Step-by-Step Mastery: Enhancing Soft Constraint Following Ability of Large Language Models

Anonymous ACL submission

Abstract

It is crucial for large language models (LLMs) to follow instructions that involve multiple constraints. However, many soft constraints are semantically related and difficult to verify through automated methods. These constraints remain a significant challenge for LLMs. To enhance the ability of LLMs to follow soft constraints, we initially design a pipeline to obtain high-quality outputs automatically. Additionally, to fully utilize the acquired data, we introduce a progressive training paradigm based on curriculum learning. We experimentally evaluate the effectiveness of our methods in improving LLMs' soft constraint following ability and analyze the factors driving the improvements. To support further research, we will release the code and data associated with this study.

1 Introduction

In the application of LLMs, generating responses that accurately satisfy user requests, known as instruction following ability, is of paramount importance (Lou et al., 2024). The capability of LLMs plays a critical role in aligning LLMs with human preferences, ensuring the reliability and helpfulness of the model's outputs (Wang et al., 2023b; Song et al., 2024).

It is a significant challenge for LLMs to follow instructions with multiple constraints (Jiang et al., 2023b; Qin et al., 2024). Existing work on improving the ability of LLMs to follow multiple constraints mainly focuses on hard constraints, which are typically based on structured data or fixed-format requirements (He et al., 2024a). These constraints can be explicitly expressed as specific rules and directly verified through programming methods (Zhou et al., 2023a). For example, Python can parse JSON to verify hard constraints. However, hard constraints fail to adequately capture the complexity in real-world scenarios shown in Fig. 1. Instructions in real-world applications often contain semantic-level limitations, which can be categorized as soft constraints. Soft constraints include restrictions related to content (Liang et al., 2024; Zhang et al., 2023), specific backgrounds (Shanahan et al., 2023; Liu et al., 2023), and stylistic objectives (Sigurgeirsson and King, 2024; Mukherjee et al., 2024). They are both widespread and critically important. A variety of tasks involve soft constraints, such as open-ended question answering (Zhuang et al., 2023), role-playing (Shanahan et al., 2023), and suggestion generation (Baek et al., 2024). As shown in Fig. 1, following soft constraints is challenging for LLMs.

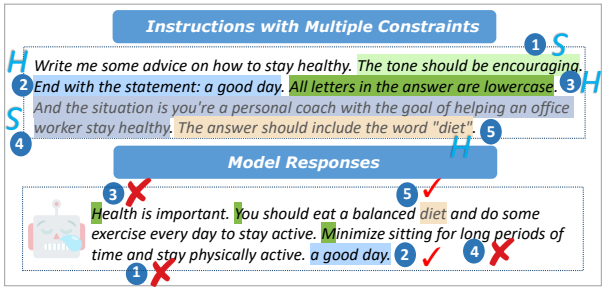

Figure 1: In real-world scenarios, user instructions contain many soft constraints, posing challenges for LLMs. H and S denote hard constraints and soft constraints, respectively.

However, following soft constraints is a non-trivial task. First, existing research on soft constraints in LLMs mainly focuses on evaluation (Chen et al., 2024a; Qin et al., 2024) rather than improving their following. Also, as shown in Fig. 1, soft constraints are ambiguous and challenging for LLMs in real applications (Wang et al., 2024). They depend on subjective interpretations and specific contexts. Unlike hard constraints, they cannot be assessed with fixed rules or scripts. Soft constraint evaluation often relies on prompting LLMs, which involves various biases (Wang et al., 2023a). The inherent difficulty makes it more challenging for LLMs to generalize from hard to

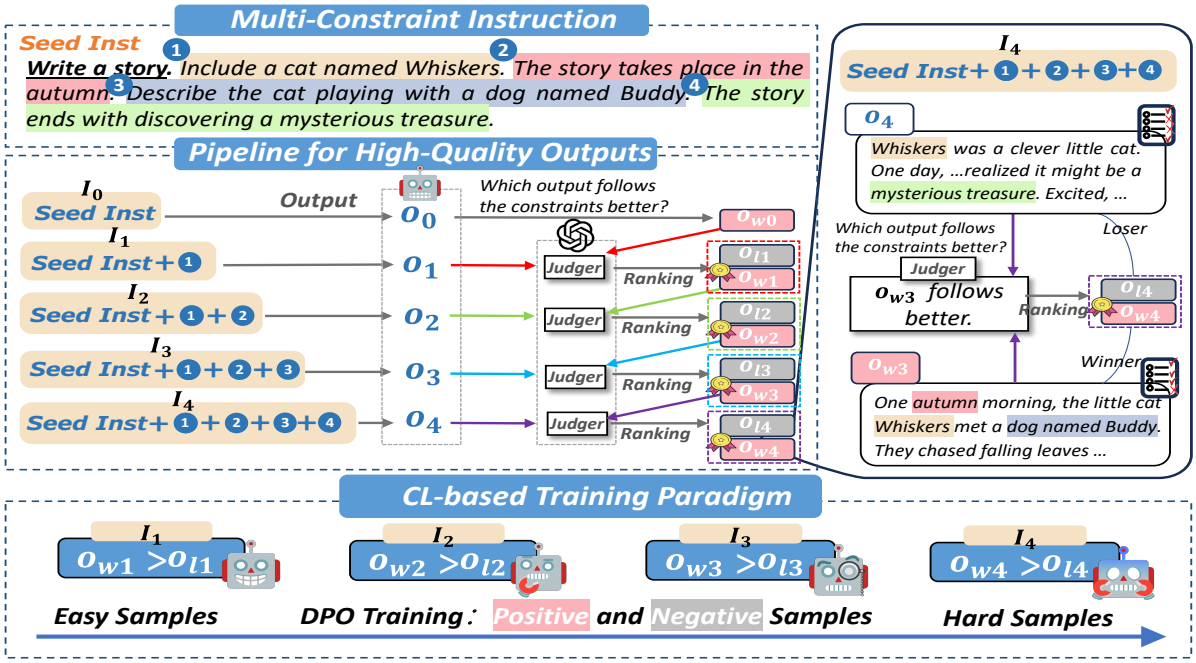

Figure 2: The framework of our study. We first design a pipeline that automates the construction of datasets with high-quality outputs for soft constraint following. Then, we propose a method to utilize positive and negative samples generated during the pipeline. Finally, we introduce a new training paradigm that leverages curriculum learning to enhance LLMs' soft constraint following ability. CL denotes curriculum Learning.

soft constraints (He et al., 2024a). Moreover, many studies utilize advanced models, such as GPT-4, to generate responses (Xu et al., 2023; Chiang et al., 2023). Soft constraints also present significant challenges for these advanced models. On the FollowBench benchmark (Jiang et al., 2023b), GPT-4 demonstrates a hard satisfaction rate of merely 74.4%, making the assurance of high-quality training data difficult. However, many studies show that the quality of training data is more important than its quantity (Zhou et al., 2024; Li et al., 2023a). Therefore, designing a more effective data construction pipeline is crucial.

In this work, we systematically investigate strategies to enhance the ability of LLMs to follow instructions with soft constraints, with the framework shown in Fig. 2. When more constraints are added to the instruction, LLMs' outputs may not fully follow certain constraints. This inconsistency can hurt the quality of data. To address this, we incorporate Judger to rank the outputs based on the extent of adherence to the instructions to obtain high-quality outputs. To fully utilize both positive and negative outputs during the ranking process, we leverage the preference learning algorithm Direct Preference Optimization (DPO) (Rafailov et al., 2024) as the training method. Subsequently, we propose a novel training paradigm that constructs a curriculum based on the number of constraints in the instruction. In this framework, the model progressively learns how to make preference judgments, beginning with easier cases and moving towards more challenging ones. Our methods improve the model's soft constraint following ability while maintaining general capabilities.

