RULER: Improving LLM Controllability by Rule-based Data Recycling

Ming Li[*1], Han Chen[*], Chenguang Wang[*2], Dang Nguyen[1], Dianqi Li, Tianyi Zhou[1]
[1]University of Maryland [2]Stony Brook University
{minglii, tianyi}@umd.edu
Project: https://github.com/tianyi-lab/RuleR

Abstract

Despite the remarkable advancement of Large language models (LLMs), they still lack delicate controllability under sophisticated constraints, which is critical to enhancing their response quality and the user experience. While supervised fine-tuning (SFT) can potentially improve LLM controllability, curating new SFT data to fulfill the constraints usually relies on human experts or proprietary LLMs, which is time-consuming and expensive. To bridge this gap, we propose **Rule-based Data Recycling (RULER)**, a human/LLM-free data augmentation method incorporating multiple constraints into the original SFT data. Instead of creating new responses from scratch, RULER integrates linguistic or formatting rules into the original instructions and modifies the responses to fulfill the rule-defined constraints. Training on the "recycled" data consolidates LLM capability to generate constrained outputs, improving LLM controllability while maintaining promising general instruction-following capabilities.

1 Introduction

Despite the remarkable advancement of the current Large language models (LLMs) and the continuous efforts to build high-quality supervised fine-tuning (SFT) datasets, one critical challenge is to generate responses better interacting with humans, with the utility and effectiveness maximized for end-users (Liu et al., 2024; Huang et al., 2024, 2023). According to the systematic investigation from Liu et al. (2024), it is essential for LLMs to constrain their outputs to follow user-specified formats or characteristics. In various practical applications, free-formed responses are not legal or directly applicable without any constraint or format being enforced. It has also been verified on LLM Agents (Li et al., 2024f; Chen et al., 2023, 2024b; Zhang et al., 2024) that enforcing predefined formats is necessary for tasks.

[*]Equal Contribution.

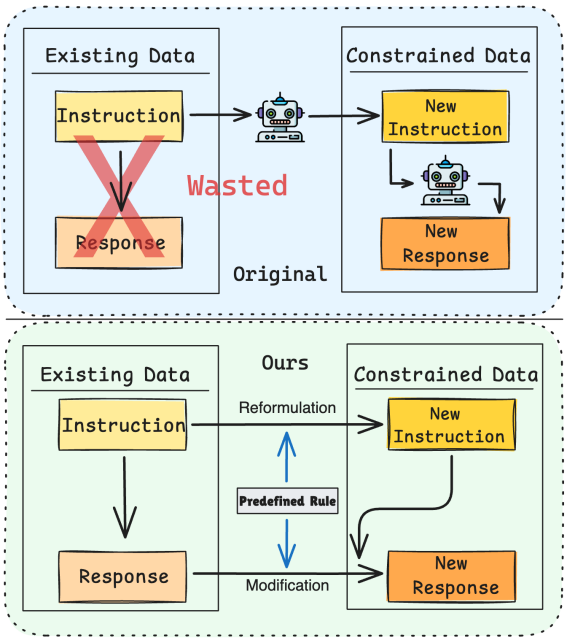

Figure 1: Comparing widely-used data generation strategy (top) and RULER (bottom) enhancing LLM controllability. Most existing methods rely on human/model rewriting to generate new instructions and responses. However, discarding existing data is a waste of effort. Our RULER demonstrates that simple rule-based (human/model-free) editing of existing data can generate new SFT data that improves LLM controllability.

However, existing SFT datasets are mainly composed of general instructions without user-specified constraints (Wei et al., 2022; Wang et al., 2022; Taori et al., 2023; Xu et al., 2023; Zhou et al., 2023a; Li et al., 2023a; Zhang et al., 2023; Xu et al., 2024) and thus result in models lacking delicate controllability of the lengths and format of responses (Chen et al., 2024a; Xia et al., 2024). To enhance the utility of existing SFT data in improving the controllability of LLMs, a potential method is to rewrite or modify instructions and responses by experts such as humans/LLMs (Xu et al., 2023; Li et al., 2023a, 2024b,a; He et al., 2024; Dong et al., 2024; Wu et al., 2024) in order to make them

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers), pages 926–943
April 29 - May 4, 2025

fulfilling multiple constraints, as shown in Figure 1 (top). However, the curation of new data is not only costly and inefficient, requiring careful editing by human experts or proprietary LLMs, but also represents a waste of previous efforts: It is impractical to discard all existing data and create brand new data every time we need to add more constraints to the instructions. Hence, we raise the question: ***Can we "recycle" existing SFT data without human/LLM editing and enforce various types of constraints in order to improve LLM controllability?***

Drawing inspiration from IFEval (Zhou et al., 2023b), which utilizes verifiable constraints to evaluate LLMs' controllability, and the human/model-free data augmentation in Mosaic-IT (Li et al., 2024c), we propose **Rule-based Data Recycling (RULER)**, which automatically "recycles" existing SFT data for improving LLM controllability. As illustrated in Figure 1 (bottom), the key insight of RULER is to automatically build constraint-augmented SFT datasets **at no cost of human/LLM efforts**, by applying predefined rules to the original instructions and responses. Specifically, we manually inspect and construct a diverse set of rules as constraints, which specify the linguistic or formatting constraints on different parts of the response.

Our predefined rules cover a wide range of diverse constraints generalizable to many application scenarios, ranging from high-level constraints, e.g., controlling the word frequency in the response, to lower-level constraints, e.g., setting specific wrapping formats of some keywords. Each rule is composed of (1) multiple templates to produce additional instructions enforcing the constraints, and (2) a piece of code that alternately edits the original instruction and response in order to make the edited response fulfill all the constraints appended to the instruction. For each sample from the original dataset, we randomly draw several rules to be applied to the editing. This produces an augmented sample with the constraints enforced so it can be used for controllability tuning. The complete list of rules and descriptions can be found in Appendix E.

