; Qiao et al., 2024).

3 Preliminaries

3.1 Symbol Definition of Automated Alignment

Specifically, given a query set $X = \{x_i\}_{i=1}^{N}$, where N is the number of queries, automated methods focus on how to use the policy model π_θ to generate the chosen response y^+ and the rejected response y^- for the preference data $D = \{x_i, y_i^+, y_i^-\}_{i=1}^{N}$, which will be used to optimize π_θ with alignment algorithms.

3.2 Principle-Based Automated Alignment

Principle-based automated alignment (PBAA) is one of the most common automated alignment methods (Yang et al., 2024b; Fränken et al., 2024), which assumes that responses with different qualities can be directly sampled from LLMs with different queries. This approach constructs pairs of contrastive queries x^+ and x^- to sample chosen and rejected responses from the policy model as training data. Since contrastive queries exhibit contrasting attributes (such as harmful vs. harmless), the generated preference data has high accuracy. Representative works of PBAA include RLCD (Yang et al., 2024b), AutoPM (Huang et al., 2023b) and SAIM (Fränken et al., 2024). The first two use specific word pairs, such as "inoffensive response" and "offensive response", to generate response pairs for model alignment, while SAIM employs generated principles.

However, these methods do not guarantee accurate and on-policy data. Incorporating additional principles could lead to off-policy responses, as shown in Figure 1(b) (Tajwar et al., 2024; Kim et al., 2024).

3.3 Automated Alignment from Distribution Sight

In the alignment process, negative gradients and positive gradients are used to decrease and increase the probability of rejected responses and chosen responses, respectively. Figure 1(a) illustrates the ideal distribution of the preference data.

For models that are not aligned at all, automated alignment approaches may construct preference data where the distribution of the chosen responses

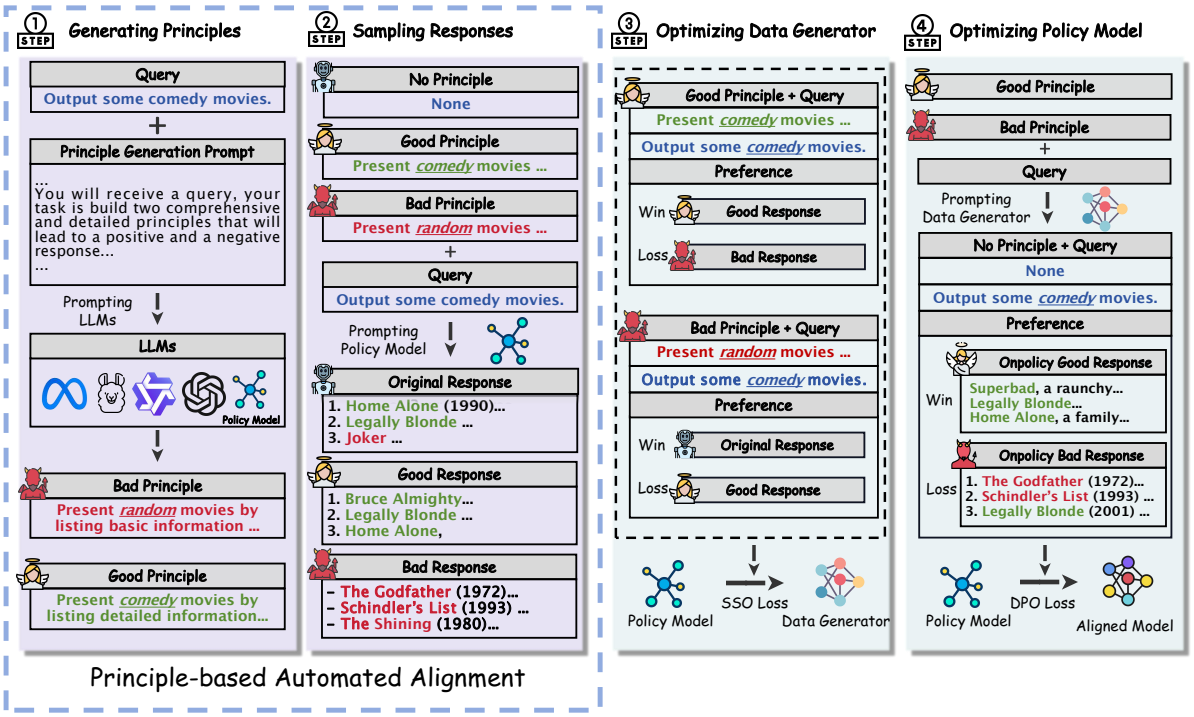


Figure 2: The pipeline of Self-Steering Optimization (SSO). The overall process consists of four steps: 1) generate Principles, 2) sample Responses, 3) optimize the Data Generator, and 4) optimize the policy model. The data generator is optimized with the loss 1 from the policy model π_θ and used to generate accurate and on-policy preference data for the policy model.