Our contributions are summarized as follows: (1) We design a pipeline that automates the construction of datasets with high-quality outputs for soft constraint following. We also propose a method that utilizes positive and negative samples generated during the pipeline. (2) We introduce a new training paradigm that leverages curriculum learning to enhance LLMs' soft constraint following ability. (3) We conduct extensive experiments to validate the effectiveness of our methods and analyze the reasons for the performance improvement.

2 Related Work

Soft Constraint Following Existing research on soft constraint following largely concentrates on evaluating the ability of LLMs to follow these constraints by constructing benchmarks (Jiang et al., 2023b; Qin et al., 2024). These benchmarks typically include a variety of fine-grained constraint

types (Zhang et al., 2024), and the results from testing LLMs on these benchmarks suggest that LLMs often struggle to follow these constraints (He et al., 2024b). Despite this, there is a notable paucity of research aimed at improving LLMs' capacity to comply with soft constraints, especially for soft constraints. Soft constraints can be categorized into several types: (1) Content soft constraints involve restrictions on the scope or depth of the responses (Zhou et al., 2023b; Ashok and Poczos, 2024). (2) Situation soft constraints refer to the background limitations of the responses (Wang et al., 2023c; Shao et al., 2023). (3) Style soft constraints limit the manner or tone of expressions (Tao et al., 2024; Pu et al., 2024). Some works directly utilize responses generated by GPT-4 to construct datasets (Sun et al., 2024; Peng et al., 2023). However, the responses to instructions with soft constraints are often unreliable. Different from these, our study focuses on how to construct datasets with high-quality outputs for improving LLMs' soft constraint following ability.

Curriculum Learning Curriculum learning is a training strategy that mimics the learning process of humans by advancing from simpler to more complex tasks (Soviany et al., 2022; Wang et al., 2021). Current research on LLMs' curriculum learning can be broadly categorized into two primary paradigms: (1) Learning Based on Data Difficulty: This approach involves constructing curricula by ranking data according to various evaluation metrics. Metrics such as sequence length (Pouransari et al., 2024), perplexity (Liu et al., 2024) have been employed to guide this process. LLMs can also construct curricula through advanced planning (Ryu et al., 2024). (2) Learning Based on Task Difficulty: This paradigm focuses on modifying the training tasks (Chen et al., 2024b) or adjusting the training objectives (Zhao et al., 2024b; Lee et al., 2024). However, our work organizes the curriculum based on the number of constraints in the instructions.

3 Method

In this section, we provide a detailed explanation of how to obtain high-quality data and how to leverage this data by establishing a new training paradigm. The pipeline is shown in Fig. 2.

3.1 High-quality Data Construction

We first synthesize multi-constraint instructions and then utilize Judger to rank the LLM's outputs of these instructions, enhancing the reliability of the dataset.

3.1.1 Multi-Constraint Instruction Synthesis

To generate complex instructions, we initially gather seed instructions from three commonly utilized datasets. Next, these instructions are rewritten to integrate multiple constraints.

We begin by collecting seed instructions from Open Assistant (Köpf et al., 2024), which includes instructions generated by users interacting with chatbots. We select rank 0 instructions and those from the first turn of conversations. Next, we gather 175 manually created instructions from the Self-Instruct (Wang et al., 2022a). The third source is Super-Natural (Wang et al., 2022b), from which we select 318 instructions after filtering out tasks with simple outputs. These three sources together provide a total of 1,500 seed instructions, offering a broad range of coverage across diverse tasks.

Subsequently, we construct soft constraints and integrate them into the seed instructions. Initially, we categorize the soft constraints into three types: content, situation, and style. Using an advanced model, we generate soft constraints based on these categories. Next, we randomly select 3 to 5 constraints for each seed instruction. For the soft constraints, GPT-4 is employed to generate corresponding descriptions. While descriptions are selected from a predefined list for the hard ones. Finally, we add only one constraint to the instruction at a time, ensuring that each instruction reflects a different level of difficulty. This approach contrasts with previous methods, which typically add all constraints at once, often making it challenging for the model to learn how to follow each constraint independently (He et al., 2024a). The construction process is detailed in the Appx. A.1.

Specifically, for seed instruction I_0, we iteratively add constraints to form the instruction set $I = \{I_1, I_2, \ldots, I_n\}$, where n ranges from 3 to 5. I_n represents the instruction with n constraints.

3.1.2 Judger for Ranking Responses

Existing works in dataset construction rely on advanced models to directly generate the outputs (Sun et al., 2024). However, even GPT-4 is struggling to follow the instructions with complex constraints, especially when the instructions contain soft con-

straints which are more intractable (Jiang et al., 2023b; Qin et al., 2024). To address this, we design a pipeline to construct datasets with high-quality outputs for soft constraints following. This pipeline consists of two steps: progressively construction and Judger reordering.

When all constraints are added into the instruction at once, it is difficult for large models to effectively follow each constraint individually. This approach is not conducive to improve the model's constraint following ability. To address this, we employ a method of incrementally adding one constraint at a time and performing step-by-step reasoning to obtain the dataset. Specifically, for each multi-constraint instruction I_k, we use GPT-4 to generate the corresponding output $O_k = \text{LLM}(I_k)$. After performing inference on all the instructions in the instruction set I, we obtain the output set $O = \{O_1, O_2, \ldots, O_n\}$, where each response corresponds to a multi-constraint instruction.

Given that LLMs may not follow all constraints when responding to multi-constraint instructions, we introduce Judger to rank the outputs based on their following of the instructions, thereby obtaining high-quality outputs . Specifically, when a new constraint is added into the instruction I_{k-1} to form I_k , the model's response O_k may not fully follow some constraints in I_k. To ensure high-quality outputs, we use Judger to rank the new output O_k with the previous winner output $O_{w_{k-1}}$ that more follows I_{k-1} to determine which one better follows the current instruction I_k:

$$O_{w_k}, O_{l_k} = Judger(I_k, O_{w_{k-1}}, O_k), \quad 1 \leq k \leq n.$$

In each ranking, we can obtain winner output O_{w_k} which follows the current instruction I_k better and the loser output O_{l_k} which follows less. Finally, after completing all n rankings, we obtain the winner set $O_w = \{O_{w_1}, O_{w_2}, \ldots, O_{w_n}\}$, which consists of the outputs that most follow their respective instructions, and the loser set $O_l = \{O_{l_1}, O_{l_2}, \ldots, O_{l_n}\}$, which contains the outputs that less follow.

The prompt used and cases are detailed in the Appx. A.2.

3.2 Curriculum-based Training Paradigm

In §3.1.2, we use Judger to obtain the positive set O_w and the negative set O_l. Supervised Fine-Tuning (SFT) (Ouyang et al., 2022) only uses the positive samples to train the model. However, the negative samples also contain valuable supervision information. Hence, we adopt reinforcement learning (Rafailov et al., 2024) to leverage both the positive and negative sets. Moreover, we develop a training paradigm based on curriculum learning to enhance the training process.