Illustrative examples are provided in Figure 2, which showcase (a) a rule that constrains the number of letters; (b) a rule that specifies the case of specific words (if presenting in the response); and (c) a rule that specifies the wrapping format of specific words. To ensure the consistency between the input constraints and the output response, we modify both the instruction and response based on

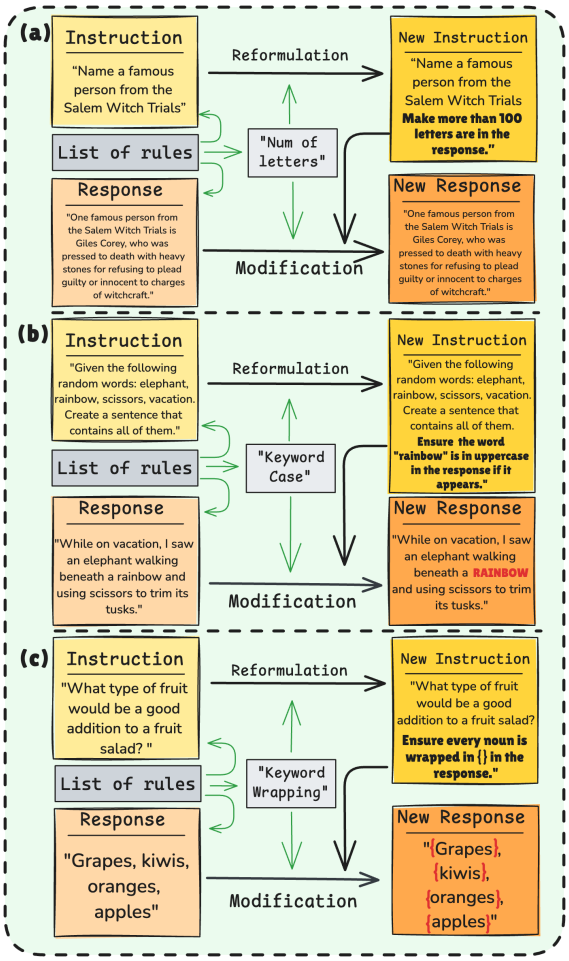

Figure 2: Examples of our data recycling workflows. (a), (b) and (c) select different predefined rules to modify the original data to fulfill constraints on the complexity or format of the response. The differences in new responses are highlighted in red, the example in (a) has already satisfied the appended constraint, thus the response is kept unchanged.

the characteristics of the original data sample. For each sample, we only sample from the rules applicable to the original response, hence avoiding the potential incorrectness of the edited responses.

Extensive experimental results on the IFEval benchmark on various base models and datasets demonstrate the effectiveness of RULER in enhancing LLM controllability without extra help from humans/models. On the other hand, RULER still preserves the general instruction-following ability promoted by the original SFT dataset. This is demonstrated by the instruction-following metrics (Pair-wise comparison and Open LLM Leaderboard). To the best of our knowledge, RULER is the **first human/model-free data augmentation and recycling approach designed to improve LLM controllability** under multiple constraints.

2 Methodology

2.1 Preliminaries

Given a supervised finetuning dataset D, there are N data samples, each represented by a tuple (x_i, y_i), where x_i represents the instruction and y_i represents the corresponding response. Let $p_\theta(\cdot)$ denote the LLM with parameters θ to be trained. In the instruction tuning setting, p_θ is typically fine-tuned by maximizing the following objective on all the N samples as (x_i, y_i), in which $y_{i,j}$ represents the j_{th} token of response y_i, $y_{i,<j}$ represents the tokens before $y_{i,j}$, and l_i represents the token length of y_i:

$$\max_{\theta} \sum_{i=1}^{N} \sum_{j=1}^{l_i} \log p_\theta \left(y_{i,j} | x_i, y_{i,<j}\right), \qquad (1)$$

2.2 Rule-based Data Recycling (RULER)

2.2.1 Rule Construction

While most existing methods still require human experts or strong teacher LLMs to generate new data (Figure 1 (top)), we aim at "recycling" instructions and responses from existing SFT datasets to build controllability-focused datasets, without expensive and time-consuming supervision from humans or LLMs. In the following, we introduce a rule-based approach "**Rule-based Data Recycling (RULER)**" to create high-quality augmented data for improving LLM controllability.

RULER reformulates the original instructions and responses by applying rule-based edits according to pre-defined constraints. However, not every constraint is applicable to a randomly selected sample without fully rewriting. Hence, we only incorporate constraints compatible with the original sample and those can be implemented with simple rectifications like regular expressions. Specifically, we focus on the characteristics of responses that can be defined by rules, e.g., by checking the 220 distinct linguistic features in the LFTK package (Lee and Lee, 2023). In addition, we collect constraints from existing works and widely used instruction-tuning datasets. These diverse characteristics include punctuation-, word-, sentence-, paragraph-level occurrences, and frequencies. Thus we construct rules constraining or specifying the characteristics of the original response. In conjunction with the rules containing these characteristics, we also create rules specifying the format of responses to improve LLM's format-following capability.

To ensure that the rules selected or sampled for each sample are applicable, we apply the following additional protocols: (1) The rules need to be applicable to the original sample. For instance, the rule "*Generating a title before giving the response*" is not applicable as we can not generate a title without the help of humans or other additional models. (2) The rules should not include removing the content of the original response. For instance, the rule "*Ensure the word xxx is not shown in the response*" is not applicable since we can not directly remove this word from the response as the removal might disrupt the original semantic integrity. (3) The rules should be compatible with the original sample. If the rule is "*Ensure there are more than N sentences in the response*", then it can not be applied to samples whose responses have $< N$ sentences. The complete list of rules is provided in Table 4, which covers both high-level constraints such as the term frequency, and lower-level constraints such as specific wrapping formats.

2.2.2 Rule Implementation

To implement each rule to original instructions and responses, we notate each pre-defined rule as a tuple for simplicity, $(\mathbf{S}_k, f_k, g_k)$, where $\mathbf{S}_k$ represents the set of manually curated instruction templates for creating instructions of the k^{th} rule, while f_k and g_k are the corresponding functions to reformulate instructions and responses (if necessary), respectively.

Specifically, the function f_k selects an appropriate template of a rule for a given data sample and augments the original instruction with an instruction generated by the template. It first selects a subset of rules applicable to the characteristics of the response (e.g., presence of keywords, number of sentences, etc.). Then, it randomly draw one rule out of the subset and create a formatted instruction of the rule from the template. Such rule instruction is appended to the original instruction.