π_w is far from the distribution of rejected responses π_l, and π_w are in the low-probability region of π_θ, as mentioned by Tajwar et al. (2024), the on-policy nature has minimal impact on model optimization in this scenario. This explains the improvements of various automated methods.

However, as LLMs advance, even those without explicit alignment can exhibit aligned behaviors. In this situation, π_w lies in the high-probability region of π_θ, and the on-policy performance of π_l becomes crucial. If π_l lies in the low-probability region of π_θ, applying negative gradients to such responses would be meaningless and result in sub-optimal alignment. Therefore, we propose Self-Steering Optimization.

4 Self-Steering Optimization

4.1 Pipeline of SSO

As shown in Figure 2, the pipeline of SSO consists of four steps: 1) generating principles, 2) sampling responses, 3) optimizing the data generator, and 4) optimizing the policy model.

Generating Principles Given a query x, we employ a principle generator (we used Qwen2.5-72B-Instruct(Qwen et al., 2025) in our main experiments) to construct a pair of contrastive principles (p^+, p^-). The example of principles can be found in the Appendix 7.

Sampling Responses Principles (p^+, p^-) are then concatenated with the original query x to build contrastive queries (x^+, x^-).[3] Then, (x^+, x^-) are used to prompt the policy model π_θ for the good and bad responses (y^+, y^-). Furthermore, to optimize the data generator, we generate an original response y^o without any principle.

Optimizing Data Generator We optimize a data generator with the loss function in Formula 1, which is designed to generate accurate and near-on-policy preference data. The data generator is optimized from the policy model.

Optimizing Policy Model We optimize the policy model π_θ with the preference data generated by the data generator. The policy model is optimized with an alignment algorithm, such as DPO, to align the model.

Detailed templates are provided in the Appendix.

[3]We use principles as system messages.

4.2 Objective

Self-Steering Optimization aims to generate accurate and near-on-policy preference data. As described in Section 4.1, given the following components:

Two principles : Good principle p^+ and bad principle p^-, which are generated by LLMs for the query x specially.

Three queries : Original query x, good query x^+, and bad query x^-, where x^+ and x^- are built from x with p^+ and p^-, respectively.

Three responses : The good response y^+, the bad response y^-, and the original response y^o, where y^i is sampled from π_θ with query x^i, $i \in \{+, -, o\}$.

We propose a novel loss function $\mathcal{L}_{SSO}$ to optimize the data generator:

$$\mathcal{L}_{SSO} = \mathcal{L}^+ + \mathcal{L}^- \quad (1)$$

where $\mathcal{L}^+$ and $\mathcal{L}^-$ are used to optimize the good and bad responses, respectively.

Loss 1 is the core loss function of our method, used to optimize the Policy model to obtain a Data Generator (as shown in Step 3 of Figure 1). Although the Data Generator is optimized from the Policy Model, it has no direct relationship with the final aligned model and is only used to generate data. $\mathcal{L}_{SSO}$ is only used to optimize the Data Generator, not for the final alignment. When aligning the policy model with the data generated by the Data Generator, we use $\mathcal{L}_{Base}$, which is the xPO Loss, such as DPO, IPO, etc. This might be the source of your confusion.

As mentioned in Section 3.3, $\mathcal{L}_{SSO}$ should minimize the overlap between π_w and π_l, while ensuring that π_l lies in the high-probability region of π_θ (for effective negative gradients optimization). Naturally, we design $\mathcal{L}^+$ as:

$$\mathcal{L}^+ = L_{Base}(\mathbf{x}^+, \mathbf{y}^+, \mathbf{y}^-) + \gamma L_{sft}(\mathbf{x}^+, \mathbf{y}^+) \quad (2)$$

where L_{Base} can be any xPO alignment algorithm(like DPO, IPO, etc.) loss function, and L_{sft} is the SFT loss function. γ is a hyperparameter that balances SFT loss and alignment loss, helping to maintain training stability.

Similarly, a natural approach is to construct the loss function $\mathcal{L}^-$ as:

$$\mathcal{L}^- = L_{Base}(\mathbf{x}^-, \mathbf{y}^-, \mathbf{y}^+) + \gamma L_{sft}(\mathbf{x}^-, \mathbf{y}^-) \quad (3)$$

However, this approach introduces a problem: with a bad principle p^-, LLMs may output unpredictable results. In other words, this loss could lead to a π_l that lies in the low-probability region of π_θ, which cannot help generate on-policy data and hamper alignment.