Given the positive set and the negative set, we can construct the training dataset with k triplets: $(I_1, O_{w_1}, O_{l_1}), (I_2, O_{w_2}, O_{l_2}), \ldots, (I_k, O_{w_k}, O_{l_k})$. In each triplet, the output from O_w is preferred than the output from O_l. To model this preference relationship, we apply Direct Preference Optimization (DPO) (Rafailov et al., 2024) as the training method.

Additionally, in the DPO training process, the model is required to learn preference judgments. As the number of constraints in the instruction increases, the complexity of judgments also rises. Inspired by curriculum learning (Bengio et al., 2009), we propose a curriculum learning training approach for preference learning, where the training dataset is organized in ascending order based on the number of constraints in the instructions.

Specifically, for the k-th curriculum, the training dataset D_k contains the triplet (I_k, O_{w_k}, O_{l_k}). The constraint set C_k contains k constraints in I_k: $D_k = \{(I_k, O_{w_k}, O_{l_k}) \mid |C_k| = k\}$. The complete training dataset D is obtained by combining training datasets for all curriculums in sequence: $D = D_1 \cup D_2 \cup D_3 \cup D_4 \cup D_5$.

Based on the preference data and the curriculum-based training paradigm, the loss function of DPO training can be defined as follows:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(I_k, O_{w_k}, O_{l_k}) \sim D} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(O_{w_k}|I_k)}{\pi_{\text{ref}}(O_{w_k}|I_k)} - \beta \log \frac{\pi_\theta(O_{l_k}|I_k)}{\pi_{\text{ref}}(O_{l_k}|I_k)} \right) \right]$$

where π_θ represents the current model, and π_{ref} denotes the reference model.

To ensure training stability (Xu et al., 2024), we add the SFT loss into the DPO loss function:

$$\mathcal{L}_{\text{DPO}} = \mathcal{L}_{\text{DPO}} + \mathcal{L}_{\text{SFT}}$$

where SFT loss is as follows:

$$\mathcal{L}_{\text{SFT}}(\pi_\theta) = -\mathbb{E}_{(I_k, O_{w_k}) \sim D}[\log \pi_\theta(O_{w_k}|I_k)]$$

3.3 Dataset Statistics

3.3.1 Diversity

To show the diversity of our dataset, we analyze the verb-noun structure of data. As shown in Fig. 3,

Model	BaseModel	FollowBench (HSR)						IFEval				
		L1	L2	L3	L4	L5	Avg	[S]P	[S]I	[L]P	[L]I	Avg
GPT4 (Achiam et al., 2023)*	GPT	84.7	75.6	70.8	73.9	61.9	73.4	76.9	83.6	79.3	85.4	81.3
GPT3.5-turbo*	GPT	80.3	68.0	68.6	61.1	53.2	66.2	-	-	-	-	-
Llama-3.1-70B-Instruct (Dubey et al., 2024)	LLaMA3	75.2	69.6	63.1	65.9	57.1	66.2	82.1	87.8	85.4	90.0	86.3
Qwen2-72B-Instruct (Yang et al., 2024)	Qwen	67.9	56.6	47.8	42.2	35.3	50.0	77.1	84.4	80.4	86.9	82.2
WizardLM-v1.2-13B (Xu et al., 2023)*	LLaMA2	68.8	**64.1**	53.1	40.8	35.8	52.5	43.6	54.4	48.4	59.1	51.4
Conifer-13B (Sun et al., 2024)	LLaMA2	60.5	53.6	48.4	40.7	31.7	47.0	42.9	53.0	47.5	57.4	50.2
Vicuna-13B-v1.5 (Chiang et al., 2023)*	LLaMA2	**71.2**	60.2	49.6	40.6	34.0	51.1	43.1	53.6	46.6	58.0	50.3
Conifer-7B-SFT (Sun et al., 2024)	Mistral	54.3	49.5	49.3	40.8	30.5	44.9	45.8	57.1	50.8	62.0	53.9
Conifer-7B-DPO (Sun et al., 2024)	Mistral	60.3	53.6	48.0	47.1	**41.0**	50.0	48.1	59.1	52.3	63.3	55.7
Mistral-7B-Instruct-v0.3$_{\text{BASE}}$	Mistral	58.7	50.9	48.5	37.5	27.6	44.6	47.0	58.0	52.1	62.7	55.0
Mistral-7B-Instruct-v0.3$_{\text{SFT}}$	Mistral	58.7	52.4	42.5	37.2	35.6	45.3	56.8	67.8	60.6	71.3	64.1
Mistral-7B-Instruct-v0.3$_{\text{DPO+Judger+CL}}$	Mistral	61.2	52.5	47.5	38.2	33.9	46.7	51.4	62.8	59.0	69.2	60.6
Llama-3-8B-Instruct$_{\text{BASE}}$	LLaMA3	67.8	54.5	46.6	50.6	39.1	51.7	67.5	76.1	72.8	80.9	74.3
Llama-3-8B-Instruct$_{\text{SFT}}$	LLaMA3	69.3	59.0	50.1	44.8	32.0	51.0	68.8	76.6	71.2	78.7	73.8
Llama-3-8B-Instruct$_{\text{DPO+Judger+CL}}$	LLaMA3	70.8	54.6	**55.6**	**51.6**	37.9	**54.1**	**72.5**	**80.1**	**78.0**	**84.5**	**78.8**

Table 1: The overall performance on FollowBench and IFEval. We use boldface for the best results and underline for the second-best results among the models ranging from 7B to 13B parameter sizes. * indicates that the results are directly sourced from the original benchmarks.

Method	Data Quality			
	Nums.	Cons.	Pair.	Open.
Conifer (Sun et al., 2024)	13600	H/S	×	✓
Suri (Pham et al., 2024)	10000	S	×	✓
AutoIF (Dong et al., 2024)	-	H	✓	×
Complex to Simple (He et al., 2024a)	1467	H	✓	✓
Ours	17043	H/S	✓	✓

Table 2: Detailed comparison of relevant works. Ours represents our dataset construction approach. 'Nums.', 'Cons.', 'Pair.', and 'Open.' denote the number of samples, constraint types, whether to perform pairwise comparison, and whether the data is open-source.

Figure 3: Verb-noun structure of multi-constraint instructions.

we illustrate the top 15 verbs in the inner circle and their 3 most frequent direct noun objects in the outer circle. This structure visually highlights the variety of actions and their corresponding contexts present in the instructions. The result reveals the instructions encompass a diverse set of linguistic patterns. This diversity of our dataset is crucial for enhancing the model's ability to generalize across various types of constraints.

3.3.2 Comparison with Other Works

As shown in Tab. 2, we compare our dataset construction approach with other related works. In terms of size, our dataset contains more samples compared to others which are open-source. From the perspective of constraint categories, our dataset includes both soft and hard constraints. Compared to datasets that only contain soft or hard constraints, this broader scope better facilitates the model's ability to learn complex constraints. Regarding pairwise com- parison, we use Judger for comparisons of the outputs, which improves the quality of the dataset.

4 Experiments

We conduct extensive experiments to evaluate the effectiveness of our proposed method, focusing on soft constraint following ability and generalization performance.

4.1 Experiment Setup

Models. We conduct experiments on two widely recognized base LLMs, Llama-3-8B-Instruct (Dubey et al., 2024) and Mistral-7B-Instruct-v0.3 (Jiang et al., 2023a), both of which

demonstrate exceptional performance among models within the parameter range of 7B to 8B. Within our experimental framework (§3), we compare three approaches: (1) **BASE** directly utilizes the original model to generate outputs. (2) **SFT** applies supervised fine-tuning on LLMs using constructed data (§3.1.1). (3) **DPO+Judger+CL** utilizes Judger to produce high-quality training data, in accordance with training the model using DPO based on curriculum learning.(§3.1.2,§3.2).

