MulDimIF: A Multi-Dimensional Constraint Framework for Evaluating and Improving Instruction Following in Large Language Models

Junjie Ye[1*], Caishuang Huang[1*], Zhuohan Chen[1], Wenjie Fu[1], Chenyuan Yang[1], Leyi Yang[1], Yilong Wu[1], Peng Wang[2], Meng Zhou[3], Xiaolong Yang[3], Tao Gui[1], Qi Zhang[1,4,5†], Zhongchao Shi[2], Jianping Fan[2], Xuanjing Huang[1]

[1]Fudan University [2]Lenovo Research [3]Tencent
[4]Shanghai Artificial Intelligence Laboratory
[5]Shanghai Key Lab of Intelligent Information Processing

jjye23@m.fudan.edu.cn, qz@fudan.edu.cn

Abstract

Instruction following refers to the ability of large language models (LLMs) to generate outputs that satisfy all specified constraints. Existing research has primarily focused on constraint categories, offering limited evaluation dimensions and little guidance for improving instruction-following abilities. To address this gap, we introduce *MulDimIF*, a multi-dimensional constraint framework encompassing three constraint patterns, four constraint categories, and four difficulty levels. Based on this framework, we design a controllable instruction generation pipeline. Through constraint expansion, conflict detection, and instruction rewriting, we construct 9,106 code-verifiable samples. We evaluate 18 LLMs from six model families and find marked performance differences across constraint settings. For instance, average accuracy decreases from 80.82% at Level I to 36.76% at Level IV. Moreover, training with data generated by our framework significantly improves instruction following without compromising general performance. In-depth analysis indicates that these gains stem largely from parameter updates in attention modules, which strengthen constraint recognition and adherence. Code and data are available in https://github.com/Junjie-Ye/MulDimIF.

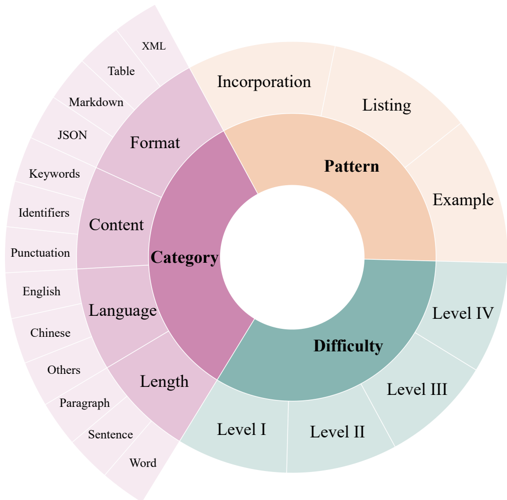


Figure 1: The hierarchical structure of the multi-dimensional constraint framework, which includes three constraint patterns, four constraint categories (subdivided into thirteen subcategories), and four levels of constraint difficulty.

1 Introduction

Instruction following is a fundamental capability of large language models (LLMs) (Bai et al., 2022; OpenAI, 2023; Reid et al., 2024; Yang et al., 2024), enabling them to generate outputs that satisfy user-specified constraints (Wen et al., 2024; Dong et al., 2025b; Zhang et al., 2025). This skill is essential in real-world applications, especially in agentic and tool-assisted workflows where outputs must adhere to strict formats such as JSON (Xi et al., 2023; Deng et al., 2024; Ye et al., 2024). Even small deviations can cause parsing failures and downstream system breakdowns (Ye et al., 2025).

Existing research has approached this challenge along two main lines. Benchmarking efforts typically rely on template-based instructions (Zhou et al., 2023; He et al., 2024c), evaluated either through code verification or LLM-as-a-judge methods (Jiang et al., 2024). In parallel, training-focused studies have explored improving instruction following via tree search (Cheng et al., 2024), reinforcement learning (RL) (He et al., 2024a), and other data-centric strategies.

However, these efforts face key limitations. Benchmarking work largely emphasizes the diversity of constraint categories (Zhou et al., 2023; Sun et al., 2024; Qin et al., 2024; Wen et al., 2024), while relying on narrow evaluation dimensions that cannot fully characterize instruction-following

[*]Equal contributions.
[†]Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 2078–2104
July 2-7, 2026 ©2026 Association for Computational Linguistics

ability. Meanwhile, training approaches often boost benchmark scores through data engineering (Zhang et al., 2024a; Cheng et al., 2024; Dong et al., 2024), but rarely examine the mechanisms driving these improvements. As a result, we still lack deeper insight into the factors that shape instruction-following performance.

To address this gap, we present *MulDimIF*, a multi-dimensional constraint framework for evaluating and improving instruction following in LLMs. As shown in Figure 1, the framework introduces three constraint patterns, namely example, listing, and incorporation, which capture common user instruction formats. Constraints are organized into four major categories, namely content, language, format, and length, which further divide into thirteen subcategories. The framework also defines four difficulty levels based on the complexity of constraint combinations within instructions. Building on this framework, we design an instruction generation pipeline consisting of constraint expansion, conflict detection, and instruction rewriting. This pipeline enables controlled transformation of initial instructions into diverse, constraint-rich variants. Using it, we generate 9,106 code-verifiable data instances.

We evaluate 18 LLMs across six model families and uncover significant variation in their ability to follow different forms of constraints. Notably, while models perform well on the example pattern, they struggle with listing and incorporation patterns, emphasizing the challenges posed by complex instructions and the benefits of few-shot prompting (Brown et al., 2020). On average, performance declines from 80.82% at Level I to 36.76% at Level IV, with even the best model scoring only 67.50% overall.

To leverage these findings, we train LLMs using the GRPO algorithm (Shao et al., 2024) on data generated by our framework. The trained models show significantly improved instruction-following abilities without sacrificing general performance. Parameter-level analysis and case studies indicate that these gains primarily result from updates within attention modules, which more effectively align models' focus with specified constraints.

Our contributions are summarized as follows:

- We propose a multi-dimensional constraint framework that evaluates and improves instruction following in LLMs;
- We design a controllable instruction generation pipeline that transforms plain instructions into constraint-rich variants;
- We generate 9,106 diverse constraint-rich data instances, evaluate the instruction-following abilities of 18 LLMs, and improve model performance through targeted training;
- We conduct parameter-level analysis, showing that improvements in instruction following primarily arise from updates within attention modules.

2 Related Works

Evaluation of Instruction Following Evaluating the instruction-following abilities of LLMs has become a central focus in recent research (Zhang et al., 2024b). Benchmarks such as IFEval (Zhou et al., 2023), FollowEval (Jing et al., 2023), and FollowBench (Jiang et al., 2024) evaluate models on dimensions like logical reasoning and stylistic consistency, using either code-based or LLM-based evaluations. Multi-IF (He et al., 2024c) extends this to multilingual, multi-turn dialogue settings, while InfoBench (Qin et al., 2024) decomposes complex instructions into simpler subtasks to evaluate execution accuracy. CIF-Bench (Li et al., 2024) focuses on the generalization abilities of Chinese LLMs under zero-shot scenarios. Despite their breadth, many of these benchmarks rely on templated or highly constrained prompts, which limits their ability to capture real-world instruction diversity and support fine-grained evaluation. Our work addresses these limitations by introducing a multi-dimensional constraint framework comprising three constraint patterns, four constraint categories, and four difficulty levels. Built on this framework, we develop a controllable instruction generation pipeline that enhances diversity and complexity through constraint expansion, conflict detection, and instruction rewriting.