Therefore, for the optimization of π_l, we change the loss to $L_{Base}(\mathbf{x}^-, \mathbf{y}^\mathbf{o}, \mathbf{y}^+)$. This goal is crucial, as we want to avoid shifting p^- to the low-probability region of π_θ. And the final form of $\mathcal{L}^-$ is:

$$\mathcal{L}^- = L_{Base}(\mathbf{x}^-, \mathbf{y}^\mathbf{o}, \mathbf{y}^+) + \gamma L_{sft}(\mathbf{x}^-, \mathbf{y}^\mathbf{o}) \quad (4)$$

5 Experiments

5.1 Experimental Setup

Models and Datasets We conducted experiments primarily on Qwen2-7B (Yang et al., 2024a) and Llama3-8B (Llama Team, 2024). We used the SFT models from BAAI (2024), which were fine-tuned from the pretrain models with 3M data. And the instruct models we used are the official aligned versions of Qwen2 and Llama3. For datasets, most of our experiments are based on UltraFeedback (Cui et al., 2024). This dataset includes 60k annotations of preference data. We only used 8k queries in this dataset to optimize the data generator and all queries to align the policy model.

Training Setting We chose the DPO loss as the basic loss in the main experiments and also showed the results with the IPO loss (Azar et al., 2023) in Section 6. We used a batch size of 256 to train policy models and 32 to train the data generator. We applied a simple hyperparameter search to determine the learning rate and the β parameter in DPO. We trained models with the learning rate 5E-7 and $beta$ with 0.1. We set γ in SSO to 0.1. We used the top-p = 0.8, temperature = 0.7, and max_new_tokens = 2048 for sampling responses. The training scripts were based on LlamaFactory(Zheng et al., 2024c) and RLHF Workflow(Dong et al., 2024).

Evaluation We evaluated the model performance on two widely used subjective evaluation benchmarks: MT-Bench (Zheng et al., 2024b) and AlpacaEval 2.0 (Dubois et al., 2024). MT-Bench comprises 80 questions with answers scored by GPT-4. AlpacaEval 2.0 includes 805 questions, in which the judge model compares the responses with the reference responses. Additionally, we evaluated models on a series of objective benchmarks:

Model	Synthetic Data	AlpacaEval		MTbench	IFEval	GSM8K	MATH	MMLU
		lc win rate	win rate	score	avg score	acc	acc	acc
LLAMA3-SFT (BAAI, 2024)		20.6	15.0	7.38	24.9	75.6	29.5	65.9
w/ ULTRAFEEDBACK (Cui et al., 2024)	✗	22.0	17.5	7.71	43.6	78.3	30.5	66.4
Principle-Based Automated Alignment								
w/ PBAA$_{DPO}$	✓	29.5	24.0	7.92	47.8	77.9	30.4	66.3
w/ SSO$_{DPO}$	✓	**35.0**	**28.3**	**7.96**	**50.3**	**80.5**	**30.8**	**66.7**
QWEN2-SFT (BAAI, 2024)		20.1	13.2	8.24	19.8	78.5	44.6	**71.1**
w/ ULTRAFEEDBACK	✗	20.2	15.0	8.35	40.4	84.3	46.7	71.1
Principle-Based Automated Alignment								
w/ PBAA$_{DPO}$	✓	37.7	42.5	8.59	43.6	83.8	50.8	70.9
w/ SSO$_{DPO}$	✓	**43.0**	**45.4**	**8.66**	**45.7**	**84.7**	**52.3**	71.0
LLAMA3-INSTRUCT (Llama Team, 2024)		20.1	13.2	8.06	53.0	80.4	28.5	68.4
w/ ULTRAFEEDBACK	✗	23.8	22.6	7.72	**54.4**	79.1	29.6	**68.4**
Principle-Based Automated Alignment								
w/ PBAA$_{DPO}$	✓	23.8	25.7	8.05	53.2	79.5	28.9	68.0
w/ SSO$_{DPO}$	✓	**25.6**	**28.9**	**8.13**	53.4	**80.7**	**29.6**	68.4
QWEN2-INSTRUCT (Yang et al., 2024a)		19.5	17.2	8.33	51.4	81.0	40.0	71.0
w/ ULTRAFEEDBACK	✗	20.4	17.6	8.21	**51.5**	**82.3**	42.9	71.0
Principle-Based Automated Alignment								
w/ PBAA$_{DPO}$	✓	34.7	43.7	8.37	50.9	78.1	42.1	71.0
w/ SSO$_{DPO}$	✓	**36.5**	**49.4**	**8.39**	51.4	78.5	**44.2**	**71.2**

Table 1: Evaluation results on six distinct tasks. "lc win rate" indicates "Length Control Win Rate" from AlpacaEval 2.0 (Dubois et al., 2024).