For baseline comparisons, we select a range of open-source and proprietary LLMs. Among the proprietary models, we include GPT-4 (Achiam et al., 2023) and GPT-3.5-turbo. Additionally, we compare our approach with several open-source LLMs, including models specifically trained to improve general instruction-following abilities, such as Vicuna-13B-v1.5 (Chiang et al., 2023). We also include models focused on enhancing the ability to follow soft instructions, such as WizardLM-v1.2-13B (Xu et al., 2023) and the Conifer series (Sun et al., 2024). We also compare our models against two 70B-sized models, Llama-3.1-70B-Instruct (Dubey et al., 2024) and Qwen2-72B-Instruct (Yang et al., 2024), which are among the most powerful models.

Evaluation Benchmarks. IFEval (Zhou et al., 2023a) is a benchmark designed to assess the adherence to hard constraints. It defines 25 distinct types of verifiable instructions and generates approximately 500 prompts, each containing between 1 and 3 constraints. These hard constraints are explicit and unambiguous, enabling programmatic validation of compliance. FollowBench (Jiang et al., 2023b), is a benchmark that evaluates the ability of models to follow both soft and hard constraints across multiple levels of granularity, offering a comprehensive assessment of instruction-following capabilities.

4.2 Main Results

As shown in Tab. 1, our method significantly enhances the model's ability to follow soft constraints, even outperforming the capabilities of larger models. Specifically, when the models are trained using the DPO+Judger+CL method, a significant performance improvement is observed across both benchmarks, particularly on IFEval. The model's performance improvement is particularly significant on complex tasks, especially at the L4-L5 difficulty levels in FollowBench. Specifically, Mistral-7B-Instruct-v0.3 shows an average improvement of 3.5% at the L4-L5 difficulty levels.

Model	BaseModel	LC Win Rate
GPT-4-0613*	GPT	30.2
GPT-3.5-Turbo-0613*	GPT	22.4
Llama-3.1-70B-Instruct-Turbo*	LLaMA3	39.3
WizardLM-13B-v1.2*	LLaMA2	14.5
Vicuna-13B-v1.5*	LLaMA2	10.5
Conifer-7B-DPO*	Mistral	17.1
Llama-3-8B-Instruct$_{\mathbf{BASE}}$	LLaMA3	21.6
Llama-3-8B-Instruct$_{\mathbf{DPO+Judger+CL}}$	LLaMA3	22.0

Table 3: Evaluation on the AlpacaEval2.0 for general LLM instruction-following ability. * indicates that the results are directly sourced from the original leaderboards.

In comparison to models designed to enhance the ability to follow complex instructions, our model demonstrates superior performance on both benchmarks. Specifically, although the performance of Mistral-7B-Instruct-v0.3 on Followench is lower than Conifer-7B-SFT, its performance surpasses the Conifer model on both benchmarks after training. Moreover, our training paradigm effectively enhances the instruction-following ability of LLMs, even when working with models of smaller parameter sizes. Specifically, compared with models in the 13B category, the performance of Llama-3-8B-Instruct is initially weaker than that of WizardLM-v1.2-13B on FollowBench. But after training, its performance surpasses the 13B model on both benchmarks.

After supervised fine-tuning on the constructed instruction-response pairs, the performance of the Llama-3-8B-Instruct model decreases on both benchmarks. This decline can be attributed to the fact that the Llama-3-8B-Instruct model incorporates various specialized training techniques during its initial training.

4.3 Generalization Experiments

Besides the ability to follow soft constraints, we also assess the model's general instruction following abilities on AlpacaEval (Li et al., 2023b). To avoid the length bias that AlpacaEval may correlate with response lengths, we use the AlpacaEval 2.0 (Zhao et al., 2024a) to evaluate the general instruction following.

In our evaluation process, we first perform supervised fine-tuning on the model, followed by DPO training using the proposed training paradigm.

Model	FollowBench (HSR)			IFEval
	L1 - L3	L4 - L5	Avg	Avg
BASE	56.3	44.9	51.7	74.3
SFT	59.5	38.4	51.0	73.8
SFT+Judger	57.3	44.8	52.3	75.4
DPO+Judger	58.8	44.6	53.1	80.7
DPO+Judger+CL	60.3	44.8	54.1	78.8

Table 4: Ablation study results on FollowBench and IFEval.

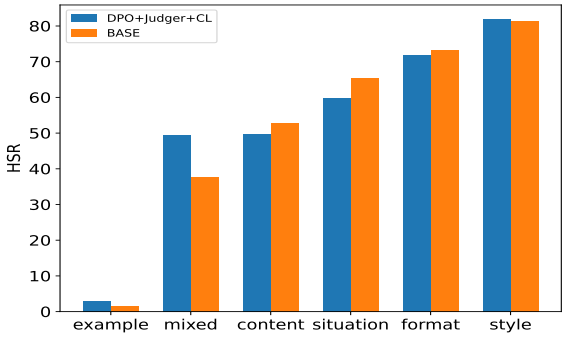

Figure 4: Results across various constraint categories in FollowBench.

Ranking Method	Kendall Tau Distance	Position Consistency
w/o Judger	0.847	0.743
w/ Judger	**0.862**	**0.794**

Table 5: Results on Judger's effectiveness in aligning data with human preferences

Specifically, we use precomputed outputs of GPT-4 Turbo on AlpacaEval as reference outputs and employ GPT-4o as evaluators. As shown in the Tab. 3, our method leads to a significant improvement in the model's general instruction-following ability, outperforming both models of comparable parameter scales and even larger models.

4.4 Ablation Studies

In this section, we conduct ablation experiments to assess the impact of Judger, as described in §3.1.2, and the curriculum-based training paradigm, outlined in §3.2, on the model's ability to follow instructions. The Llama-3-8B-Instruct model is used as the base model, and evaluations are conducted on the IFEval and FollowBench benchmarks.

As shown in Tab. 4, using the constructed data directly for SFT without Judger adjustments underperforms the full method on both benchmarks, even resulting in a slight performance decline relative to the base model. It is evident that performance decreases significantly at the L4-L5 levels of FollowBench. This observation suggests that Judger plays a critical role in ranking responses to more challenging instructions. In contrast, the model trained with DPO outperforms the SFT baseline, especially on IFEval, further emphasizing the effectiveness of the DPO training approach over SFT in constraint following tasks. However, it still falls short of the performance of the full method.

Additionally, the results indicate that randomly organizing DPO training data leads to a decrease in performance. In contrast, our curriculum-based approach where training data is organized based on the number of constraints in the instructions learning leads to a significant improvement in the model's ability to follow instructions, particularly those at higher difficulty levels in L4-L5 levels of FollowBench. These findings strongly validate the necessity of Judger for constructing high-quality DPO training data and the proposed curriculum learning paradigm for enhancing the model's ability to follow complex instructions.