Specifically, for each data sample (x_i, y_i), the augmented instruction $x_{i,aug}$ is reformulated according to the characteristics of the original sample and the corresponding template sets, i.e.,

$$x_{i,aug} = f_k(x_i, y_i, \mathbf{S}_k), \qquad (2)$$

The function g_k is designed to modify the response to be consistent with the augmented instruction (after applying function f_k), i.e., with the rule applied. Applying g_k either preserves the original response or revises it. Some rules do not require

editing of responses, e.g., keyword appearance, the number of nouns, etc. For rules defining the case or format of certain parts in the response, modifications are needed. The detailed descriptions of each predefined rule can be found in Appendix E. Specifically, the augmented response $y_{i,aug}$ will be optionally modified based on the selected rule k:

$$y_{i,aug} = g_k(x_i, y_i, \mathbf{S}_k). \tag{3}$$

With the augmented sample $(x_{i,aug}, y_{i,aug})$, the training objective becomes:

$$\max_{\theta} \sum_{i=1}^{N} \sum_{j=1}^{l_{i,aug}} \log p_\theta \left(y_{i,aug,j} | x_{i,aug}, y_{i,aug,<j}\right), \tag{4}$$

where $y_{i,aug,j}$ represents the jth token of response $y_{i,aug}$ and $l_{i,aug}$ represents its token length.

2.2.3 Multi-rule Implementation

To create more complex, diverse, and challenging samples, we can extend the previous process to multiple rules randomly drawn from the feasible set of rules. We provide examples of multi-rule augmentation in Appendix B, which forces LLMs to learn to follow multiple constraints.

The detailed experimental setup can be found in Appendix A, including Implementation Details, Training Datasets, and Evaluation Metrics.

3 Experimental Results

3.1 Main Results

The main experimental results are presented in Table 1, containing the performance comparison on the Instruction Following Eval (Zhou et al., 2023b), Pair-wise Comparison Winning Score, and the Open LLM Leaderboard (Gao et al., 2021a), on 3 different base models and several different instruction tuning datasets. The Pair-wise Comparison Winning Score is calculated as *(Num(Win)−Num(Lose))/Num(All)* $+1$ and the values that are greater than 1.0 represent better responses generated. Detailed descriptions of evaluation metrics can be found in the Appendix D.

Compared to the baseline, our method has consistent improvements on the IF Eval benchmark, across different base models and datasets, which aims at measuring LLMs' constraint-following abilities by using verifiable instructions. It is astonishing that our method can improve the IF Eval scores by approximately 10% on some of the configurations, by just utilizing the rule-based recycling method on the original data, without any human/model edition. Moreover, the performances keep being positive on originally diverse and high-quality datasets like Recycled WizardLM (Li et al., 2023a) and DEITA (Liu et al., 2023), which further verify the potential of our method. Compared with existing methods which enhance LLMs' constraint controllability by generating totally new data, our method focuses more on fully utilizing the potential of existing data.

Furthermore, our method not only improves the constraint controllability but also keeps the general instruction-following ability of the original data. The Pair-wise comparison and Open LLM leaderboard results showcase comparable or sometimes better performances compared with the baseline models. We hypothesize that the additional constraints largely complicate the original instructions, as shown in Figure 3, thus forcing the LLMs to understand each constraint before generating responses, thus leading to potentially improved instruction-following abilities.

3.2 Ablation Studies

In this section, ablation experiments are conducted on Mistral-7B with the Alpaca-GPT4 dataset, aiming to evaluate the impact of several factors.

Effect of Templates: As shown in Table 2, "Single Temp." represents utilizing only one rule-instruction template for each rule, while "Diverse Temp." represents utilizing several different templates, approximately 10, with the same meaning for each rule and those templates are randomly sampled during the augmentation. "Diverse Temp." demonstrates superior performance in both IF Eval and Pair-wise winning scores, with slightly lower accuracy on the Open LLM leaderboard. This result suggests that "Diverse Temp." enhances the model's constraint controllability while enhancing its general instruction-following capabilities compared to "Single Temp." On the contrary, when fixing the rule templates to one single template, potential overfitting to the template might occur and thus negatively influence the model performances.

Effect of Rule Numbers: "Max Rule = x" represents the setting in which at most x different rules can be sampled and utilized on each original data sample. In the augmentation process, a random value will be sampled in the range of $[0, x]$ as the number of rules in this sample. The examples in Figure 2 showcase the scenario when only one rule

Model	Dataset	Method	Instruction Following Eval				Pair-wise	Open LLM Leaderboard				
			Prompt (S)	Inst (S)	Prompt (L)	Inst (L)	Winning Score	Average	A	H	M	T
Mistral-7B	Alpaca-GPT4	Baseline	32.72	42.45	35.67	45.44	1.000	**61.24**	56.23	81.07	56.22	51.42
		Ours	**39.56**	**51.44**	**43.44**	**55.40**	**1.044**	61.05	56.66	80.50	57.24	49.81
	Alpaca	Baseline	33.64	44.60	36.23	47.96	1.000	55.15	51.96	74.61	52.85	41.20
		Ours	**35.12**	**46.76**	**37.89**	**49.52**	**1.158**	**56.21**	54.61	77.70	54.54	38.00
	Wizard-70k	Baseline	37.34	48.68	40.85	52.04	1.000	59.38	54.61	79.96	55.68	47.27
		Ours	**46.77**	**57.07**	**49.17**	**59.71**	**1.168**	**59.75**	55.38	80.75	55.59	47.27
	Recycled Wizard	Baseline	30.87	42.41	35.86	46.40	**1.000**	59.61	54.10	77.85	57.61	48.87
		Ours	**39.56**	**51.08**	**45.10**	**56.24**	0.987	**60.43**	56.66	78.01	58.61	48.43
	DEITA 6K	Baseline	41.22	51.08	44.55	54.92	1.000	64.82	60.41	82.52	61.57	54.76
		Ours	**42.14**	**52.28**	**46.77**	**56.59**	**1.010**	**65.43**	61.86	82.71	62.66	54.49
Llama2-7B	Alpaca-GPT4	Baseline	26.25	36.33	30.31	40.29	1.000	58.71	54.69	80.05	47.89	52.21
		Ours	**32.35**	**42.09**	**35.30**	**45.56**	**1.070**	**59.77**	56.74	80.67	48.45	53.21
	Alpaca	Baseline	31.42	40.77	33.46	43.17	1.000	**55.25**	54.35	78.65	47.02	40.98
		Ours	**34.38**	**44.36**	**37.34**	**47.36**	**1.023**	55.24	54.61	78.76	46.17	41.42
	Wizard-70k	Baseline	31.24	44.24	35.49	48.68	1.000	57.09	54.18	79.25	46.93	48.02
		Ours	**38.82**	**50.12**	**42.33**	**53.48**	**1.087**	**57.25**	55.20	79.81	46.61	47.38
Llama2-13B	Alpaca-GPT4	Baseline	32.90	44.60	36.23	48.08	**1.000**	61.47	58.70	83.12	54.13	49.92
		Ours	**36.60**	**47.00**	**37.89**	**49.28**	0.977	**61.96**	59.47	82.88	53.98	51.52
	Alpaca	Baseline	34.94	44.36	36.41	46.29	**1.000**	**57.63**	57.25	81.23	54.13	37.91
		Ours	**36.04**	**48.20**	**41.22**	**52.88**	0.977	57.16	57.17	81.11	52.70	37.65
	Wizard-70k	Baseline	43.07	53.84	46.40	57.67	1.000	**61.24**	57.04	83.39	55.76	48.78
		Ours	**45.47**	**58.15**	**50.09**	**61.99**	**1.010**	60.84	58.28	82.37	54.35	48.36