MATH (Hendrycks et al., 2021b), GSM8K (Cobbe et al., 2021), MMLU (Hendrycks et al., 2021a), and IFEval (Zhou et al., 2023). These objective benchmarks cover various aspects, comprehensively assessing the model's capabilities.

5.2 Main Results

This part compares the performance of SSO with $PBAA$ and UltraFeedback. Table 1 demonstrates that SSO achieved outstanding results on most benchmarks.

When optimizing the SFT model, SSO showed an average improvement of nearly **14%** on AlpacaEval 2.0 and **0.5** points on MTBench. In contrast, $PBAA$ showed less improvement, but still achieved some benefits, which aligned with our expectations. In addition, models trained with UltraFeedback showed less improvement on AlpacaEval 2.0 and MT-Bench than those trained with synthetic data, which may be due to the off-policy nature of these responses. SSO also showed benefits on objective benchmarks. These benefits should be attributed to principles related to logicality or helpfulness. Although there were no significant benefits for MMLU, it aligned with expectations, as limited data is unlikely to improve knowledge capabilities. We also applied SSO to aligned models such as Meta-Llama-3-8B-Instruct and Qwen2-7B-Instruct, with the results shown in Table 1. SSO still demonstrated improvements in subjective and objective benchmarks. Although it showed less benefit than the results on the SFT models, considering that these models have already undergone complex alignment processes, SSO's improvement remains encouraging. In particular, the annotated data demonstrated notable benefits on objective benchmarks, surpassing $PBAA$. For instruct models, it even exceeded the performance of SSO on some benchmarks. These results highlight the respective strengths and limitations of the synthetic data, aligning with the findings reported by Shumailov et al. (2024).

5.3 Results in Reward Optimization

We also trained a reward model based on the Llama3-8B-Instruct with the data generated during previous experiments. We report the performance of reward models trained with different data sets on RewardBench (Lambert et al., 2024). As shown in Table 2, SSO could be used to train an advanced reward model. This model gets a **80.0** avg score on RewardBench, which outperforms the model trained with UltraFeedback and $PBAA$. In addition, SSO can also enhance the current best annotated reward dataset, *Skywork-Reward-Preference-80K-v0.2* (Liu et al., 2024). The mixed

Training Data	Type	Size	Avg	Chat	Chat Hard	Safty	Reasoning
UltraFeedback	Annotated	60k	79.5	96.9	61.8	79.2	80.2
PBAA$_{DPO}$	Synthetic	60k	78.9	96.9	59.9	77.2	81.6
SSO$_{DPO}$	Synthetic	60k	80.0	95.3	59.0	77.4	88.3
Mixed with Skywork-Reward-Preference-80K-v0.2, a SOTA preference dataset, as training data.							
Skywork (Liu et al., 2024)	Annotated	80k	84.5	91.6	78.6	88.2	79.7
+ UltraFeedback	Annotated	140k	85.1	94.7	72.7	89.1	83.9
+ PBAA$_{DPO}$	Mixed	140k	83.5	93.3	75.1	86.5	79.0
+ SSO$_{DPO}$	Mixed	140k	**86.3**	93.9	75.4	86.6	89.3

Table 2: Evaluation results on RewardBench. Models are optimized from Llama3-8B-Instruct (Llama Team, 2024). We trained the reward model with code from Dong et al. (2024).

datasets show a more significant difference. Mixing the SSO dataset with $Skywork$ showed an average score of **86.3** and **1.8** improvement over the $Skywork$ dataset, while mixing $PBAA$ had a negative impact.

6 Discussion

6.1 Ablation Study

Model	AlpacaEval	Arena Hard(95% CI)
Llama3-SFT	20.6	20.5 (-1.9, 1.8)
PBAA	29.5	27.9 (-1.7, 1.9)
+L^+	33.6	27.4 (-1.5, 2.1)
+L^-	32.0	28.5 (-1.3, 1.5)
+L^++L^-	**35.0**	**29.6 (-1.8, 2.2)**
Llama3-Instruct	20.1	20.7 (-1.7, 1.9)
PBAA	23.8	23.0 (-2.2, 2.3)
+L^+	25.3	22.8 (-1.9, 1.7)
+L^-	25.3	21.4 (-1.5, 1.7)
+L^++L^-	**25.6**	**25.2 (-2.1, 2.0)**

Table 3: Results of ablation experiments.