4.5 Analysis

4.5.1 Category Analysis

In this section, we analyze the model's performance across different types of constraints. Specifically, we compare the performance of Llama-3-8B-Instruct$_{\text{BASE}}$ and Llama-3-8B-Instruct$_{\text{DPO+Judger+CL}}$ on FollowBench. FollowBench encompasses five different constraint categories: Content, Situation, Style, Format, and Example. Each category consists of instructions from various tasks, incorporating both soft and hard constraints. Additionally, FollowBench defines Mixed Constraints as a composition of multiple constraint categories, simulating complex real-world scenarios. As shown in Fig. 4, the model's performance improves in Style with soft constraints, and Example with hard constraints. For categories that contain both soft and hard constraints, the model's performance slightly decreases. However, the trained model demonstrates a significant improvement over the base model on Mixed Constraints, suggesting a notable enhancement in the model's ability to handle complex constraints in real-world scenarios.

4.5.2 The Role of Judger

In this section, we investigate the factors contributing to the effectiveness of the Judger in constructing high-quality outputs. Judger ranks the outputs

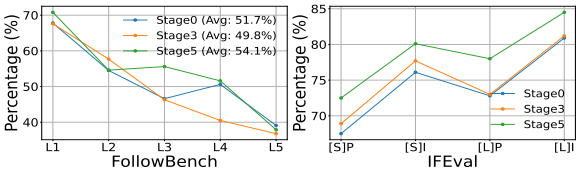

Figure 5: Results of the model on FollowBench and IFEval across different training stages in curriculum learning

to better alignment with human preferences. To examine the underlying effectiveness of the Judger, we conduct an experiment designed to evaluate whether it facilitates this alignment.

Specifically, we randomly select 100 output sets from the construction process in §3.1.1, each containing 3 to 5 outputs. These outputs are manually annotated with the correct rankings, which serve as the reference standard for comparison. We evaluate the rankings in three distinct scenarios: (1) sequential rankings, (2) rankings adjusted by Judger, and (3) rankings annotated by human experts.

To assess the similarity between these rankings, we employ two complementary metrics. The first is the Kendall Tau distance, a statistical measure that quantifies the number of discordant pairs between two sequences, thereby reflecting the extent of their relative order differences. In addition, we introduce the position consistency metric, which quantifies the proportion of elements that occupy the same relative positions across both rankings. This metric provides a direct evaluation of the alignment between rankings at each specific position. The results, presented in Tab. 5, demonstrate that the rankings adjusted by the Judger exhibit greater alignment with human-annotated rankings when compared to sequential rankings. This finding suggests that Judger enhances the quality of the training data by improving its consistency with human judgments, thus making the preference data more reliable for training.

4.5.3 The Role of Curriculum Learning

In this section, we analyze the effects of the curriculum-based training paradigm at different stages of the training process. Specifically, we examine the performance of Llama-3-8B-Instruct with the full method across three training stages, each corresponding to a different level of curriculum learning difficulty. Stage0 represents inference conducted using the base model, while Stage3 and Stage5 represent the stages where the model completes the curriculum with 3 constraints and 5 constraints, respectively.

As shown in Fig. 5, our proposed training paradigm progressively enhances the model's instruction following capability across various training stages. Specifically, after three stages of curriculum learning, the model trained in Stage3 demonstrates superior performance compared to the base model across tasks L1-L3. In contrast, the model's performance at L4-L5 in Stage3 is lower than Stage0. The possible reason is that Stage3 may not have adequately prepared for the complexity of L4-L5. The gap between these difficulty levels could have led to the initial performance drop. Subsequentially, when the model progresses to Stage5, after learning all courses, performance improves significantly at these levels. The results on IFEval further support this conclusion, showing that Stage5 achieves the highest average performance across all stages, with a notable peak at [L]I. In contrast, Stage0 demonstrates the lowest average performance across all indicators. By initially focusing on simpler preference learning and gradually progressing to more complex one, the model's ability to adhere to instructions improves incrementally. This progression enables the model to achieve better performance on increasingly difficult instruction following tasks.

5 Conclusion

In this paper, we systematically study how to improve LLMs' overall ability to follow instructions with soft constraints. Initially, we design a pipeline to automate the construction of datasets with high-quality outputs for soft constraint following. Based on the pipeline, we introduce a method utilizing positive and negative samples generated during the pipeline. Moreover, we propose a new training paradigm that leverages curriculum learning to enhance LLMs' constraint following ability. Our experiments show that our methods enhance models' ability to follow soft constraints effectively while maintaining general capabilities.

6 Limitations

We discuss the limitations of our study as follows. First, we improve the model's ability to following complex constraints, thereby improving its overall instruction following capability. However, even when the model's output meets all the specified constraints, it may still struggle to fully comply

with complex instructions due to limitations in reasoning capacity or the knowledge it masters. Additionally, we need to define constraints with greater precision to more accurately capture the complexity of real-world scenarios.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Dhananjay Ashok and Barnabas Poczos. 2024. Controllable text generation in the instruction-tuning era. *arXiv preprint arXiv:2405.01490*.

Jinheon Baek, Nirupama Chandrasekaran, Silviu Cucerzan, Allen Herring, and Sujay Kumar Jauhar. 2024. Knowledge-augmented large language models for personalized contextual query suggestion. In *Proceedings of the ACM on Web Conference 2024*, pages 3355–3366.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Yihan Chen, Benfeng Xu, Quan Wang, Yi Liu, and Zhendong Mao. 2024a. Benchmarking large language models on controllable generation under diversified instructions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17808–17816.

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

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. 2024. Self-play with execution feedback: Improving instruction-following capabilities of large language models. *arXiv preprint arXiv:2406.13542*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Qianyu He, Jie Zeng, Qianxi He, Jiaqing Liang, and Yanghua Xiao. 2024a. From complex to simple: Enhancing multi-constraint complex instruction following ability of large language models. *arXiv preprint arXiv:2404.15846*.

Qianyu He, Jie Zeng, Wenhao Huang, Lina Chen, Jin Xiao, Qianxi He, Xunzhe Zhou, Jiaqing Liang, and Yanghua Xiao. 2024b. Can large language models understand real-world complex instructions? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18188–18196.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023a. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. 2023b. Followbench: A multi-level fine-grained constraints following benchmark for large language models. *arXiv preprint arXiv:2310.20410*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. 2024. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36.

JoonHo Lee, Jae Oh Woo, Juree Seok, Parisa Hassanzadeh, Wooseok Jang, JuYoun Son, Sima Didari, Baruch Gutow, Heng Hao, Hankyu Moon, et al. 2024. Improving instruction following in language models through proxy-based uncertainty estimation. *arXiv preprint arXiv:2405.06424*.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason Weston, and Mike Lewis. 2023a. Self-alignment with instruction backtranslation. *arXiv preprint arXiv:2308.06259*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models.

Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, Feiyu Xiong, et al. 2024. Controllable text generation for large language models: A survey. *arXiv preprint arXiv:2408.12599*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Yinpeng Liu, Jiawei Liu, Xiang Shi, Qikai Cheng, Yong Huang, and Wei Lu. 2024. Let's learn step by step: Enhancing in-context learning ability with curriculum learning. *arXiv preprint arXiv:2402.10738*.

Renze Lou, Kai Zhang, and Wenpeng Yin. 2024. Large language model instruction following: A survey of progresses and challenges. *Computational Linguistics*, pages 1–10.

Sourabrata Mukherjee, Atul Kr Ojha, and Ondřej Dušek. 2024. Are large language models actually good at text style transfer? *arXiv preprint arXiv:2406.05885*.

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*, 35:27730–27744.