Improving of Instruction Following A range of algorithms have been proposed to improve the instruction-following performance of LLMs. RL approaches such as PPO (Schulman et al., 2017) and DPO (Rafailov et al., 2023) optimize model behavior based on user preferences. IOPO (Zhang et al., 2024a) augments this by aggregating question-answer pairs across datasets to enrich preference signals and refine the optimization objective. Coninfer (Sun et al., 2024) adopts

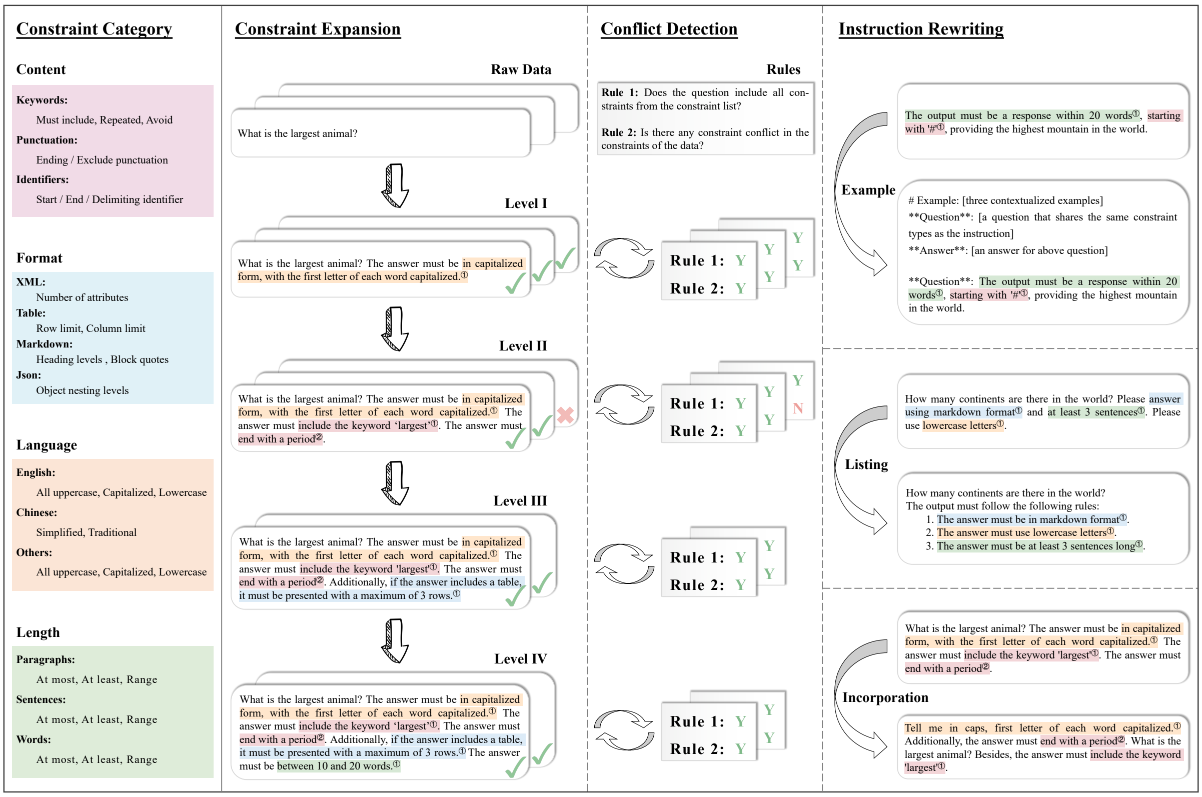


Figure 2: Illustration of the instruction generation pipeline. **Constraint Expansion**: Randomly selects a constraint category not yet included in the instruction and adds 1–2 specific constraints. **Conflict Detection**: Identifies whether the new instruction introduces redundant constraints or conflicts, and discards conflicting instructions. **Instruction Rewriting**: Rewrites the remaining instructions based on different constraint patterns.

a curriculum learning approach, incrementally increasing task difficulty during fine-tuning to improve constraint handling. While these methods yield measurable gains, they often lack in-depth analysis of the model characteristics driving these improvements, which limits their interpretability and generalizability. To fill this gap, we introduce a comprehensive data construction pipeline, enabling the creation of high-quality instruction-following datasets. Our parameter-level analysis suggests that a significant portion of the performance improvement arises from tuning the model's attention mechanisms, which enhances its ability to recognize and comply with constraints.

3 Approaches

3.1 Multi-Dimensional Constraint Framework

Existing studies (Zhou et al., 2023; Jing et al., 2023; Jiang et al., 2024) often focus solely on the diversity of constraint categories, limiting the dimension for evaluating and improving the instruction-following ability. To address this limitation, we propose a multi-dimensional constraint framework, as shown in Figure 1.[1]

3.1.1 Constraint Pattern

Drawing inspiration from publicly available guidelines for writing instructions (Saravia, 2022), we identify three common patterns used to introduce constraints during interactions with LLMs.

Example The example pattern involves adding several question-answer pairs that share the same constraint type as the instruction to be followed. This method enhances the model's ability to comply with constraints through contextualized examples, a technique commonly known as in-context learning (Brown et al., 2020).

Listing The listing pattern presents constraints in a clearly structured, point-by-point format. This approach provides explicit communication of constraint requirements, making it especially effective in zero-shot scenarios.

[1]Examples are available in Appendix C.

Data	Constraint Pattern			Constraint Category				Constraint Difficulty				Total
	Example	Listing	Incorporation	Content	Format	Language	Length	Level I	Level II	Level III	Level IV	
Training	1225	3308	3373	8888	9850	6168	9541	610	1614	2522	3160	7906
Test	400	400	400	1175	1210	694	1101	300	300	300	300	1200

Table 1: Distribution of training and test data constructed on the automated instruction generation pipeline.

Incorporation The incorporation pattern integrates constraints directly into the instruction, rather than listing them separately. While this approach maintains fluency, it may make it more difficult for LLMs to clearly interpret individual constraint requirements.

3.1.2 Constraint Category

Beyond the method of presentation, the types of constraints can vary widely. To enable fine-grained analysis, we categorize constraints into four main categories with thirteen subcategories.

Content Content constraints restrict the elements present in the model's output. These can be further divided into three subcategories: ensuring the inclusion of specific keywords, adhering to particular punctuation requirements, or referencing predefined identifiers.

Format Format constraints require the output to follow specific structural rules, often necessary for post-processing tasks. Common examples include outputs in XML, Table, Markdown, or JSON.

Language Language constraints specify the language to be used in the output, which is fundamental in translation tasks (Ye et al., 2023). Based on the language category, constraints are classified into English, Chinese, or other languages.

Length Length constraints enforce limits on the output's size. Depending on granularity, these constraints can apply at the paragraph level, sentence level, or word level.

3.1.3 Constraint Difficulty

In addition to presentation and type, the number of constraints also impacts task difficulty. We define four difficulty levels based on the number and variety of constraints present in the instruction.

Level I Level I includes an instruction with a single type of constraint, containing one or two individual constraint elements.

Level II Level II includes an instruction with two types of constraints, comprising a total of two to four individual constraint elements.

Level III Level III includes an instruction with three types of constraints, comprising a total of three to six individual constraint elements.

Level IV Level IV includes an instruction with four types of constraints, comprising a total of four to eight individual constraint elements.

3.2 Instruction Generation Pipeline

Building upon the multi-dimensional constraint framework, we introduce a controllable pipeline that transforms raw instructions into constrained versions that can be verified through code, as illustrated in Figure 2.

Constraint Expansion Constraint expansion involves adding new constraints to a given instruction. Specifically, we randomly select a constraint category not yet covered and add one or two specific constraints from that category. This process progressively generates instructions across varying levels of constraint difficulty.

Conflict Detection Conflict detection ensures the soundness of instructions after constraint expansion. It consists of two checks: first, verifying that the newly specified constraints have been correctly incorporated; second, ensuring that the constraints do not conflict with each other (e.g., requiring a sentence to be entirely lowercase while also demanding the presence of uppercase words). If either check fails, the instruction is discarded. Otherwise, it is retained, and constraint expansion continues until difficulty Level IV is reached.

Instruction Rewriting Instruction rewriting enhances instruction diversity by transforming a given instruction to match a specified pattern. Specifically, we randomly select a constraint pattern and rewrite the instruction accordingly. When handling example-based constraints, we uniformly select three question-answer pairs that share the same constraint subcategories as the original instruction to serve as contextual examples.