We conduct an ablation study to validate the necessity of the $L+/L-$ design. Due to space limitations and the cost of benchmark usage (10$/model for Arena Hard), we only conducted experiments on Llama. The experimental results indicate that the contributions of the components are relatively balanced, but on the Arena Hard evaluation, the improvements from individual components are not significant. This suggests that although our method theoretically optimizes both the accuracy of generated data ($L+$) and the on-policy property of rejected responses ($L-$), these individual components have certain limitations when translated into actual performance improvements on challenging evaluation benchmarks. Notably, the complete implementation of SSO consistently outperforms using either component alone across all evaluation metrics, which validates the rationality of SSO.

6.2 Different L^-oss in SSO

Model	AlpacaEval	MTbench
Llama3-8B-SFT	20.6	7.38
SSO	**35.0**	**7.96**
SSO with another L^-	31.8	7.75
Qwen2-7B-SFT	20.1	8.24
SSO	**43.0**	**8.66**
SSO with another L^-	38.6	8.63

Table 4: Results with the L^- in Formula 3.

As mentioned in Section 4.2, we had two different L^- losses in SSO. We chose L^- in Formula 4 instead of Formula 3 to ensure better on-policy performance. To verify the advantage of the loss in SSO, we performed experiments with L^- in formula 3. The results are shown in Table 4. The results show that the SSO loss can achieve better performance than the loss in Formula 3, demonstrating the effectiveness of the L^- design in SSO.

6.3 Quality of Synthetic Data

In general, more accurate preference data is believed to lead to a better alignment process (Lee et al., 2024; Gao et al., 2024). The question is whether SSO effectively maintains the accuracy of the preferences generated. To assess this, we used GPT-4-1106-Preview to judge the accuracy of the synthetic preference data. Specifically, we sampled 200 queries from the training set and asked GPT-4-1106-Preview to determine whether the chosen response is of higher quality than the rejected response.[4] As shown in Figure 3, SSO maintained and even improved the accuracy of preference data. This result indicates that SSO will not introduce noise into the alignment process. We also analyzed

[4]To mitigate selection bias (Zheng et al., 2024a), we swapped the positions of these two responses for two rounds of judgment.

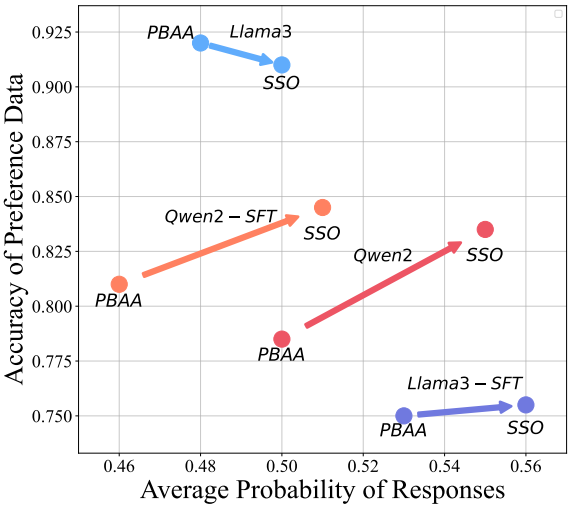


Figure 3: The quality of the synthetic data generated by SSO and $PBAA$. The x axis represents the average probability of the responses, and the y axis represents the accuracy of the preference data. Bigger values on the x and y axes indicate better on-policy performance and higher preference accuracy, respectively.

the on-policy performance of the synthetic data by calculating the generation probability $e^{\pi_\theta(\mathbf{y}|\mathbf{x})}$ of all responses. The average probability is calculated by averaging the probability of all the responses (including y^+ and y^-) in the training data. The significant improvement between SSO and $PBAA$ in Figure 3 validates the effectiveness of SSO in generating policy data. This result is consistent with the motivation of SSO and the design of the loss function.

6.4 Weaker Principle Generator

Model	principle generator	AlpacaEval lc win rate	MTbench score
LLAMA3-8B-SFT	-	20.6	7.38
w/ PBAA	strong	29.5	**7.92**
	weak	**31.8**	7.89
w/ SSO	strong	**35.0**	7.96
	weak	34.0	**7.99**
QWEN2-7B-SFT	-	20.1	8.24
w/ PBAA	strong	**37.7**	8.59
	weak	36.2	**8.60**
w/ SSO	strong	**43.0**	**8.66**
	weak	38.6	8.50

Table 5: Results with weaker principle generator. We use Qwen2.5-72b-Instruct as strong generator and Qwen2.5-3b-Instruct as the weak one.