Table 1: **Main Results.** Evaluation on the Instruction Following Eval, Pair-wise Comparison Winning Score, and the Open LLM Leaderboard. We compare RULER with Baseline for finetuning three base models on several different instruction tuning datasets. *Baseline* – models trained with the original dataset; *Ours* – models trained with RULER-recycled datasets; *Prompt* – Prompt-level accuracy; *Inst* – Instruction-level accuracy; *S* and *L* represent Strict and Loose versions. *A*, *H*, *M*, and *T* denote ARC, HellaSwag, MMLU, and TruthfulQA.

Evaluation Metrics	IF Eval	Pair-wise	Open LLM
Baseline	39.07	1.000	61.24
Single Temp.	42.62	0.987	**61.43**
Diverse Temp. (*)	**47.46**	**1.044**	61.05
Max Rule = 1	46.14	**1.168**	61.22
Max Rule = 2	47.09	1.117	61.15
Max Rule = 3 (*)	**47.46**	1.044	61.05
Max Rule = 4	46.89	1.003	**61.55**
Max Rule = 5	44.36	1.013	60.06
Aug Rate = 0.1	41.72	1.020	**61.34**
Aug Rate = 0.3	42.08	1.037	61.20
Aug Rate = 0.5	46.55	**1.111**	61.31
Aug Rate = 0.7	46.23	**1.111**	61.18
Aug Rate = 0.9 (*)	**47.46**	1.044	61.05

Table 2: **Ablation Study.** "(*)" represents default.

is implemented and examples in Figure 3 showcase the scenario when multiple rules are implemented. Compared to the baseline, nearly all settings show performance improvements across the three evaluation metrics. However, the IF Eval score initially increases, reaching its peak when "Max Rule = 3", before declining. The Pair-wise score, on the other hand, consistently decreases from "Max Rule = 1" to "Max Rule = 5". These results suggest that applying too many rules to a single sample may impair the LLM's capability, even though sampling and applying multiple rules can be beneficial when done in moderation, which might be because the original instruction becomes so complex.

Effect of Augmentation Rate: "Aug Rate = x" represents there is a probability of x to apply our augmentation to each sample. It is observed that as the augmentation rate increases, performance improves on the IF Eval and mostly improves on the Pair-wise evaluation. This phenomenon indicates that increasing the augmentation rate primarily enhances the LLM's constraint controllability, and it also has a positive impact on its general instruction-following capability. However, the gaps on IF Eval are much larger than the other 2 metrics, indicating this rate will mostly influence the multi-constraint controllability of LLM.

Detailed Sub-Category Analysis: The detailed sub-category analysis can be found in Appendix C.

4 Conclusion

In this work, we proposed **Rule-based Data Recycling (RULER)**, which modified the original instructions and responses from an existing dataset by rule-defined constraints. The "recycled" data aims to enhance the LLMs' capability to generate outputs fulfilling the constraints specified in the input, thereby improving the controllability of LLMs. RULER took the first step of exploring rule-based data recycling, which can serve as a plug-and-play and easy-to-use method that converts any existing SFT datasets to new datasets for better controllability.

Limitations

Our method focuses on improving LLM controllability by rule-based editing of existing data, thereby avoiding the extra cost of data generation by humans or expert models. Though it saves the cost of human/model editing, the rules inevitably limit the types of constraints that can be applied to modify the original data. In the presented RULER, all the constraints and rules are based on verifiable shallow syntactic characteristics such as the occurrence and frequency of words or sentences while lacking constraints and controllability on the semantic feature or content. This implies a potential of RULER to be further enhanced by modifying the semantic content with a smaller model, which retains a comparable efficiency of rule-based editing.

References

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. Fireact: Toward language agent fine-tuning. *Preprint*, arXiv:2310.05915.

Yihan Chen, Benfeng Xu, Quan Wang, Yi Liu, and Zhendong Mao. 2024a. Benchmarking large language models on controllable generation under diversified instructions. *arXiv preprint arXiv:2401.00690*.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. 2024b. Agent-flan: Designing data and methods of effective agent tuning for large language models. *Preprint*, arXiv:2403.12881.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023a. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Preprint*, arXiv:2205.14135.

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*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021a. A framework for few-shot language model evaluation.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. 2021b. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, page 8.

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, et al. 2024. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561*.

Yue Huang, Qihui Zhang, Lichao Sun, et al. 2023. Trustgpt: A benchmark for trustworthy and responsible large language models. *arXiv preprint arXiv:2306.11507*.

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. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.

Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. 2020. Look at the first sentence: Position bias in question answering. *arXiv preprint arXiv:2004.14602*.