Family	Version	Constraint Pattern			Constraint Category				Constraint Difficulty				Overall
		Example	Listing	Incorporation	Content	Format	Language	Length	Level I	Level II	Level III	Level IV	
Open-Source LLMs (Non Reasoning)													
LLaMA3.1	Instruct-8B	40.25	36.00	32.25	80.13	64.74	44.28	49.60	64.67	40.67	27.33	12.00	36.17
	Instruct-70B	68.00	54.25	48.25	86.62	73.65	80.90	65.73	78.00	61.33	51.33	36.67	56.83
Qwen3	8B-Direct.	63.00	55.00	51.00	84.03	72.27	83.00	69.89	87.67	60.33	45.67	31.00	56.17
	14B-Direct.	64.00	63.00	52.00	87.92	77.00	89.87	65.86	88.00	66.67	50.67	33.67	60.00
	32B-Direct.	70.00	59.00	49.00	87.14	78.29	78.29	68.01	85.00	65.67	51.00	34.00	58.92
Open-Source LLMs (Reasoning)													
DeepSeek-R1-Distill-LLaMA	Instruct-8B	42.00	28.00	28.00	80.00	50.44	44.72	53.09	59.67	40.33	18.33	12.33	32.67
	Instruct-70B	59.50	64.00	57.50	84.55	83.69	84.80	65.32	77.00	63.00	57.00	44.33	60.33
Qwen3	8B-Reason.	67.75	61.25	54.00	85.84	71.02	87.70	72.58	87.00	68.33	52.67	36.00	61.00
	14B-Reason.	71.50	64.00	57.75	87.79	77.29	**92.47**	72.31	86.00	72.33	58.67	40.67	64.42
	32B-Reason.	70.50	69.50	59.50	87.14	83.06	90.30	74.19	86.33	72.67	61.00	46.00	66.50
Closed-Source LLMs													
Gemini1.5	Flash	71.50	61.00	64.50	88.70	80.68	87.84	74.06	86.00	68.00	57.67	51.00	65.67
	Pro	**73.50**	61.75	**65.25**	87.40	81.81	88.28	75.67	**86.67**	70.00	59.00	51.67	66.83
Claude3.5	Haiku	44.25	53.50	52.00	84.29	82.81	49.06	68.01	71.33	51.67	41.67	35.00	49.92
	Sonnet	72.50	**69.00**	61.00	86.62	83.44	84.23	**76.21**	82.67	**71.67**	**60.67**	**55.00**	**67.50**
GPT	3.5-Turbo	49.00	41.25	38.00	84.94	70.39	42.98	66.26	77.33	48.00	30.33	15.33	42.75
	4-Turbo	70.75	59.25	52.25	**89.09**	79.17	77.57	73.25	82.67	68.00	55.00	37.33	60.75
	4o-Mini	67.50	67.00	59.00	85.71	**84.57**	84.37	72.04	84.33	70.00	59.00	44.67	64.50
	4o	70.50	62.50	59.00	87.92	84.44	82.78	69.35	84.33	69.33	57.33	45.00	64.00

Table 2: Results of the evaluation of LLMs' instruction-following ability across different dimensions. 'Overall' denotes the overall score. The best results in each dimension are highlighted in **bold**.

Dataset We randomly sample data from ShareGPT[2] and generate 9,106 data instances using the aforementioned process. To ensure quality, each instance undergoes *manual review*,[3] and Python validation code is written for every data point to enable automated evaluation. Statistical details of the data are presented in Table 1.[4]

4 Evaluations of LLMs

4.1 Models

Based on the test set described in Table 1, we conduct an evaluation of 18 LLMs from six model families, including four open-source and three closed-source. Among the open-source LLMs, we select *LLaMA3.1-Instruct-8B* and *LLaMA3.1-Instruct-70B* from the **LLaMA3.1 family** (Team, 2024); *DeepSeek-R1-Distill-LLaMA-8B* and *DeepSeek-R1-Distill-LLaMA-70B* from the **DeepSeek-R1-Distill-LLaMA family** (DeepSeek-AI et al., 2025); and *Qwen3-8B*, *Qwen3-14B*, and *Qwen3-32B* from the **Qwen3 family** (Yang et al., 2025). Since Qwen3 includes specialized optimizations for reasoning, we evaluate its models in both non-reasoning mode (i.e., *-Direct.*) and reasoning mode (i.e., *-Reason.*). For the closed-source LLMs, we include *Gemini1.5-Flash* and *Gemini1.5-Pro* from the **Gemini1.5 family** (Reid et al., 2024); *Claude3.5-Haiku* and *Claude3.5-Sonnet* from the **Claude3.5 family** (Bai et al., 2022); and *GPT-3.5-Turbo*, *GPT-4-Turbo*, *GPT-4o-Mini*, and *GPT-4o* from the **GPT family** (OpenAI, 2023).[5]

4.2 Experimental Setup

For instruction generation, we use GPT-4o.[6] To ensure that each model's capabilities are accurately represented, we use the built-in chat template for open-source models and the official API interface for closed-source models. For consistency and reproducibility, we apply greedy decoding across all evaluations. For LLMs that support a reasoning mode, we evaluate only the final answer and exclude the intermediate reasoning process.

4.3 Main Results

Table 2 summarizes the results of our multi-dimensional evaluation of various LLMs. Several key findings emerge from this analysis.

There is substantial variation in current LLMs' ability to follow different constraint forms. For constraint patterns, most models perform best on the example pattern, while performance steadily declines under the incorporation patter. This underscores the effectiveness of in-context learning (Min et al., 2022) but also highlights the persistent difficulty of adhering to free-form constraints. Across constraint categories, models generally follow content-based constraints well but struggle with language- and length-related

[2] `https://www.sharegpt.com`

[3] More details can be found in Appendix B.

[4] A comparison between MulDimIF and other approaches is provided in Appendix A.

[5] More information can be found in Appendix D.

[6] The prompts are provided in Appendix F.

Dataset	Ability	# Number
Training		
Ours	Instruction Following	7906
Test		
Ours	Instruction Following	1200
IFEval	Instruction Following	541
Multi-IF	Instruction Following	13447
MMLU	Knowledge	14042
GSM8K	Reasoning	1319
MATH	Reasoning	5000
HumanEval	Coding	164
MBPP	Coding	257

Table 3: Overview of the training and test datasets used when improving instruction-following capabilities.

ones, often producing additional content to improve naturalness at the cost of constraint violations. In terms of difficulty, average accuracy drops sharply from 80.82% at Level I to 36.76% at Level IV, suggesting that current models face considerable challenges when handling multiple or more complex constraints.

Scaling up models generally improves their ability to follow instructions, though exceptions are observed. Within most families, larger models exhibit stronger adherence, especially in challenging scenarios such as level IV. This trend is consistent with prior findings on scaling laws for LLMs (Kaplan et al., 2020; Chung et al., 2022). However, the Qwen3 and GPT families deviate from this pattern. For instance, Qwen3-32B-Direct. underperforms relative to Qwen3-14B-Direct., and GPT-4o falls short compared to GPT-4o-Mini. Such anomalies may reflect an alignment tax (Ouyang et al., 2022), whereby optimization for broader capabilities comes at the expense of precision in instruction following.

Optimizing reasoning capabilities has raised the lower bound of a model's instruction-following ability without enhancing its upper bound. In the Qwen3 family, enhancements in reasoning mode primarily benefit previously weaker dimensions (e.g., incorporation, length, and Level IV) while providing little improvement in stronger ones (e.g., example, content, and Level I). Additionally, DeepSeek-R1-Distill-LLaMA-Instruct-8B shows reduced performance compared to LLaMA3.1-Instruct-8B. These findings underscore the urgent need for reasoning optimization strategies specifically designed to enhance instruction-following capabilities.

5 Improvements of LLMs

As shown in Table 1, we construct 7,906 single-turn, constraint-rich instructions designed for RL to enhance the instruction-following abilities of LLMs.[7] Our approach strengthens these abilities while preserving overall performance. Analysis indicates that the observed gains arise primarily from updates to attention modules, which increase the model's sensitivity to task-specific constraints.