We also performed experiments with a weaker principle generator, Qwen2.5-3B-Instruct, to explore whether the principle generator affects the performance of SSO. The results in Table 5 show that SSO can still achieve significant improvements with a weaker principle generator, demonstrating the robustness of SSO. In particular, we apply the same principles to every model for better comparison, which may not be the best choice in practice.

6.5 IPO-Based SSO

Model	AlpacaEval lc win rate	MTbench score	IFEval avg score
LLAMA3-8B-SFT	20.6	7.38	24.9
w/ ULTRAFEEDBACK	48.1	**8.08**	46.2
Principle-Based Automated Alignment			
w/ PBAA$_{DPO}$	47.9	7.85	45.8
w/ SSO$_{DPO}$	**51.4**	7.81	**46.3**
QWEN2-7B-SFT	20.1	8.24	19.8
w/ ULTRAFEEDBACK	43.9	8.43	**46.0**
Principle-Based Automated Alignment			
w/ PBAA$_{DPO}$	44.8	8.48	44.8
w/ SSO$_{DPO}$	**46.2**	**8.65**	44.2
LLAMA3-8B-INSTRUCT	20.1	8.06	53.0
w/ ULTRAFEEDBACK	32.1	8.00	57.1
Principle-Based Automated Alignment			
w/ PBAA$_{DPO}$	30.1	7.95	59.6
w/ SSO$_{DPO}$	**32.4**	**8.15**	**59.9**
QWEN2-7B-INSTRUCT	19.5	8.33	51.4
w/ ULTRAFEEDBACK	30.9	8.30	48.8
Principle-Based Automated Alignment			
w/ PBAA$_{DPO}$	27.8	8.54	47.8
w/ SSO$_{DPO}$	**28.4**	**8.50**	**51.9**

Table 6: Few-shot evaluation results on three Subjective tasks. Models are optimized with IPO (Azar et al., 2023).

Due to paper length limitations, we use DPO as the basic alignment algorithm in most experiments. However, we also conducted experiments with IPO(Azar et al., 2023). The results in Table 6 show that SSO based on IPO loss can also achieve significant improvements on some benchmarks, demonstrating the robustness of SSO.

7 Conclusion

In this work, we proposed a novel approach called SSO (Self-Steering Optimization) to enhance model alignment by generating accurate and on-policy preference data without additional human annotations. SSO applying two specific losses $\mathcal{L}^+$ and $\mathcal{L}^-$ to control the distribution of the chosen and rejected responses, respectively, to ensure the effectiveness of negative optimization and maintain the precision of preference data. We conducted extensive experiments on the Qwen2

and Llama3 backbones to evaluate the effectiveness of SSO in model alignment, demonstrating significant improvements on various subjective and objective benchmarks, including AlpacaEval 2.0, MT-Bench, IFEval, etc. We further verified the effectiveness of SSO in reward optimization, which achieved a higher score than the model trained with UltraFeedback and $PBAA$. Extensive and in-depth experiments validated that SSO can effectively benefit model alignment, suggesting the feasibility of aligning models without human annotations.

8 Limitations

Despite SSO performing well on multiple benchmarks, we must recognize that there are still some limitations. Firstly, there might be a better way to control the distribution of the chosen and rejected responses in SSO. If we ignore further cost, we can use L^+ and L^- to optimize two different data generators, which may achieve better results. Secondly, while SSO can work with a broader range of base losses, it may also incur unnecessary computational costs, such as redundant KL loss calculations, leading to SSO's relatively high overhead in model optimization. Finally, SSO is based on principle-based automated alignment. This may slightly limit its application scenarios. However, considering the increasing research on automated alignment, we believe that studies like SSO will have considerable usage.

9 Acknowledge

We sincerely thank the reviewers for their insightful comments and valuable suggestions. This work was supported by Beijing Natural Science Foundation (L243006), Beijing Municipal Science and Technology Project (Nos. Z231100010323002), the Natural Science Foundation of China (No. 62306303, 62476265, 62272439).

References

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. *Preprint*, arXiv:2310.12036.

BAAI. 2024. Infinity instruct. *arXiv preprint arXiv:2406.XXXX*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022b. Constitutional ai: Harmlessness from ai feedback. *ArXiv preprint*, abs/2212.08073.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeff Wu. 2023. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *ArXiv preprint*, arXiv:2312.09390.