Bruce W. Lee and Jason Lee. 2023. LFTK: Handcrafted features in computational linguistics. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 1–19.

Ming Li, Jiuhai Chen, Lichang Chen, and Tianyi Zhou. 2024a. Can LLMs speak for diverse people? tuning LLMs via debate to generate controllable controversial statements. In *Findings of the Association for*

Computational Linguistics ACL 2024, pages 16160–16176, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. 2024b. Selective reflection-tuning: Student-selected data recycling for LLM instruction-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 16189–16211, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, and Tianyi Zhou. 2023a. Reflection-tuning: Recycling data for better instruction-tuning. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Ming Li, Pei Chen, Chenguang Wang, Hongyu Zhao, Yijun Liang, Yupeng Hou, Fuxiao Liu, and Tianyi Zhou. 2024c. Mosaic it: Enhancing instruction tuning with data mosaics. *Preprint*, arXiv:2405.13326.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. 2024d. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14255–14273, Bangkok, Thailand. Association for Computational Linguistics.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024e. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7595–7628, Mexico City, Mexico. Association for Computational Linguistics.

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.

Zelong Li, Wenyue Hua, Hao Wang, He Zhu, and Yongfeng Zhang. 2024f. Formal-llm: Integrating formal language and natural language for controllable llm-based agents. *arXiv preprint arXiv:2402.00798*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Michael Xieyang Liu, Frederick Liu, Alexander J. Fiannaca, Terry Koo, Lucas Dixon, Michael Terry, and Carrie J. Cai. 2024. "we need structured output": Towards user-centered constraints on large language model output. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI '24. ACM.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2023. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*.

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

Andrea Sottana, Bin Liang, Kai Zou, and Zheng Yuan. 2023. Evaluation metrics in the era of gpt-4: reliably evaluating large language models on sequence to sequence tasks. *arXiv preprint arXiv:2310.13800*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Peiyi Wang, Lei Li, Liang Chen, 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*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia,

Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Siyuan Wu, Yue Huang, Chujie Gao, Dongping Chen, Qihui Zhang, Yao Wan, Tianyi Zhou, Xiangliang Zhang, Jianfeng Gao, Chaowei Xiao, et al. 2024. Unigen: A unified framework for textual dataset generation using large language models. *arXiv preprint arXiv:2406.18966*.

Congying Xia, Chen Xing, Jiangshu Du, Xinyi Yang, Yihao Feng, Ran Xu, Wenpeng Yin, and Caiming Xiong. 2024. Fofo: A benchmark to evaluate llms' format-following capability. *arXiv preprint arXiv:2402.18667*.

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*.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *ArXiv*, abs/2402.13116.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Jianguo Zhang, Tian Lan, Rithesh Murthy, Zhiwei Liu, Weiran Yao, Juntao Tan, Thai Hoang, Liangwei Yang, Yihao Feng, Zuxin Liu, Tulika Awalgaonkar, Juan Carlos Niebles, Silvio Savarese, Shelby Heinecke, Huan Wang, and Caiming Xiong. 2024. Agentohana: Design unified data and training pipeline for effective agent learning. *Preprint*, arXiv:2402.15506.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023. Instruction tuning for large language models: A survey. *Preprint*, arXiv:2308.10792.

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

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

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

A Experimental Setup

A.1 Implementation Details

We utilize the prompt and code base from Vicuna (Chiang et al., 2023a) and flash attention (Dao et al., 2022) for all our experiments.

The Adam optimizer (Kingma and Ba, 2017) is utilized with the batch size of 128 and with the max token length of 2048. For training on Llama2-7B and Llama2-13B (Touvron et al., 2023), the maximum learning rate is set to 2×10^{-5} with a warmup rate of 0.03 for 3 epochs. For training on Mistral-7B (Jiang et al., 2023), the maximum learning rate is set to 1×10^{-5} with a warmup rate of 0.1 for 2 epochs. When utilizing our method, we run the augmentation process 3/2 times to simulate the epochs of training. These augmented data are then mixed together and used for training 1 epoch. All other configurations are kept the same as the baselines.

A.2 Training Datasets

We utilize 5 SFT datasets to evaluate the effectiveness of our method:

Alpaca dataset (Taori et al., 2023): This dataset consists of 52,000 instruction-following samples created using the self-instruct paradigm (Wang et al., 2023b) and OpenAI's text-davinci-003 model. Characterized as a classical dataset with moderate-quality attributes, it serves as the fundamental validation.

Alpaca-GPT4 dataset (Peng et al., 2023): This dataset is an enhanced Alpaca dataset that includes responses generated by GPT-4.

WizardLM dataset (Xu et al., 2023): This dataset is generated by the novel Evol-Instruct method, which utilizes ChatGPT-3.5 to rewrite instructions step by step into more complex ones and generate the corresponding responses. We utilize the 70k version in our method, which comprises 70,000 high-quality SFT samples.

Recycled WizardLM Dataset (Li et al., 2023a): This dataset is an improved version of the WizardLM dataset, by utilizing the Reflection-Tuning method. In the Reflection-Tuning, the initial dataset undergoes two main phases: Reflection on Instruction and Reflection on Response. In the first phase, specific criteria are carefully curated to evaluate and refine the initial instructions. During the second phase, responses are thoroughly examined and improved to align with the refined instructions. This process generates a dataset with superior quality compared to the original dataset.

DEITA dataset (Liu et al., 2023): This dataset leverages the DEITA (Data-Efficient Instruction Tuning for Alignment) method to select high-quality data from a pool comprised of several high-quality datasets, such as WizardLM and Alpaca. DEITA employs a score-first, diversity-aware data selection strategy to optimize the selection process. This strategy uses a GPT-as-a-judge scoring system that combines complexity and quality in a practical and straightforward manner. The scores are incorporated with the diversity-based selection, ensuring that all the data maintains high standards of complexity, quality, and diversity.

A.3 Evaluation Metrics

We employ 3 commonly accepted metrics for the evaluation, including **IFEval** (Instruction-Following Eval), **Pair-wise Comparison**, and **Open LLM Leaderboard**.