5.1 Test Set

To compare model performance before and after RL, we evaluate four abilities: 1) **Instruction Following.** We evaluate on our test set, along with IFEval and Multi-IF. 2) **Knowledge.** General knowledge is evaluated using MMLU (Hendrycks et al., 2021a). 3) **Reasoning.** We use GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b) to measure logical and mathematical reasoning. 4) **Coding.** Programming ability is evaluated with HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021). The information of the data is shown in Table 3.

5.2 Experimental Setup

We conduct experiments on six LLMs without more than 14 billion parameters, applying the GRPO algorithm. The setup includes a batch size of 1,024, mini-batch size of 512, 32 rollouts per update, and a learning rate of 1e-6. We use a sampling temperature of 0.7 and set the maximum output length to 8,192 tokens. The reward function is defined as the number of constraints satisfied in the output. Training is performed for one epoch on eight NVIDIA A800 GPUs. For evaluation, we apply each model's official chat template and use greedy decoding for consistency.

5.3 Results and Analysis

Performance Improvements Figure 3 presents the performance of individual LLMs before and after applying GRPO.[8] The results demonstrate substantial performance gains on our custom test set, with LLaMA3.1-Instruct-8B notably outperforming other models. Importantly, these improvements extend to out-of-domain instruction-following benchmarks and significantly enhance performance in multi-turn dialogue scenarios (i.e., Multi-IF), despite training being conducted solely

[7]The full distribution of training data is provided in Table 1.
[8]Detailed results are provided in Appendix E.

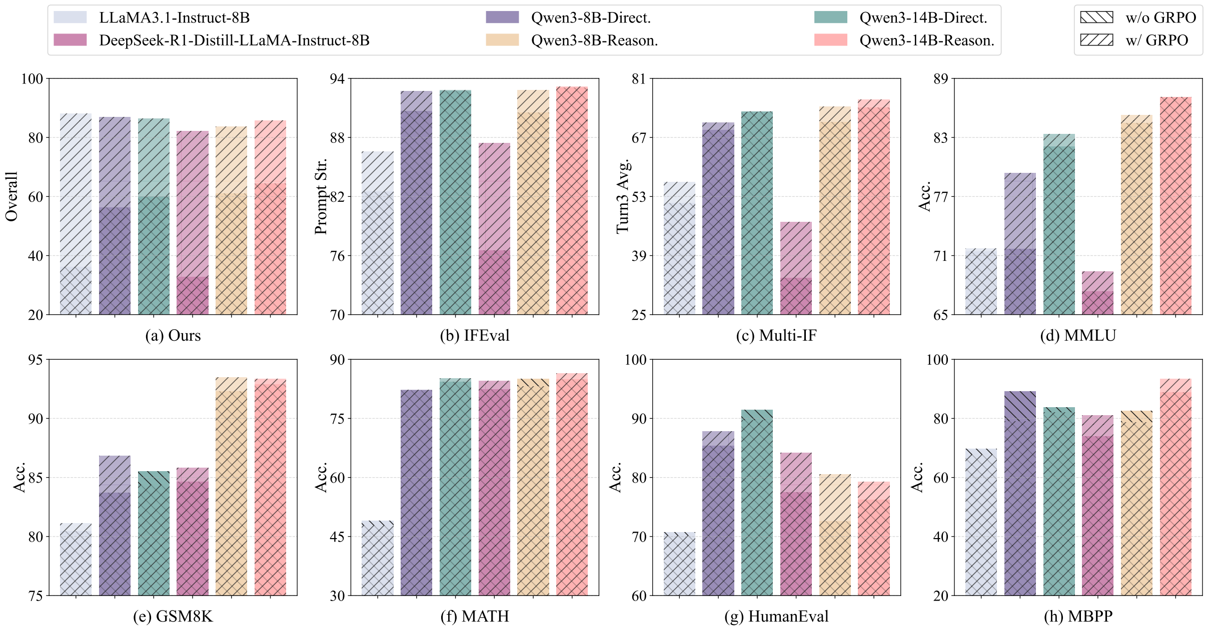
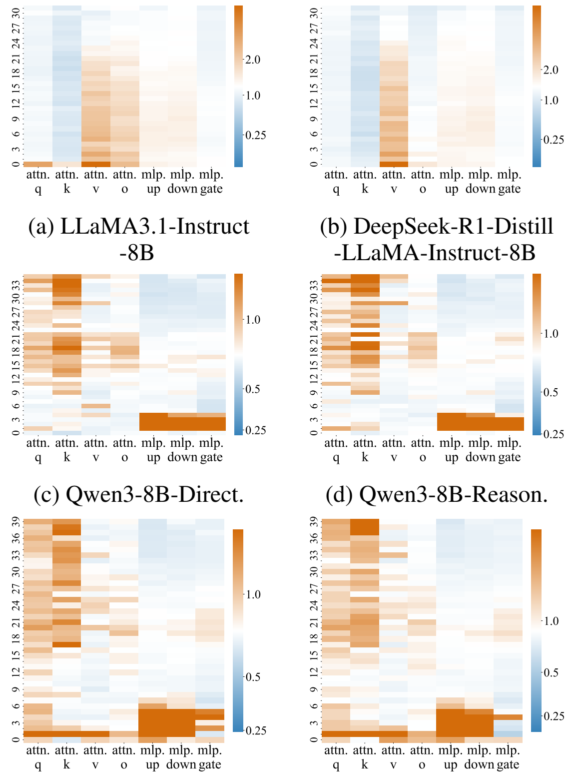


Figure 3: Performance comparison of each LLM on the test sets before and after applying GRPO.

on single-turn data. This suggests that the data generated by our multi-dimensional constraint framework exhibits strong generalization ability. Furthermore, although our training is focused on improving instruction-following abilities, it does not degrade general-purpose performance. On general benchmarks, post-GRPO LLMs maintain parity with their original counterparts and, in some cases (e.g., MMLU), show clear improvements. These findings indicate that our pipeline produces data that is both compatible with and complementary to existing training corpora, enabling straightforward performance enhancements when integrated into current LLMs.

Parameter-Level Analysis To better understand the sources of performance improvement, we conduct a parameter-level analysis. We compute the relative change rates of model parameters following GRPO, broken down by model modules, and summarize the results in Figure 4. Notably, the most substantial updates occur in attention modules, suggesting that GRPO primarily refines the model's attention mechanisms. These changes are distributed relatively uniformly across layers, indicating a global rather than localized adjustment. Overall, applying GRPO with our data effectively tunes the model's attention distribution, enhancing its ability to identify and focus on critical input information and thereby improving its instruction-following and general performance.



(a) LLaMA3.1-Instruct -8B
(b) DeepSeek-R1-Distill -LLaMA-Instruct-8B
(c) Qwen3-8B-Direct.
(d) Qwen3-8B-Reason.
(e) Qwen3-14B-Direct.
(f) Qwen3-14B-Reason.

Figure 4: Parameter change rates of LLMs after GRPO relative to the original ones across different modules.