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

Chau Minh Pham, Simeng Sun, and Mohit Iyyer. 2024. Suri: Multi-constraint instruction following for long-form text generation. *arXiv preprint arXiv:2406.19371*.

Hadi Pouransari, Chun-Liang Li, Jen-Hao Rick Chang, Pavan Kumar Anasosalu Vasu, Cem Koc, Vaishaal Shankar, and Oncel Tuzel. 2024. Dataset decomposition: Faster llm training with variable sequence length curriculum. *arXiv preprint arXiv:2405.13226*.

Xiao Pu, Tianxing He, and Xiaojun Wan. 2024. Stylecompress: An llm-based prompt compression framework considering task-specific styles. *arXiv preprint arXiv:2410.14042*.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. Infobench: Evaluating instruction following ability in large language models. *arXiv preprint arXiv:2401.03601*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Kanghyun Ryu, Qiayuan Liao, Zhongyu Li, Koushil Sreenath, and Negar Mehr. 2024. Curricullm: Automatic task curricula design for learning complex robot skills using large language models. *arXiv preprint arXiv:2409.18382*.

Murray Shanahan, Kyle McDonell, and Laria Reynolds. 2023. Role play with large language models. *Nature*, 623(7987):493–498.

Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. 2023. Character-llm: A trainable agent for role-playing. *arXiv preprint arXiv:2310.10158*.

Atli Sigurgeirsson and Simon King. 2024. Controllable speaking styles using a large language model. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10851–10855. IEEE.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2024. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18990–18998.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. 2022. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565.

Haoran Sun, Lixin Liu, Junjie Li, Fengyu Wang, Baohua Dong, Ran Lin, and Ruohui Huang. 2024. Conifer: Improving complex constrained instruction-following ability of large language models. *arXiv preprint arXiv:2404.02823*.

Zhen Tao, Dinghao Xi, Zhiyu Li, Liumin Tang, and Wei Xu. 2024. Cat-llm: Prompting large language models with text style definition for chinese article-style transfer. *arXiv preprint arXiv:2401.05707*.

Fei Wang, Chao Shang, Sarthak Jain, Shuai Wang, Qiang Ning, Bonan Min, Vittorio Castelli, Yassine Benajiba, and Dan Roth. 2024. From instructions to constraints: Language model alignment with automatic constraint verification. *arXiv preprint arXiv:2403.06326*.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023a. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Xin Wang, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022a. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022b. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *arXiv preprint arXiv:2204.07705*.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023b. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Zekun Wang, Ge Zhang, Kexin Yang, Ning Shi, Wangchunshu Zhou, Shaochun Hao, Guangzheng Xiong, Yizhi Li, Mong Yuan Sim, Xiuying Chen, et al. 2023c. Interactive natural language processing. *arXiv preprint arXiv:2305.13246*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. 2024. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2023. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Computing Surveys*, 56(3):1–37.

Tao Zhang, Yanjun Shen, Wenjing Luo, Yan Zhang, Hao Liang, Fan Yang, Mingan Lin, Yujing Qiao, Weipeng Chen, Bin Cui, et al. 2024. Cfbench: A comprehensive constraints-following benchmark for llms. *arXiv preprint arXiv:2408.01122*.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. 2024a. Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning. *arXiv preprint arXiv:2402.04833*.

Zirui Zhao, Hanze Dong, Amrita Saha, Caiming Xiong, and Doyen Sahoo. 2024b. Automatic curriculum expert iteration for reliable llm reasoning. *arXiv preprint arXiv:2410.07627*.

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

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

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

Wangchunshu Zhou, Yuchen Eleanor Jiang, Ethan Wilcox, Ryan Cotterell, and Mrinmaya Sachan. 2023b. Controlled text generation with natural language instructions. In *International Conference on Machine Learning*, pages 42602–42613. PMLR.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2023. Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, 36:50117–50143.

A Details of Data

A.1 Details of Soft Constraints

The three categories of soft constraints that we define are as follows:

- **Soft Constraints in Content**: Content soft constraints refer to limitations associated with the data itself. These constraints govern the elements of information, the logical relationships between them, and the scope of topics that need to be covered in the response. When multiple content soft constraints are imposed, the model is required to not only generate comprehensive and coherent content but also ensure that the response aligns with the specific logical definitions and boundaries outlined by the instruction. This presents a significant challenge, as it demands both the integration of diverse elements and the maintenance of internal consistency. To address this challenge, we define the following tasks for constructing and applying content soft constraints:

    1. **Inclusion of Key Elements**: The response must incorporate the key points specified in the instruction. This requires the model to effectively extract and integrate relevant information, ensuring that the essential components are included without omitting critical details.
    2. **Topic Focus**: The model must narrow the discussion to a specific subtopic, avoiding broad generalizations or irrelevant tangents. This task emphasizes the importance of maintaining focus and precision within the scope defined by the instruction.
    3. **Strict Structure**: The generated content must adhere to a predefined structure, such as being organized into coherent

paragraphs, utilizing subheadings, or following a specific format. This task imposes a higher demand on the model's ability to generate well-organized and structured outputs, aligning with the required presentation structure.

We provide the prompt template for constructing the Content Soft Constraint in Tab. 7.

- **Soft Constraints in Situation**: Situation soft constraints are those related to the context within which the response is situated. These constraints require the response to be adjusted according to the context or assumptions specified in the instruction, ensuring that the content is appropriate to the given background. Such adjustments may involve factors like a particular time or location, the assumption of a specific role, or drawing conclusions based on certain premises. The response must dynamically adapt to situational changes and maintain consistency with the contextual elements. The tasks defined by these constraints can be categorized as follows:

  1. **Role-Playing**: The response must be framed from the perspective of a specific role or persona, ensuring alignment with the contextual expectations associated with that role.
  2. **Decision Support**: The response should provide advice or recommendations that support decision-making within a particular context.
  3. **Storytelling**: The response should construct a narrative that is situated within a defined time, location, or background, maintaining coherence with the provided contextual elements.

  We provide the prompt template for constructing the Situation Soft Constraint in Tab. 8.

- **Soft Constraints in Style**: Style soft constraints pertain to the mode of expression, encompassing factors such as the formality or informality of tone, the level of conciseness in language, and the emotional tenor. These constraints require the response to adjust its style in accordance with the given requirements, adapting to different linguistic contexts. The following task types are defined under this category:

  1. **Tone Requirement**: The generated content must adopt a specific tone, such as formal, humorous, or otherwise defined.
  2. **Language Complexity Control**: The complexity of the language used must adhere to specific standards, such as maintaining conciseness and clarity or employing academic expressions.
  3. **Emotional Expression**: The response must convey a particular emotion, such as positivity or sadness, as dictated by the context.

  We provide the prompt template for constructing the Style Soft Constraint in Tab. 9.

A.2 Details of Judger Ranking

We provide the prompt of Judger ranking in Tab. 10 and examples of how the Judger ranks responses in Tab. 6.

B Details of Experiments

B.1 Training hyperparameters

We train Mistral-7B-Instruct-v0.3 and Llama-3-8B-Instruct using LLaMA-Factory (Zheng et al., 2024) on 4 NVIDIA A100 80GB GPUs, applying LoRA (Hu et al., 2021) for efficient training. The lora target is set to all, and both models use the following training parameters, with training running for 3 epochs. The per device train batch size is set to 1, and gradient accumulation steps is set to 8. The warm-up ratio is set to 0.1. For SFT, Mistral-7B-Instruct-v0.3 is trained with a learning rate 5.0e-7 , while the learning rate of Llama-3-8B-Instruct is 1.0e-4. For DPO, the learning rate is set to 5.0e-6, with a beta value of 0.1.