IFEval (Zhou et al., 2023b) is the primary evaluation metric employed in our study due to its compatibility with our motivation. It focuses on evaluating how LLMs follow various additional constraints, such as specifying a word count or requiring the inclusion of certain keywords a specified number of times. To avoid the utilization of LLMs during evaluation, it proposes 25 distinct types of verifiable instructions. There are 541 prompts in total and each of them incorporates one or more of these verifiable instructions, ensuring the comprehensiveness of the evaluation. IFEval serves as a great inspiration for our method, and there exist semantical overlappings between their verifiable instructions and our rules. However, **the specific prompts used in IFEval are kept unknown in the construction of our rule templates, avoiding potential template leakage**. Moreover, IFEval only needs to verify the responses, while our method needs to modify the responses for the training, which pushes this process a step further. Consequently, this benchmark not only facilitates a comprehensive comparison but also provides valuable insights that align with our purpose.

Pair-wise Comparison involves evaluating responses from LLMs like GPT-4, especially in open-domain contexts. This method has shown a notable alignment with human assessments, providing a credible evaluative foundation (Zheng et al., 2024; Li et al., 2023b; Sottana et al., 2023). We utilize test instruction sets from WizardLM (Xu et al., 2023) and Vicuna (Chiang et al., 2023b), comprising 218 and 80 diverse human-curated in-

structions, respectively. Following the framework by (Li et al., 2024e,d), we prompt judging LLM to rate responses on a scale from 1 to 10 across multiple dimensions such as relevance and accuracy. To mitigate positional bias (Ko et al., 2020; Wang et al., 2023a), responses are presented in two distinct sequences. Evaluation outcomes are categorized into the 'win-tie-lose' system for each instruction. One model's response can be noted as Win/Lose only if it has a Higher/Lower score in both directions. The specific evaluation prompt is presented in the Appendix D.

Open LLM Leaderboard employs Eval Harness (Gao et al., 2021a) to systematically assess generative language models through diverse tasks. Key benchmarks include ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), MMLU (Hendrycks et al., 2020), and TruthfulQA (Lin et al., 2021), collectively evaluating reasoning abilities, common-sense knowledge, and factual accuracy. This approach highlights LLM strengths and areas for improvement in critical aspects of language understanding and generation (Gao et al., 2021b).

B Examples

Examples of multi-rule augmentation are shown in Figure 3.

[Instruction]
Ensure the 2nd paragraph is enclosed in "-| |-" in the response.
Write a story of no more than 150 words about a knight who defends their kingdom from a dragon.
Make sure the 2nd sentence is enclosed within "$ $", the 4th sentence is enclosed within "$$$ $", the 6th sentence is enclosed within "## #", the 8th sentence is enclosed within "*** *" in the response.
[Response]
In the kingdom of Arvania, there lived Sir Cedric, a valiant knight, sworn to defend the land against any threat. $One day, the kingdom came under attack from a fire-breathing dragon, terrorizing the people and burning homes.$ The king called on Sir Cedric, charging him to defeat the dragon and restore peace to the land.
$$$-|Wielding his sword and shield, Sir Cedric rode out to face the dragon, fearlessly engaging it in combat.$ The battle was fierce, with the dragon's flames clashing against the knight's steel. ##But Sir Cedric was determined and unrelenting, his bravery and skill unmatched.|- #
Finally, with a mighty blow, Sir Cedric struck the dragon down, ending its reign of terror. ***The people of Arvania rejoiced, celebrating the bravery of their hero.* And Sir Cedric returned in triumph, hailed as the dragon-slaying knight.

[Instruction]
Ensure the 1st sentence is in uppercase, the 2nd sentence is in uppercase, the 3rd sentence is in uppercase in the response.
Keep the word "way" bounded by "@ @" in the response if it appears.
Describe an example of a time you used to influence in a positive way
Keep the 2nd sentence wrapped in "[[ ]]", the 4th sentence wrapped in "*** ***", and the 5th sentence wrapped in "# #" in the response.
[Response]
I RECENTLY HAD A TEAM PROJECT AT WORK WHERE I HAD TO INFLUENCE MY TEAM MEMBERS TO COMPLETE THE TASK IN THE MOST EFFICIENT @WAY@ POSSIBLE. [[I IDENTIFIED AREAS WHERE OUR TEAM COULD IMPROVE, AND I CONSTANTLY COMMUNICATED AND ENCOURAGED MY TEAMMATES TO STAY FOCUSED AND ON TRACK.]] I ALSO SOUGHT FEEDBACK ON HOW WE COULD IMPROVE THE PROCESS. ***Through my efforts, I managed to motivate my team to complete the project ahead of schedule while also maintaining high-quality work.*** #This helped to boost morale and foster collaboration between our team members.# Everyone was very satisfied with the resulting outcome.

Figure 3: Examples with multiple rules selected and implemented. The randomly generated rule-instructions are colored in violet. The upper example is augmented by 2 different rules (Paragraph Wrapping, and Sentence Wrapping); the bottom example is augmented by 3 different rules (Sentence Case, Keyword Wrapping, and Sentence Wrapping). The format differences in new responses are highlighted in red.

C Detailed Sub-Category Analysis

In this section, detailed comparisons between our models and baseline models are provided in Table 3 for further analysis of the effects of our method. It is worth noting that **the specific prompts used in IFEval are kept unknown in the construction of our rule templates, avoiding potential overfitting to the templates.**

Models trained with our recycled data outperform the baseline model consistently in "Case", "Combination", "Punctuation", and "Start End" categories, which are partially well-recycled by our method. In the "Length" category, our models are only slightly better than the baseline model although our recycling method contains this kind of constraints. After further investigation, we find the performance in this category is mainly influenced by the original data length distributions. Since our method does not introduce more data samples, thus not able to improve the performance dramatically, but the better performance indeed provides the model with a better understanding of response length. Interestingly, the performance in the "Language" category also shows consistent improvement although we do not introduce any more data. Considering the consistency between this performance and the Pair-wise comparison performance, we hypothesize this improvement is caused by the better general instruction-following abilities provided by our method.

The performance in the "Content" category presents one of the limitations of our Rule-based Recycling method, without utilizing other models or human experts to rewrite the instruction and response, it's hard for our method to modify the content of the existing response. The performances of our models in the "Json" and "Keywords" categories are merely slightly lower, which is mainly affected by the diversity of original training datasets.