Case Studies To visualize how modifications in attention mechanisms affect model behavior, we adopt the information flow analysis method proposed by Wang et al. (2023). We compute

<table>
<tr><th>Case (w/o GRPO)</th><th>Case (w/ GRPO)</th></tr>
<tr><td colspan="2">LLaMA3.1-Instruct-8B: After applying GRPO, the model becomes more attentive to nuanced constraints, especially specific details like the ‘5 words’ requirement, ensuring it meets the length constraint.</td></tr>
<tr><td><|begin_of_text|><|begin_of_text|><|start_header_id|>system<|end_header_id|>
Cutting Knowledge Date: December 2023
Today Date: 26 Jul 2024
<|eot_id|><|start_header_id|>user<|end_header_id|>
What is the color of an apple ? When answering , ensure that the response is concise , with a maximum of 5 words . It should include the keyword 'red ' and conclude with a period . Additionally , format the answer as a level 1 heading in Markdown , and capitalize the first letter of each word in English .
<|eot_id|><|start_header_id|>assistant<|end_header_id|>
The Color Of An Apple
Red, Green, And Sometimes Yellow. <|eot_id|></td><td><|begin_of_text|><|begin_of_text|><|start_header_id|>system<|end_header_id|>
Cutting Knowledge Date: December 2023
Today Date: 26 Jul 2024
<|eot_id|><|start_header_id|>user<|end_header_id|>
What is the color of an apple ? When answering , ensure that the response is concise , with a maximum of 5 words . It should include the keyword 'red ' and conclude with a period . Additionally , format the answer as a level 1 heading in Markdown , and capitalize the first letter of each word in English .
<|eot_id|><|start_header_id|>assistant<|end_header_id|>
The Red Color. <|eot_id|></td></tr>
<tr><td colspan="2">Qwen3-8B-Direct.: After applying GRPO, the model increased its attention to details such as ‘capitalized’ and ‘each word’, ensuring that every word in the output is capitalized.</td></tr>
<tr><td><|im_start|> user
How should " Gaza " be pronounced in Arabic ?
The output must follow the following rules :
1 . Must include the phonetic transcription of " Gaza " in Arabic .
2 . Ending punctuation must be a period .
3 . The response must use capitalized letters for each word . <|im_end|>
<|im_start|> assistant
<think>

</think>
GAZA is pronounced as /'ɡaːzaː/ in Arabic. <|im_end|></td><td><|im_start|> user
How should " Gaza " be pronounced in Arabic ?
The output must follow the following rules :
1 . Must include the phonetic transcription of " Gaza " in Arabic .
2 . Ending punctuation must be a period .
3 . The response must use capitalized letters for each word . <|im_end|>
<|im_start|> assistant
<think>

</think>
GAZA IS PRONOUNCED AS /'Zaːzaː/ IN ARABIC. <|im_end|></td></tr>
<tr><td colspan="2">DeepSeek-R1-Distill-LLaMA-Instruct-8B: After applying GRPO, the model places greater emphasis on factors such as length, keywords, and content, enabling it to meet multiple complex constraints.</td></tr>
<tr><td>< | begin_of_sentence | >< | User | > What does asbestos feel like ? Please provide a detailed description that is at least three sentences long , ensuring a comprehensive understanding of its texture and sensory characteristics . It is important to note that the description must be sufficiently detailed to meet the requirement of being at least three sentences in length . Additionally , the answer must include a level 2 heading titled ' Asbestos Texture ' to clearly introduce the section on its texture , adhering to the specified markdown format .
< | Assistant | ><think>
...
</think>

Asbestos Texture
Asbestos feels lightweight and smooth, with a fine, almost silky texture. Its fibers are soft to the touch, creating a sensation similar to handling a smooth fabric or mineral powder. When manipulated, it has a slight flexibility but does not feel gritty or rough, making it appear almost weightless in the hands. < | end_of_sentence | ></td><td>< | begin_of_sentence | >< | User | > What does asbestos feel like ? Please provide a detailed description that is at least three sentences long , ensuring a comprehensive understanding of its texture and sensory characteristics . It is important to note that the description must be sufficiently detailed to meet the requirement of being at least three sentences in length . Additionally , the answer must include a level 2 heading titled ' Asbestos Texture ' to clearly introduce the section on its texture , adhering to the specified markdown format .
< | Assistant | ><think>
...
</think>

Asbestos Texture
Asbestos has a unique texture that is both lightweight and crisp, resembling the feel of crushed leaves or dry, flaky soil. It is smooth to the touch but retains a slight granular quality, making it distinct from other minerals. When handled, asbestos feels cool and slightly rough, with a resistance to heat that makes it durable in high-temperature environments. Its texture is both soft and firm, creating a sense of stability while remaining lightweight. < | end_of_sentence | ></td></tr>
<tr><td colspan="2">Qwen3-8B-Reason.: After applying GRPO, the model places stronger emphasis on word count and sentence length, ensuring compliance with length constraints.</td></tr>
<tr><td><|im_start|> user
What is the meaning of PCF in the context of shadow maps ?
The output must follow the following rules :
1 . The answer must use capitalized letters for each word .
2 . The answer must contain at most 10 words .
3 . The answer must include at least 2 sentences . <|im_end|>
<|im_start|> assistant
<think>
...
</think>

PCF STANDS FOR PERCENTAGE-CLOSER FILTERING. IT AVERAGES SAMPLES TO REDUCE SHADOW ARTIFACTS. <|im_end|></td><td><|im_start|> user
What is the meaning of PCF in the context of shadow maps ?
The output must follow the following rules :
1 . The answer must use capitalized letters for each word .
2 . The answer must contain at most 10 words .
3 . The answer must include at least 2 sentences . <|im_end|>
<|im_start|> assistant
<think>
...
</think>

PCF STANDS FOR PERCENTAGE CLOSER FILTERING. IT REDUCES SHADOW ARTIFACTS. <|im_end|></td></tr>
</table>

Table 4: Visualization of the importance of each input token to the output. Darker colors indicate greater significance.

the importance of each input token with respect to the model's output and present representative visualizations in Table 4. After applying GRPO, the importance of constraint-related tokens increases, while the influence of irrelevant tokens diminishes; the relevance of core problem components remains largely unchanged. This suggests that the model has improved its ability to identify and prioritize constraint-related information without sacrificing its understanding of the overall input. Additionally, the reduced attention to irrelevant tokens may help minimize distraction from non-essential elements, which could explain the observed stability or gains in general task performance. These case studies further validate the effectiveness of our proposed framework and the utility of the data it produces.

6 Conclusion

In this paper, we propose a multi-dimensional constraint framework that categorizes instructions based on constraint pattern, constraint category, and constraint difficulty. Building on it, we design a controllable instruction generation pipeline that transforms raw instructions into constraint-based variants through the processes of constraint expansion, conflict detection, and instruction rewriting. Using this pipeline, we generate 9,106 instances. We then conduct a comprehensive evaluation of 18 LLMs from six model families and apply the GRPO algorithm to enhance LLMs' abilities. The results show that models trained with our data achieve significant improvements while preserving general capabilities. Parameter-level analysis and case studies suggest that these gains primarily arise from increased model sensitivity to constraint-relevant information.

Limitations

Although we evaluate and enhance the instruction-following ability of current LLMs using constructive data and analyze the underlying reasons for their improvement, our work still has two key limitations. On one hand, due to resource constraints, we assess model performance before and after RL only on a limited set of representative datasets, which may not fully capture changes in the models' original abilities. Nevertheless, our results show that performance on these widely used benchmarks remains largely stable, underscoring the robustness of our constructed data. On the other hand, since our focus is primarily on enhancing instruction-following abilities, we do not explore the effects of applying our method to domain-specific datasets. However, because our approach can convert any instruction into a constraint-based version, and case studies confirm that the model retains its focus on the core problem components, we believe that applying this method to other domains (e.g., reasoning, coding) can also yield additional performance gains.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 62476061, 62576106, 62376061).

References

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *CoRR*, abs/2108.07732.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Jiale Cheng, Xiao Liu, Cunxiang Wang, Xiaotao Gu, Yida Lu, Dan Zhang, Yuxiao Dong, Jie Tang, Hongning Wang, and Minlie Huang. 2024. Spar: Self-play with tree-search refinement to improve instruction-following in large language models. *CoRR*, abs/2412.11605.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, and 12 others. 2022. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416.

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

DeepSeek-AI. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 81 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Yang Deng, Xuan Zhang, Wenxuan Zhang, Yifei Yuan, See-Kiong Ng, and Tat-Seng Chua. 2024. On the multi-turn instruction following for conversational web agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 8795–8812. Association for Computational Linguistics.

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. *CoRR*, abs/2406.13542.

Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. 2025a. Self-play with execution feedback: Improving instruction-following capabilities of large language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Guanting Dong, Xiaoshuai Song, Yutao Zhu, Runqi Qiao, Zhicheng Dou, and Ji-Rong Wen. 2025b. Toward verifiable instruction-following alignment for retrieval augmented generation. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 23796–23804. AAAI Press.

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. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 10864–10882. Association for Computational Linguistics.