B.2 Full Results on FollowBench

We present the full results on FollowBench includng the Hard Satisfaction Rate (HSR) metric and Soft Satisfaction Rate (SSR) metric in Tab. 11. As shown in Fig. 6 and Fig. 7, we also illustrate the average HSR and SSR scores of Llama-3-8B-Instruct on different constraint categories in FollowBench.

Judger Ranking Case	
Instruction	Please write a brief description of quantum physics, ensuring that the terms "wave function" and "superposition state" are included. Avoid using overly complex formulas. Keep the response under 100 words.Ensure that this description is easy to understand.
Model Reponses	Output (a) : Quantum physics studies the behavior of tiny particles. The *wave function* describes a particle's state, while *superposition state* means a particle can exist in multiple states at once. Output (b) : Quantum physics studies the behavior of tiny particles. In the quantum world, particles don't have fixed positions or states; they can be in many states at once, and only take a definite state when measured.
Judger Output	[[A]]. From the perspective of constraint adherence, Output (a) is better. It clearly includes both required terms, "wave function" and "superposition state," and explains them in a simple and concise way. The response stays under 100 words and avoids complex formulas, making it easy to understand.Output (b), on the other hand, doesn't mention "wave function" and its explanation of "many states" is less clear, making it less aligned with the instruction.

Table 6: The case for Judger ranking outputs.

Prompt Template for Content Soft Constraint

You are an Instruction Rewriting Expert. You need to rewrite #Given Instruction# based on #Rewriting Requirement#, in order to obtain a #Rewritten Instruction#. Basically, #Rewritten Instruction# should adhere to the following guidelines:
1. Your rewriting cannot omit the non-text parts such as the table and code in #Given Instruction#.
2. #Rewritten Instruction# must be reasonable and must be understood and responded to by humans.
3. You should try your best not to make the #Rewritten Instruction# become verbose, #Rewritten Instruction# can only add 10 to 20 words into #Given Instruction#.
/ The Given Instruction */*
{Given Instruction}
/ Rewriting Requirement */*
Please add one proper content constraint to the #Given Instruction#. The content constraints include but are not limited to:
1. Information Inclusion: The response must include the key points mentioned in the instruction.
2. Topic Focus: The discussion should be focused on a specific subtopic, avoiding generalizations.
3. Strict Structure: The generated content must follow a specific structure.
4. Clarity of Purpose: The response should clearly align with the goal or purpose outlined in the instruction.
5. Detail Depth: The response should offer sufficient detail without oversimplifying.

Table 7: The prompt template for constructing the Content Soft Constraint.

Prompt Template for Situation Soft Constraint

You are an Instruction Rewriting Expert. You need to rewrite #Given Instruction# based on #Rewriting Requirement#, in order to obtain a #Rewritten Instruction#. Basically, #Rewritten Instruction# should adhere to the following guidelines: 1. Your rewriting cannot omit the non-text parts such as the table and code in #Given Instruction#. 2. #Rewritten Instruction# must be reasonable and must be understood and responded to by humans. 3. You should try your best not to make the #Rewritten Instruction# become verbose, #Rewritten Instruction# can only add 10 to 20 words into #Given Instruction#. */* The Given Instruction */*
{Given Instruction}
/ Rewriting Requirement */*
Please add one proper situation constraint to the #Given Instruction#. The situation constraints include but are not limited to: 1.Specify a Role: Clearly define the role or persona the response should adopt. 2.Decision Support: Offer advice that aids decision-making within a particular context or situation. 3.Introduce a Conflict or Challenge: Present a specific problem, conflict, or challenge that needs to be resolved. 4.Introduce Time Constraints: Set a time limit for completing specific actions or tasks. 5.Contextual Storytelling: Require the response to include a story or narrative based on a defined time, location, or background.

Table 8: The prompt template for constructing the Situation Soft Constraint.

Prompt Template for Style Soft Constraint

You are an Instruction Rewriting Expert. You need to rewrite #Given Instruction# based on #Rewriting Requirement#, in order to obtain a #Rewritten Instruction#. Basically, #Rewritten Instruction# should adhere to the following guidelines: 1. Your rewriting cannot omit the non-text parts such as the table and code in #Given Instruction#. 2. #Rewritten Instruction# must be reasonable and must be understood and responded to by humans. 3. You should try your best not to make the #Rewritten Instruction# become verbose, #Rewritten Instruction# can only add 10 to 20 words into #Given Instruction#. */* The Given Instruction */*
{Given Instruction}
/ Rewriting Requirement */*
Please add one proper style constraint to the #Given Instruction#. The style constraints include but are not limited to: 1. Tone Requirement: The response must adopt a specific tone. 2. Language Complexity Control: The complexity of the language used must meet specific standards. 3. Emotional Tone Expression: The response must convey a specific emotion. 4. Precision of Expression: The response should be concise and direct, avoiding ambiguity or verbosity. 5. Rhetorical Devices: The response may employ rhetorical devices to enhance its expressive power.

Table 9: The prompt template for constructing the Style Soft Constraint.

Prompt Template for Judger

You are a helpful assistant who reviews a debate between two other assistants in evaluating the quality of the outputs for a given instruction.
The two assistants, Assistant (a) and Assistant (b), are given an instruction, Output (a) and Output (b). They are asked to select the Output (a) or Output (b) that is better for the given instruction. Output (a) and Output (b) are generated by two different AI chatbots respectively.
Assistant (a) and Assistant (b) have conflicting evaluations. Your goal is to review their evaluations and give your final decision on which output is better.
Here are some rules of the evaluation:
(1) You should prioritize evaluating whether the output honestly/precisely/closely executes the instruction, then consider its helpfulness, accuracy, level of detail, harmlessness, etc.
(2) Outputs should NOT contain more/less than what the instruction asks for, as such outputs 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 outputs were presented should NOT affect your judgment, as Output (a) and Output (b) are **equally likely** to be the better.
Output your final verdict by strictly following this format: "[[A]]" if Output (a) is better, "[[B]]" if Output (b) is better, and "[[C]]" for a tie.
{Given instruction}
{question}
/ The Start of Output (a) */*
{answer of assistant a}
/ The Start of Output (b) */*
{answer of assistant b}

Table 10: The prompt template for Judger to rank the responses.