Comparing the performance changes across augmentation rates, LLMs obtain better performances when the augmentation rate is higher, except for "Case", indicating the easiness of case-related constraints for LLMs to understand and learn.

Sub-Category	Case	Combination	Content	Json	Keywords	Language	Length	Punctuation	Start End
Baseline (Strict)	22.47	16.92	**75.47**	**63.06**	**44.79**	58.06	31.47	12.12	58.21
Aug Rate = 0.1	**78.65**	63.08	45.28	59.24	33.13	74.19	32.17	30.30	79.10
Aug Rate = 0.3	76.40	58.46	45.28	45.86	30.06	58.06	**35.66**	22.73	71.64
Aug Rate = 0.5	70.79	69.23	49.06	45.22	39.88	**77.42**	28.67	13.64	61.19
Aug Rate = 0.7	74.16	61.54	43.40	47.77	33.13	51.61	30.77	**77.27**	79.10
Aug Rate = 0.9	67.42	**75.38**	47.17	47.13	34.97	61.29	31.47	66.67	**82.09**
Baseline (Loose)	24.72	20.00	**75.47**	**66.88**	**48.47**	64.52	37.06	15.15	58.21
Aug Rate = 0.1	**79.78**	69.23	45.28	61.15	37.42	74.19	36.36	40.91	80.60
Aug Rate = 0.3	77.53	66.15	45.28	47.13	36.81	64.52	39.16	25.76	76.12
Aug Rate = 0.5	77.53	70.77	49.06	45.86	42.94	**80.65**	32.17	15.15	65.67
Aug Rate = 0.7	77.53	70.77	43.40	48.41	35.58	58.06	35.66	**84.85**	80.60
Aug Rate = 0.9	70.79	**80.00**	47.17	48.41	40.49	67.74	**39.16**	69.70	**85.07**

Table 3: Sub-category performance on IFEval benchmark of Mistral-7B finetuned with RULER-augmented Alpaca-GPT4 data. The top section represents the performance by the strict criterion while the bottom represents the loose.

D Evaluation Metrics

The prompt for pair-wise comparison is shown in Figure 4.

Prompt for Performance Evaluation

System Prompt
You are a helpful and precise assistant for checking the quality of the answer.

User Prompt
[Question]
Question
[The Start of Assistant 2's Answer]
Answer 2
[The End of Assistant 2's Answer]
[The Start of Assistant 2's Answer]
Answer 2
[The End of Assistant 2's Answer]

We would like to request your feedback on the performance of two AI assistants in response to the user question displayed above.
Please rate the helpfulness, relevance, accuracy, level of details of their responses. Each assistant receives an overall score on a scale of 1 to 10, where a higher score indicates better overall performance.
Please first output a single line containing only two values indicating the scores for Assistant 1 and 2, respectively. The two scores are separated by a space. In the subsequent line, please provide a comprehensive explanation of your evaluation, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment.

Figure 4: The prompt we used to request GPT4 to evaluate the responses.

E Predefined Rules

In this section, we will dive into the predefined rules describing each constraint specifically.

Keyword Appearance simulates the scenario where specific keywords are required to appear in the responses. In this rule, several non-stop words are randomly selected from the original data sample and used as the desired characteristics. The placeholder *Keyword* in the constraint template will be replaced by the sampled keyword as the rule-instruction. This process can be repeated to simulate the constraints on multiple keywords. The augmented instruction is the concatenation of the original instruction and rule-instruction. The original response does not need to be modified in this rule and is directly used as the augmented response.

Keyword Frequency simulates controlling the frequency of specific keywords in generated responses. In this rule, several non-stop words and their frequencies are randomly sampled and used as the desired characteristics. There are three random sub-situations in the rule: More, Less, or Equal. In the "Equal" situation, the placeholders {*N*} and {*Keyword*} will be directly replaced by the sampled keyword and its frequency. In the "More" or "Less" situations, a small random number x will be randomly generated to adjust the keyword frequency to meet the desired constraint template, such as "Ensure there are more than {*N* - x} {*Keyword*}" or "Ensure there are fewer than {*N* + x} {*Keyword*}." This process can be repeated to simulate constraints on multiple keywords. The augmented instruction is the concatenation of the original instruction and rule-instruction. The original response does not need to be modified in this rule and is directly used as the augmented response.

Num of Adjectives simulates controlling the total number of adjectives in generated responses. In this rule, the adjectives in the original response are identified and counted using part-of-speech tagging (POS). There are three possible sub-situations in the rule: More, Less, or Exact. In the "Exact" situation, the placeholder {*N*} will be replaced by the number of adjectives. In the "More" or "Less" situations, a small random number x is randomly generated to adjust *N* to meet the constraints, such as "Ensure the response has more than {*N* - x} adjectives" or "Ensure the response has fewer than {*N* + x} adjectives." This process can only be used once for each sample. The augmented instruction is the concatenation of the original instruction and rule-instruction. The original response does not need to be modified in this rule and is directly used as the augmented response.

Num of Nouns simulates controlling the number of nouns in generated responses, similar to the "Num of Adjectives".

Num of Verbs simulates controlling the number of verbs in generated responses.

Num of Characters simulates controlling the number of characters in generated responses.

Num of Letters simulates controlling the number of letters in generated responses.

Num of Words This rule simulates controlling the number of words in generated responses.

Num of Sentences simulates controlling the number of sentences in generated responses. The sentences from the original response are segmented by utilizing dependency parsing.

Num of Paragraphs simulates controlling the number of paragraphs in generated responses. The paragraphs from the original response are segmented by regular expressions.

Num of Bullets simulates controlling the number of bullet points in generated responses. The bullet points from the original response are segmented by regular expressions.

Instruction Repetition simulates the scenario where the LLM is requested to repeat the instructions before providing the response. This process can be applied only once for each instruction. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the concatenation of repeated original instruction and the response.

Response Repetition simulates the scenario where the LLM is requested to repeat the responses several times. In this rule, the response is repeated {*N*} times, where {*N*} is a random number. This process can only be applied once per data sample. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the concatenation of *N* identical responses.

UP Case simulates requesting the entire response is required in uppercase. In this rule, the original response is converted to uppercase format entirely. This process can only be used once for each response. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the all-uppercase version of the original response.