Qianyu He, Jie Zeng, Qianxi He, Jiaqing Liang, and Yanghua Xiao. 2024b. From complex to simple: Enhancing multi-constraint complex instruction following ability of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 10864–10882. Association for Computational Linguistics.

Yun He, Di Jin, Chaoqi Wang, Chloe Bi, Karishma Mandyam, Hejia Zhang, Chen Zhu, Ning Li, Tengyu Xu, Hongjiang Lv, Shruti Bhosale, Chenguang Zhu, Karthik Abinav Sankararaman, Eryk Helenowski, Melanie Kambadur, Aditya Tayade, Hao Ma, Han Fang, and Sinong Wang. 2024c. Multi-if: Benchmarking llms on multi-turn and multilingual instructions following. *CoRR*, abs/2410.15553.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Comput.*, 3(1):79–87.

Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. 2024. Followbench: A multi-level fine-grained constraints following benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 4667–4688. Association for Computational Linguistics.

Yimin Jing, Renren Jin, Jiahao Hu, Huishi Qiu, Xiaohua Wang, Peng Wang, and Deyi Xiong. 2023. Followeval: A multi-dimensional benchmark for assessing the instruction-following capability of large language models. *CoRR*, abs/2311.09829.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *CoRR*, abs/2001.08361.

Yizhi Li, Ge Zhang, Xingwei Qu, Jiali Li, Zhaoqun Li, Noah Wang, Hao Li, Ruibin Yuan, Yinghao Ma, Kai Zhang, Wangchunshu Zhou, Yiming Liang, Lei Zhang, Lei Ma, Jiajun Zhang, Zuowen Li, Wenhao Huang, Chenghua Lin, and Jie Fu. 2024. Cif-bench: A chinese instruction-following benchmark for evaluating the generalizability of large language models. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 12431–12446. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11048–11064. Association for Computational Linguistics.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex

Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Yunjia Qi, Hao Peng, Xiaozhi Wang, Amy Xin, Youfeng Liu, Bin Xu, Lei Hou, and Juanzi Li. 2025. AGENTIF: benchmarking instruction following of large language models in agentic scenarios. *CoRR*, abs/2505.16944.

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. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 13025–13048. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy P. Lillicrap, Jean-Baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, Ioannis Antonoglou, Rohan Anil, Sebastian Borgeaud, Andrew M. Dai, Katie Millican, Ethan Dyer, Mia Glaese, Thibault Sottiaux, Benjamin Lee, and 34 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *CoRR*, abs/2403.05530.

Elvis Saravia. 2022. Prompt Engineering Guide. *https://github.com/dair-ai/Prompt-Engineering-Guide*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

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. *CoRR*, abs/2404.02823.

Meta Team. 2024. Introducing llama 3.1: Our most capable models to date.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 9840–9855. Association for Computational Linguistics.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxing Xu, Yiming Liu, Jie Tang, Hongning Wang, and Minlie Huang. 2024. Benchmarking complex instruction-following with multiple constraints composition. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, and 10 others. 2023. The rise and potential of large language model based agents: A survey. *CoRR*, abs/2309.07864.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, Jie Zhou, Siming Chen, Tao Gui, Qi Zhang, and Xuanjing Huang. 2023. A comprehensive capability analysis of GPT-3 and GPT-3.5 series models. *CoRR*, abs/2303.10420.

Junjie Ye, Guanyu Li, Songyang Gao, Caishuang Huang, Yilong Wu, Sixian Li, Xiaoran Fan, Shihan Dou, Tao Ji, Qi Zhang, Tao Gui, and Xuanjing Huang. 2025. Tooleyes: Fine-grained evaluation for tool learning capabilities of large language models in real-world scenarios. In *Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025*, pages 156–187. Association for Computational Linguistics.

Junjie Ye, Yilong Wu, Sixian Li, Yuming Yang, Tao Gui, Qi Zhang, Xuanjing Huang, Peng Wang, Zhongchao Shi, Jianping Fan, and Zhengyin Du. 2024. Tl-training: A task-feature-based framework for training large language models in tool use. *CoRR*, abs/2412.15495.

Ming Zhang, Yujiong Shen, Jingyi Deng, Yuhui Wang, Yue Zhang, Junzhe Wang, Shichun Liu, Shihan Dou, Huayu Sha, Qiyuan Peng, Changhao Jiang, Jingqi Tong, Yilong Wu, Zhihao Zhang, Mingqi Wu, Zhiheng Xi, Mingxu Chai, Tao Liang, Zhihui Fei, and 6 others. 2025. Llmeval-3: A large-scale longitudinal study on robust and fair evaluation of large language models. *CoRR*, abs/2508.05452.

Xinghua Zhang, Haiyang Yu, Cheng Fu, Fei Huang, and Yongbin Li. 2024a. IOPO: empowering llms with complex instruction following via input-output preference optimization. *CoRR*, abs/2411.06208.

Yue Zhang, Ming Zhang, Haipeng Yuan, Shichun Liu, Yongyao Shi, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024b. Llmeval: A preliminary study on how to evaluate large language models. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 19615–19622. AAAI Press.

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

A Comparison of MulDimIF with Existing Approaches

To visually highlight the similarities and differences between our approach and existing methods, we provide a comprehensive comparison in Table 5. This comparison shows that MulDimIF offers several key advantages:

- **Unified generation of both training and testing data:** Whereas most prior methods focus on only one, our framework systematically constructs both in a principled and consistent way.
- **Broad coverage of constraint categories and patterns:** MulDimIF supports a richer and more comprehensive set of constraint dimensions.
- **Fully automated data construction:** This enables scalable data augmentation while minimizing manual effort.
- **Code-verifiable outputs:** The generated data can be validated automatically, strengthening the rigor and reliability of evaluation.

B Details about Manual Review

To ensure the quality of the constructed data, we carry out a thorough manual inspection process as outlined below. Six trained annotators (computer science graduate and undergraduate students) spend four weeks verifying and refining all 9,106 data instances along with their corresponding validation code.

Annotators first examine each generated instance to confirm that all specified constraints are correctly represented, mutually compatible, and expressed unambiguously. When issues are identified, the instance is revised and rechecked. Each item undergoes cross-validation by at least two annotators and is retained only when both agree that it is error-free.

After the data are finalized, we group the instances by constraint category. DeepSeek-V3 (DeepSeek-AI, 2024) then generates initial validation code for each group. Annotators review and refine this code to ensure logical correctness and full coverage of the data. A second annotator independently evaluates the revised code. Code is accepted only when both reviewers approve it; otherwise, it is returned for further revision.

These procedures are designed to maximize the quality of both the dataset and the validation code.

C Examples for Different Forms of Constraints

To illustrate the instructions generated by our method, we present a selection in Table 6, annotated with their corresponding dimension information. These examples demonstrate that our approach overcomes the templating limitations of prior methods and better captures the diversity of real-world needs. Additionally, the multi-dimensional categorization enables a more fine-grained performance analysis.

D Detailed Information of Models

We conduct an evaluation of 18 LLMs from seven families, including four open-source and three closed-source, that collectively represent the capabilities of current LLMs.

Method	# Train Data	# Test Data	Turn Category	Constraint Category	Automated Pipeline	Code Verifiable	Incorporation Pattern	Listing Pattern	Example Pattern
IFEval (Zhou et al., 2023)	0	541	Single	9	✓	✓	✓	×	×
Conifer (Sun et al., 2024)	13,606	0	Single	–	✓	×	✓	×	×
CI-DPO (He et al., 2024b)	2,781	0	Single	–	×	✓	✓	×	×
IOPO (Zhang et al., 2024a)	119,345	1,042	Single	4	✓	×	✓	×	×
FollowBench (Jiang et al., 2024)	0	820	Single	4	×	×	✓	×	✓
Multi-IF (He et al., 2024c)	0	13,447	Multi	9	✓	✓	✓	×	×
AutoIF (Dong et al., 2025a)	61,492	0	Single	–	✓	✓	✓	×	×
AGENTIF (Qi et al., 2025)	0	707	Multi	3	×	×	✓	×	✓
MulDimIF (Ours)	7,906	1,200	Single	4	✓	✓	✓	✓	✓

Table 5: Comparison of MulDimIF with existing approaches.