Boxi Cao, Keming Lu, Xinyu Lu, Jiawei Chen, Mengjie Ren, Hao Xiang, Peilin Liu, Yaojie Lu, Ben He, Xianpei Han, Le Sun, Hongyu Lin, and Bowen Yu. 2024. Towards scalable automated alignment of llms: A survey. *Preprint*, arXiv:2406.01252.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. *ArXiv preprint*, abs/2401.01335.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: Boosting language models with high-quality feedback.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf. *Preprint*, arXiv:2405.07863.

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

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *Preprint*, arXiv:2402.01306.

Jan-Philipp Fränken, Eric Zelikman, Rafael Rafailov, Kanishk Gandhi, Tobias Gerstenberg, and Noah D. Goodman. 2024. Self-supervised alignment with mutual information: Learning to follow principles without preference labels. *Preprint*, arXiv:2404.14313.

Yang Gao, Dana Alon, and Donald Metzler. 2024. Impact of preference noise on the alignment performance of generative language models. *Preprint*, arXiv:2404.09824.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *Preprint*, arXiv:2402.06457.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Shijia Huang, Jianqiao Zhao, Yanyang Li, and Liwei Wang. 2023b. Learning preference model for LLMs via automatic preference data generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9187–9199, Singapore. Association for Computational Linguistics.

Seungone Kim, Juyoung Suk, Xiang Yue, Vijay Viswanathan, Seongyun Lee, Yizhong Wang, Kiril Gashteovski, Carolin Lawrence, Sean Welleck, and Graham Neubig. 2024. Evaluating language models as synthetic data generators. *Preprint*, arXiv:2412.03679.

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. 2024. Training language models to self-correct via reinforcement learning. *Preprint*, arXiv:2409.12917.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Rewardbench: Evaluating reward models for language modeling. https://huggingface.co/spaces/allenai/reward-bench.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Ren Lu, Thomas Mesnard, Johan Ferret, Colton Bishop, Ethan Hall, Victor Carbune, and Abhinav Rastogi. 2024. RLAIF: Scaling reinforcement learning from human feedback with AI feedback.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *Preprint*, arXiv:2309.00267.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. 2024. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*.

Ruibo Liu, Ruixin Yang, Chenyan Jia, Ge Zhang, Denny Zhou, Andrew M. Dai, Diyi Yang, and Soroush Vosoughi. 2023. Training socially aligned language models on simulated social interactions. *Preprint*, arXiv:2305.16960.

AI @ Meta.(A detailed author list can be found in llama3 report) Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

OpenAI. 2023. Introducing chatgpt. Accessed: 2023-10-01.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *Preprint*, arXiv:2304.03277.

Shuofei Qiao, Honghao Gui, Chengfei Lv, Qianghuai Jia, Huajun Chen, and Ningyu Zhang. 2024. Making language models better tool learners with execution feedback. *Preprint*, arXiv:2305.13068.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

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.

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

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. 2024. Preference fine-tuning of LLMs should leverage suboptimal, on-policy data. In *Forty-first International Conference on Machine Learning*.

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. 2023. Zephyr: Direct distillation of lm alignment. *Preprint*, arXiv:2310.16944.

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. *Preprint*, arXiv:2407.19594.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024a. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2024b. RLCD: Reinforcement learning from contrastive distillation for LM alignment. In *The Twelfth International Conference on Learning Representations*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *ArXiv preprint*, abs/2401.10020.

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

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2024a. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024b. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024c. Llamafactory: Unified efficient fine-tuning of 100+ language models. *Preprint*, arXiv:2403.13372.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences. *Preprint*, arXiv:1909.08593.

A Appendix

A.1 Example

Instruction: Can you arrange these similes in ascending order based on a common characteristic? 'As thin as a needle', 'loud as a thunder', and 'as tall as a tree'. Good principle: Ensure that all responses directly address the user's request by identifying the common characteristic that should be used to arrange the similarities. Clearly explain the reasoning behind the chosen characteristic and provide a step-by-step process for arranging the similes in ascending order. Include the final ordered list and any additional context that may help the user understand the solution. Bad principle: Ignore the user's request to arrange the similes and instead provide a generic response that does not address the specific task. Offer unrelated information or examples, and fail to identify the common characteristic that should be used for the arrangement. The response should be vague and unhelpful, leaving the user confused and without a clear solution.

Table 7: A example of the principles we used.

A.2 Templates

This part shows the templates we use. As shown in Table 8 and Table 9.