Rule Type	Rule Name	Example of an Instruction Template for the Rule
Keyword Frequency	Keyword Appearance	Ensure {*Keyword*} is in the response.
Keyword Frequency	Keyword Frequency	Ensure there are more/less/exact {N} {*Keyword*} in the response.
Number Constraint	Num of Adjectives	Ensure the response has more/less/exact {N} adjectives.
Number Constraint	Num of Nouns	Ensure the response has more/less/exact {N} nouns.
Number Constraint	Num of Verbs	Ensure the response has more/less/exact {N} verbs.
Number Constraint	Num of Characters	Ensure the response has more/less/exact {N} characters.
Number Constraint	Num of Letters	Ensure the response has more/less/exact {N} letters.
Number Constraint	Num of Words	Ensure the response has more/less/exact {N} words.
Number Constraint	Num of Sentences	Ensure the response has more/less/exact {N} sentences.
Number Constraint	Num of Paragraphs	Ensure the response has more/less/exact {N} paragraphs.
Number Constraint	Num of Bullets	Ensure the response has more/less/exact {N} bullet points.
Repetition	Instruction Repetition	Repeat the instruction before providing the response.
Repetition	Response Repetition	Repeat the response {N} times.
Case All	Up Case	Ensure the response is all in upper case.
Case All	Low Case	Ensure the response is all in lowercase.
Case Target	Letter Case	Ensure all the letters {x} in the response are in uppercase.
Case Target	Keyword Case	Ensure all the word {*Keyword*} in the response are in uppercase.
Case Target	Sentence Case	Ensure {i}-th sentence in the response is in uppercase.
Case Target	Paragraph Case	Ensure {i}-th paragraph in the response is in uppercase.
Punctuation All	All Removal	Ignore all punctuation in the response.
Punctuation All	All Replacement	Use {*Symbol*} to replace all punctuation in the response.
Punctuation Target	Target Removal	Ignore {*Punctuation*} punctuation in the response.
Punctuation Target	Target Replacement	Use {*Symbol*} to replace {*Punctuation*} in the response.
Format Wrapping	Keyword Wrapping	Ensure every {*Keyword*} is wrapped in {*Format*} in the response.
Format Wrapping	Sentence Wrapping	Ensure {i}-th sentence is wrapped in {*Format*} in the response.
Format Wrapping	Bullet Wrapping	Ensure {i}-th bullet point is wrapped in {*Format*} in the response.
Format Wrapping	Paragraph Wrapping	Ensure {i}-th paragraph is wrapped in {*Format*} in the response.
Formatted Repeating	Instruction Wrapping	Repeat the instruction in {*Format*} before providing the response.
Formatted Repeating	Response Wrapping	Repeat the response {N} times in {*Format*}.

Table 4: The list of predefined constraint rules. Each rule contains (1) a set of constraint templates that serve as additional rule-instructions, on constraints the LLM should follow, and (2) specified methods that alternately edit the instruction and response to reach an alignment between them.

Low Case simulates requesting the entire response is required in lowercase.

Letter Case simulates the scenario where specific types of letters in the response are required to be in uppercase. In this rule, the specific letter x is sampled from the response, and all occurrences of this letter in the response are capitalized. This process can be repeated on different random letters. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the original response with all specific letters in uppercase.

Keyword Case simulates the scenario where specific keywords in the response are required to be in uppercase. The specific keyword *Keyword* is sampled from the response.

Sentence Case simulates the scenario where the specific sentences in the response are required to be in uppercase. The index of the sentence i is randomly selected within the total number of sentences in the response.

Paragraph Case simulates the scenario where the specific paragraphs in the response are required to be in uppercase. The index of the paragraph

i is randomly selected within the total number of paragraphs in the response

All Removal simulates controlling LLM to ignore the use of punctuation. In this rule, the punctuation marks in the response will be removed completely, and the new response will serve as the augmented response. This process can only be used once for each sample.

All Replacement simulates controlling LLM to replace all the original punctuation with a predefined symbol {*Symbol*}. In this rule, the punctuation marks in the response will be replaced, and the new response will serve as the augmented response. This process can only be used once for each sample.

Target Removal simulates the scenario where a specific type of punctuation mark {*Punctuation*} in the response is ignored. In this rule, a random type of punctuation is identified, and all occurrences of this mark are removed, the new response will serve as the augmented response. This process can only be used once for each sample to avoid confusion.

Target Replacement simulates the scenario where a specific type of punctuation mark {*Punctuation*} in the response is replaced by a specified symbol {*Symbol*}. In this rule, a random type of punctuation mark is identified, and all occurrences of this mark will be replaced by a predefined symbol in the response. This process can only be used once.

Keyword Wrapping simulates the scenario where specific keywords in the response are required to be wrapped in a specified format. In this rule, a randomly chosen {*Keyword*} is identified, and all occurrences of this keyword are wrapped in the randomly specified {*Format*} in the response. This process can be repeated several times on different words with various formats. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the original response with all keywords wrapped in the format.

Sentence Wrapping simulates the scenario where specific sentences in the response are required to be wrapped in a specified format. The index of the sentence *i* is randomly selected within the total number of sentences in the response.

Bullet Wrapping simulates the scenario where specific bullet points in the response are required to be wrapped in a specified format. The index of the bullet point *i* is randomly selected within the total number of bullet points in the response.

Paragraph Wrapping simulates the scenario where a specific paragraph in the response are required to be wrapped in a specified format. The index of the paragraph *i* is randomly selected within the total number of paragraphs in the response.

Instruction Wrapping simulates a scenario where the original instruction is required to be repeated in a specified format before providing the response. In this rule, the original instruction is restated with wrapping in the randomly chosen {*Format*} before giving the actual response. This process can be applied only once for each instruction. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the concatenation of the original instruction wrapped in the specific format and the response.

Response Wrapping simulates a scenario where the response wrapped in the specific format is required to be repeated several times. In this rule, the response is repeated {*N*} times wrapeed in the specified {*Format*}, with {*N*} and {*Format*} being randomly selected. This process can be applied only once. The augmented instruction is the concatenation of the original instruction and rule-instruction. The augmented response is the concatenation of the repeated response wrapped in the format.