Open-Source LLMs We evaluate eleven open-source LLMs across four model families:

- **LLaMA3.1 family**, developed by Meta, comprises open-source models that demonstrate strong performance in general knowledge and multilingual translation. We evaluate *LLaMA3.1-Instruct-8B* and *LLaMA3.1-Instruct-70B*.
- **DeepSeek-R1-Distill-LLaMA family** consists of models distilled from the LLaMA family using data generated by DeepSeek-R1 (DeepSeek-AI et al., 2025). These models exhibit notable gains in mathematical performance. We evaluate *DeepSeek-R1-Distill-LLaMA-8B* and *DeepSeek-R1-Distill-LLaMA-70B*.
- **Qwen3 family**, released by Alibaba, is the latest version of the Qwen model family. It includes both dense and Mixture-of-Experts (MoE) (Jacobs et al., 1991) architectures, with parameter sizes ranging from 0.6 to 235 billion. In our study, we evaluate *Qwen3-8B*, *Qwen3-14B*, and *Qwen3-32B*. Since Qwen3 includes specialized optimizations for reasoning, we evaluate its models in both non-reasoning mode (i.e., *-Direct.*) and reasoning mode (i.e., *-Reason.*).

Closed-Source LLMs We evaluate eight closed-source LLMs across three model families:

- **Gemini1.5 family**, developed by Google, represents a new generation of models built on the MoE architecture. These models demonstrate enhanced performance, particularly in long-context understanding across multiple modalities. We evaluate *Gemini1.5-Flash* and *Gemini1.5-Pro*.
- **Claude3.5 family**, developed by Anthropic, features models with strong general-purpose capabilities and coding capabilities. We evaluate *Claude3.5-Haiku* and *Claude3.5-Sonnet*.
- **GPT family**, released by OpenAI, includes models that represent the current frontier in LLM development. We evaluate *GPT-3.5-Turbo*, *GPT-4-Turbo*, *GPT-4o-Mini*, and *GPT-4o*.

E Detailed Results

E.1 Details of Training Process

Table 7 and Table 8 present the model's critic scores and its performance on MulDimIF, respectively, as training progresses. The results show a clear pattern of gradual convergence over time. At the same time, the consistently upward trend further demonstrates steady performance improvements throughout training, underscoring the effectiveness of our data.

E.2 Details of Final Results

Table 9 to Table 14 present the detailed performance of the LLMs on each test set before and after GRPO. In these tables, Δ denotes the performance difference between the post-GRPO and pre-GRPO models, with positive values highlighted in green and negative values in red. The results demonstrate that applying our data for GRPO substantially enhances the models' capabilities across all aspects, highlighting the effectiveness of our data.

F Prompts for Instruction Generation

Our automated instruction generation pipeline includes constraint expansion, conflict detection, and instruction rewriting, with several steps utilizing GPT-4o. The corresponding prompts are listed from Figure 5 to Figure 8. Some of the information used in the prompts is presented in Table 15 and Table 16.

Instruction	Pattern	Category	Difficulty
# Example 1: **Question**: What Is The Room? The answer must use capitalized letters for each word. **Answer**: The Room Is A Living Room. # Example 2: **Question**: Describe life in 1980. The answer must use capitalized letters for each word. **Answer**: Life In 1980 Was Characterized By The Rise Of Personal Computers, Vibrant Music Scenes With Genres Like Disco And Rock, And Fashion Trends Featuring Bold Colors And Styles. People Enjoyed Watching Movies On VHS Tapes And Listening To Music On Cassette Players. # Example 3: **Question**: Can I insert PDFs or documents? The answer must use capitalized letters for each word. **Answer**: Yes, You Can Insert PDFs Or Documents. **Question**: Write a script about Matt Rhule's 0-3 record so far. The answer must use capitalized letters for each word.	Example	Language	Level I
# Example 1: **Question**: What are the best books to learn Cybersecurity in 2023? The answer must use capitalized letters for each word and must include the keyword 'Cybersecurity'. **Answer**: "Cybersecurity For Beginners" And "The Art Of Invisibility" Are Among The Best Books To Learn Cybersecurity In 2023. # Example 2: **Question**: Hi chat GPT4, how many questions can I ask from you when I have a free account here? The answer must be in all uppercase letters and must include the keyword 'LIMIT'. **Answer**: THE LIMIT IS 50 QUESTIONS PER MONTH. # Example 3: **Question**: I've read about a female wrestler using a move called "Kegel Crush". What kind of move could this be? It sounds like a creative name for a submission move. The answer must use capitalized letters for each word and must include the keyword 'submission'. **Answer**: The Kegel Crush Could Be A Unique Submission Move Where The Wrestler Applies Pressure To The Opponent's Core Muscles, Forcing A Submission Through Intense Abdominal Constriction. **Question**: I want you to create a detailed curriculum for mastering each of the skills listed below. Divide each skill into multiple sub-topics and list the skills required for each sub-topic and suggest the best online courses and books for each sub-topic. Make sure it should be free. Additionally, the answer must use capitalized letters for each word and must include the keyword 'Curriculum Design'.	Example	Content Language	Level II

Instruction	Pattern	Category	Difficulty
# Example 1: **Question**: What are the best books to learn Java in 2023? Please ensure that the answer uses capitalized letters for each word, is at most 50 words long, and includes the keyword ‘Java Programming’. **Answer**: Effective Java By Joshua Bloch, Java: The Complete Reference By Herbert Schildt, And Head First Java By Kathy Sierra Are Excellent Books For Learning Java Programming In 2023. # Example 2: **Question**: What is the ice level in Plants vs Zombies mini-games? The answer must use capitalized letters for each word, be at most 20 words, and include the keyword ‘Plants vs Zombies’. **Answer**: The Ice Level In Plants Vs Zombies Mini-Games Is Called Icebound Battleground. # Example 3: **Question**: Please provide a detailed explanation of performance evaluation by competencies and the S20/23 evaluation cited in APA standards that is specific to Ecuador. The response must be at least 150 words to ensure a comprehensive understanding of the topic. Additionally, the answer must use capitalized letters for each word. Furthermore, the explanation must include the keyword ‘Ecuadorian Educational Standards’ to ensure relevance to the local context. **Answer**: Performance Evaluation By Competencies In Ecuador Focuses On Assessing Individuals Based On Specific Skills And Abilities Relevant To Their Roles. This Approach Aligns With The Ecuadorian Educational Standards, Which Emphasize Competency-Based Learning To Ensure Students Acquire Practical Skills. The S20/23 Evaluation, Cited In APA Standards, Is A Framework Used To Measure These Competencies Through A Structured Assessment Process. In Ecuador, This Evaluation Is Adapted To Reflect Local Educational Goals And Cultural Contexts, Ensuring That Students Meet The National Standards. The Ecuadorian Educational Standards Play A Crucial Role In Shaping The Criteria For These Evaluations, Ensuring That They Are Relevant And Comprehensive. By Focusing On Competencies, The Evaluation Aims To Provide A Holistic View Of A Student’s Abilities, Preparing Them For Future Challenges. This Methodology Is Integral To The Educational System In Ecuador, As It Promotes Continuous Improvement And Alignment With Global Educational Practices. **Question**: Can you provide a list of airlines that have branded fares? Please ensure that the answer is at most 50 words long, making it concise and to the point. Additionally, the answer must use capitalized letters for each word. The answer must also include the keyword ‘branded fares.’	Example	Content Language Length	Level III