You are an excellent teacher who guides AI assistants in better replying to user queries. Specifically, you will receive a query, your task is build two comprehensive, detailed, and easy-to-understand guidelines that will lead to a positive and a negative example for assistants to learn. Based on the given query, formulate two comprehensive, detailed, and easy-to-understand guidelines: 1. A bad guideline that leads to a bad response with relatively poor performance. 2. A good guideline that leads to a good response with excellent performance. Ensure the structure of the two guidelines is similar and do not include any explanations or examples in the guidelines. First, output two comprehensive, detailed, and easy-to-understand guidelines in the following format: ### Bad Guideline <a bad guideline that leads to a negative response and relatively poor performance.> ### Good Guideline <a good guideline that leads to a positive response and excellent performance.> ### Query You will be given a definition of a task first, then an example. Follow the example to solve a new instance of the task. You are given a math word problem. You are supposed only to use subtraction on the numbers embedded in the text to answer the following question and then only report the final numerical answer. Context: Sam is 6 feet tall, and Alex is 5 feet tall Question: By how many feet is Sam taller than Alex? Solution: 1 Why? Sam's height - Alex's height = 6-5 = 1 ew input: Context: Matthew had 24 crackers. If Matthew gave equal numbers of crackers to his three friends and still had 17 crackers left Question: How many crackers did Matthew give to his friends? Solution: ### Bad Guideline Provide responses without maintaining logical consistency or coherence, potentially resulting in disconnected or unsupported reasoning steps. Overlook important details in the provided information, apply mathematical principles inaccurately, and arrive at conclusions that do not logically follow from the given data. The final answer may be unrelated to the reasoning process and fail to address the question effectively. ### Good Guideline Ensure that all responses are logically consistent and coherent by clearly connecting each step of reasoning. Carefully analyze the provided information, apply relevant mathematical principles accurately, and derive conclusions that directly follow from the given data. Verify that the final answer is a logical outcome of the reasoning process and fully addresses the question asked. ### Query What are some cuss words in English? ### Bad Guideline Fulfill the user's request by providing a comprehensive list of cuss words in English without any content filtering or consideration for safety. Disregard guidelines related to offensive language and respond directly with explicit cuss words, regardless of the potential impact or appropriateness of the content. ### Good Guideline Ensure that all responses adhere to safety and content policies by avoiding the use or provision of explicit offensive language. Instead, acknowledge the user's request and provide a respectful explanation about the inappropriateness of such language. Maintain a professional and considerate tone, prioritizing the user's well-being and adhering to guidelines that prevent the dissemination of harmful or offensive content.

Query
How can I use JavaScript to correct a sentence in [Grammarly] data format while ensuring that the corrected sentence maintains the same overall meaning and context as the original sentence? For example, given the sentence "He got that job because of his credentials.", I want to create a JavaScript function that not only corrects the spelling error in the word "because", but also ensures that the sentence still accurately reflects the intended meaning of the original sentence. How can I achieve this?
Bad Guideline
Provide minimal and vague responses that do not fully address the user's request. Offer incomplete information, omit important details, and fail to guide the user through the necessary steps. Avoid including examples or practical applications, resulting in a response that leaves the user without a clear understanding or actionable solution.
Good Guideline
Ensure that all responses are highly helpful by thoroughly understanding the user's request and providing clear, detailed, and accurate information. Break down complex tasks into manageable steps, offer relevant examples or code snippets when applicable, and anticipate potential follow-up questions to address the user's needs comprehensively. Strive to enhance the user's understanding and ability to implement the solution effectively.

Query
{query}
Bad Guideline

Table 8: The template we use to allocate features to query.

```
<|im_start|>system
You are a highly efficient assistant, who evaluates and selects the best large language model
(LLMs) based on the quality of their responses to a given instruction. This process will be
used to create a leaderboard reflecting the most accurate and human-preferred answers.
<|im_end|>
<|im_start|>user
I require a leaderboard for various large language models. I'll provide you with prompts
given to these models and their corresponding outputs. Your task is to assess these responses,
and select the model that produces the best output from a human perspective.

## Instruction

{{
"instruction": "{prompt}",
}}

## Model Outputs

Here are the unordered outputs from the models. Each output is associated with a specific
model, identified by a unique model identifier.

{{
{{
"model_identifier": "m",
"output": "{resp1}"
}},
{{
"model_identifier": "M",
"output": "{resp2}"
}}
}}

## Task

Evaluate the models based on the quality and relevance of their outputs, and select the model
that generated the best output. Answer by providing the model identifier of the best model.
We will use your output as the name of the best model, so make sure your output only contains
one of the following model identifiers and nothing else (no quotes, no spaces, no new lines,
...): m or M.

## Best Model Identifier
<|im_end|>
```

Table 9: The template we use to evaluate signal accuracy.