Model	BaseModel	FollowBench (HSR)						FollowBench (SSR)					
		L1	L2	L3	L4	L5	Avg	L1	L2	L3	L4	L5	Avg
GPT4 (Achiam et al., 2023)*	GPT	84.7	75.6	70.8	73.9	61.9	73.4	84.7	77.0	75.3	77.0	72.3	77.2
GPT3.5-turbo*	GPT	80.3	68.0	68.6	61.1	53.2	66.2	80.3	71.2	74.2	69.6	67.1	72.5
Llama-3.1-70B-Instruct (Dubey et al., 2024)	LLaMA3	75.2	69.6	63.1	65.9	57.1	66.2	74.7	71.2	69.3	64.3	57.2	67.3
Qwen2-72B-Instruct (Yang et al., 2024)	Qwen	67.9	56.6	47.8	42.2	35.3	50.0	78.0	71.7	69.3	65.1	65.6	69.9
WizardLM-v1.2-13B (Xu et al., 2023)*	LLaMA2	68.8	**64.1**	53.1	40.8	35.8	52.5	68.8	65.7	61.8	53.4	53.9	60.7
Conifer-13B (Sun et al., 2024)	LLaMA2	60.5	53.6	48.4	40.7	31.7	47.0	60.5	58.3	58.2	53.9	51.1	56.4
Vicuna-13B-v1.5 (Chiang et al., 2023)*	LLaMA2	**71.2**	60.2	49.6	40.6	34.0	51.1	71.2	64.8	59.9	54.5	53.6	60.8
Conifer-7B-SFT (Sun et al., 2024)	Mistral	54.3	49.5	49.3	40.8	30.5	44.9	53.9	57.6	53.7	54.5	49.7	53.9
Conifer-7B-DPO (Sun et al., 2024)	Mistral	60.3	53.6	48.0	47.1	**41.0**	50.0	60.3	55.7	55.7	55.9	53.3	56.2
Mistral-7B-Instruct-v0.3$_{\text{BASE}}$	Mistral	58.7	50.9	48.5	37.5	27.6	44.6	78.0	71.7	69.3	65.1	65.6	69.9
Mistral-7B-Instruct-v0.3$_{\text{SFT}}$	Mistral	58.7	52.4	42.5	37.2	35.6	45.3	**82.8**	70.5	72.2	66.9	**71.0**	**72.7**
Mistral-7B-Instruct-v0.3$_{\text{DPO+Judger+CL}}$	Mistral	61.2	52.5	47.5	38.2	33.9	46.7	78.0	71.7	69.3	65.1	65.6	69.9
Llama-3-8B-Instruct$_{\text{BASE}}$	LLaMA3	67.8	54.5	46.6	50.6	39.1	51.7	80.9	72.1	67.3	70.1	67.1	71.5
Llama-3-8B-Instruct$_{\text{SFT}}$	LLaMA3	69.3	59.0	50.1	44.8	32.0	51.0	82.6	**75.0**	69.7	**71.1**	62.0	72.1
Llama-3-8B-Instruct$_{\text{DPO+Judger+CL}}$	LLaMA3	70.8	54.6	**55.6**	**51.6**	37.9	**54.1**	81.8	66.4	**75.2**	69.1	68.8	72.3

Table 11: Full results on FollowBench. We use boldface for the best results and underline for the second-best results among the models ranging from 7B to 13B parameter sizes. * indicates that the results are directly sourced from the original benchmarks.

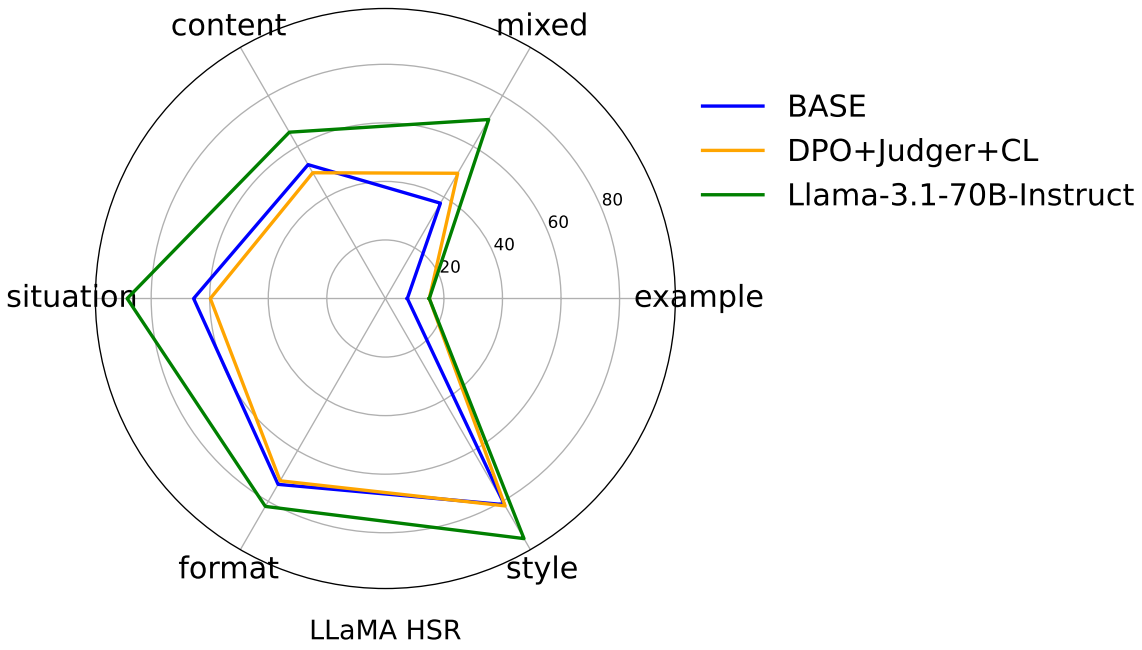

Figure 6: Average HSR results for LLaMA models across various constraint categories in FollowBench.

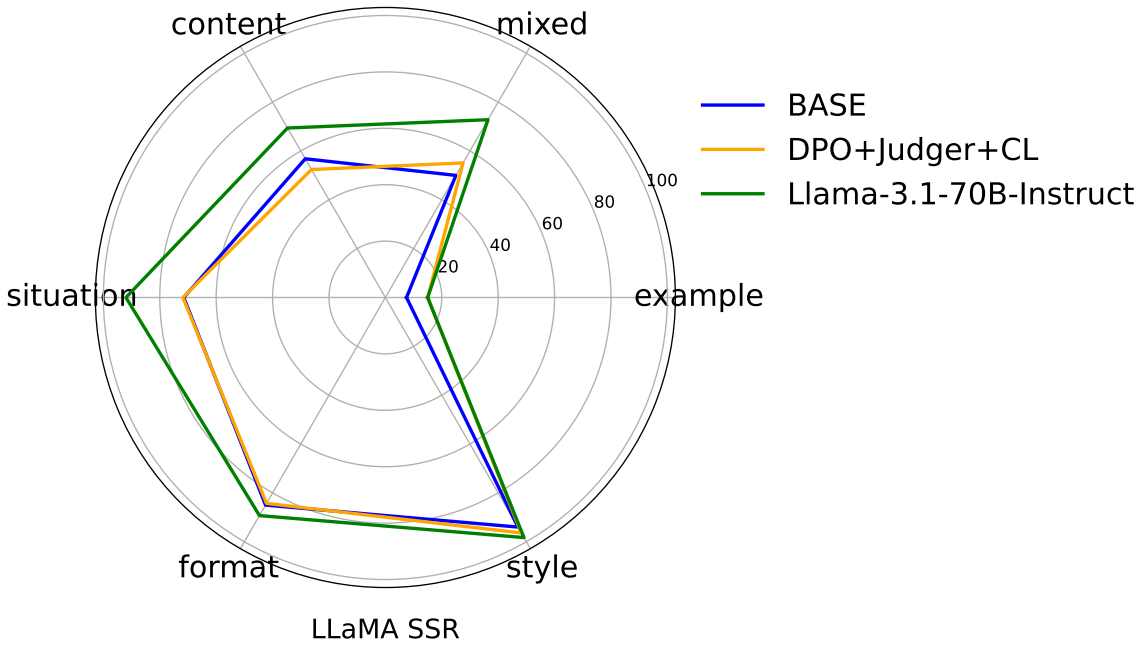

Figure 7: Average SSR results for LLaMA models across various constraint categories in FollowBench.