Instruction	Pattern	Category	Difficulty
# Example 1: **Question**: Provide a definition for the name “Patrick.” The answer should be concise, using at most 50 words, and must use capitalized letters for each word. Additionally, the answer must include the keyword ‘Saint’ to reflect the historical and cultural significance associated with the name. Furthermore, the answer must be formatted with a level 2 heading in Markdown. **Answer**: ## Patrick: A Name Historically Linked To Saint Patrick, The Patron Saint Of Ireland, Known For Spreading Christianity And Celebrated On Saint Patrick’s Day. # Example 2: **Question**: What type of energy do herbivores and omnivores gain when they eat plants? The answer must be at most 5 words long, use capitalized letters for each word, include the keyword ‘Energy’, and be formatted as a level 2 heading in Markdown. **Answer**: ## Chemical Energy From Plants # Example 3: **Question**: What is choline? Please provide your answer in at most 3 sentences, and ensure that each word in your response is capitalized. Additionally, the answer must include the keyword ‘nutrient’. Furthermore, the answer must include a level 2 heading formatted in Markdown. **Answer**: ## What Is Choline? Choline Is An Essential Nutrient That Supports Various Bodily Functions, Including Brain Development And Liver Function. It Plays A Crucial Role In The Synthesis Of Phospholipids, Which Are Vital Components Of Cell Membranes. **Question**: Format your response using markdown, ensuring the use of at least two heading levels, subheadings, bullet points, and bold to organize the information. List some conjugations for “to read” in Mandarin Chinese. The response must be concise, with a maximum of 150 words, and must include the conjugations in Traditional Chinese characters. Additionally, ensure that the response ends with a period.	Example	Content Format Language Length	Level IV

Instruction	Pattern	Category	Difficulty
Which 10 countries have the highest number of golf courses relative to their population size? The output must follow the following rules: 1. Use at most 50 words. 2. Provide the information in at least 3 sentences.	Listing	Length	Level I
Can you provide the schedule for DEF CON 11? The output must follow the following rules: 1. The answer must contain at most 50 words. 2. The answer must be presented in a table format with a maximum of 5 rows.	Listing	Format Length	Level II
Could you provide information on some tourist attractions in Norway and suggest the duration of visits for each? The output must follow the following rules: 1. The answer must use at least two heading levels to organize the information. 2. Any table included in the answer must not exceed five rows. 3. The answer must be between 200 and 300 words. 4. The answer must include at least three paragraphs. 5. The answer must use capitalized letters for each word.	Listing	Format Language Length	Level III
Can The Window Size Of A Microsoft Form Be Adjusted To Fit A Mobile Screen? The output must follow the following rules: 1. The answer must use capitalized letters for each word. 2. The answer must be no more than two sentences. 3. The answer must include the keyword 'responsive design.' 4. The answer must be formatted using a level 2 heading in Markdown.	Listing	Content Format Language Length	Level IV
Where can I find GPT-4 for free? When answering, it is essential to include the keyword 'GPT-4 access' as part of the response.	Incorporation	Content	Level I
How can I reverse engineer Geometry Dash? When answering, it is essential to include a JSON object that is structured with at least three levels of nesting. This requirement ensures that the reverse engineering process is detailed and comprehensive. Additionally, the explanation must incorporate the keyword 'decompilation' to emphasize this critical aspect of the reverse engineering process.	Incorporation	Content Format	Level II
Please list all Roman Emperors by length of reign, ensuring that the context is clear by including the keyword 'Roman Empire' in your answer. The response should be in English, with the first letter of each word capitalized. Additionally, present the information in a table format, adhering to a column limit of 3. The table should include columns for 'Emperor Name', 'Length of Reign', and 'Additional Information'.	Incorporation	Content Format Language	Level III
Who Was Ida B. Wells? In Your Response, It Is Important To Use Block Quotes To Highlight Key Quotes Or Important Points About Her Life And Contributions, As This Will Help Emphasize Her Impact. Additionally, Ensure That Your Response Contains At Least 150 Words To Provide A Comprehensive Overview Of Her Achievements And Influence. The Response Must Also Include The Keyword 'Civil Rights' To Highlight Her Significant Role In The Movement. Furthermore, The Response Should Use Capitalized Letters For Each Word To Maintain A Consistent Style Throughout The Text.	Incorporation	Content Format Language Length	Level IV

Table 6: Examples of instructions generated by the proposed approach.

Family	Version	S. 35	S. 70	S. 105	S. 140	S. 175	S. 205	S. 245
LLaMA3.1	Instruct-8B	4.1426	4.2842	4.4463	4.4316	4.4453	4.0215	4.2656
Qwen3	8B-Direct.	4.0088	4.0488	4.3008	4.3770	4.3779	3.9727	4.2168
	14B-Direct.	4.2197	4.1475	4.4033	4.4258	4.4268	3.9404	4.2031
DeepSeek-R1-Distill-LLaMA	Instruct-8B	3.7061	3.9570	4.2100	4.1670	4.3252	3.8203	4.2148
Qwen3	8B-Reason.	4.0107	4.0322	4.2236	4.2910	4.3594	3.9229	4.1992
	14B-Reason.	4.1094	4.1025	4.3398	4.4453	4.4033	3.3590	4.1816

Table 7: Critic score at different training steps.

Family	Version	S. 35	S. 70	S. 105	S. 140	S. 175	S. 205	S. 245
LLaMA3.1	Instruct-8B	77.67	83.42	86.92	88.17	87.17	89.17	89.96
Qwen3	8B-Direct.	71.33	76.42	80.58	83.25	85.00	86.50	87.42
	14B-Direct.	72.91	77.92	81.83	82.17	84.33	85.42	85.58
DeepSeek-R1-Distill-LLaMA	Instruct-8B	62.58	68.92	74.50	77.50	78.25	80.25	81.83
Qwen3	8B-Reason.	71.42	76.42	78.83	80.67	82.08	83.67	83.83
	14B-Reason.	74.58	78.33	79.83	82.42	83.92	84.83	86.17

Table 8: Performance on MulDimIF at different training steps.

Models	Constraint Pattern			Constraint Category				Constraint Difficulty				Overall
	Example	Listing	Incorporation	Content	Format	Language	Length	Level I	Level II	Level III	Level IV	
LLaMA3.1-Instruct-8B												
w/o GRPO	40.25	36.00	32.25	80.13	64.74	44.28	49.60	64.67	40.67	27.33	12.00	36.17
w/ GRPO	89.00	91.00	84.25	95.23	92.20	98.70	93.38	94.33	88.67	85.33	84.00	88.08
Δ	48.75	55.00	52.00	15.10	27.46	54.42	43.78	29.66	48.00	58.00	72.00	51.91
Qwen3-8B-Direct.												
w/o GRPO	63.00	55.00	51.00	84.03	72.27	83.00	69.89	87.67	60.33	45.67	31.00	56.17
w/ GRPO	90.50	82.75	87.50	94.35	91.94	98.26	92.21	94.33	87.33	84.00	82.00	86.92
Δ	27.50	27.75	36.50	10.32	19.67	15.26	22.32	6.66	27.00	38.33	51.00	30.75
Qwen3-14B-Direct.												
w/o GRPO	64.00	63.00	52.00	87.92	77.00	89.87	65.86	88.00	66.67	50.67	33.67	60.00
w/ GRPO	89.50	82.75	87.00	94.73	89.25	97.97	94.29	94.00	87.67	83.00	81.00	86.42
Δ	25.50	19.75	35.00	6.81	12.25	8.10	28.43	6.00	21.00	32.33	47.33	26.42
DeepSeek-R1-Distill-LLaMA-Instruct-8B												
w/o GRPO	42.00	28.00	28.00	80.00	50.44	44.72	53.09	59.67	40.33	18.33	12.33	32.67
w/ GRPO	83.75	85.50	77.00	93.48	83.60	98.55	92.60	91.33	85.00	78.00	74.00	82.08
Δ	41.75	57.50	49.00	13.48	33.16	53.83	39.51	31.66	44.67	59.67	61.67	49.41
Qwen3-8B-Reason.												
w/o GRPO	67.75	61.25	54.00	85.84	71.02	87.70	72.58	87.00	68.33	52.67	36.00	61.00
w/ GRPO	87.50	80.00	83.50	94.86	86.69	97.25	92.08	93.33	84.67	80.67	76.00	83.67
Δ	19.75	18.75	29.50	9.02	15.67	9.55	19.50	6.33	16.34	28.00	40.00	22.67
Qwen3-14B-Reason.												
w/o GRPO	71.50	64.00	57.75	87.79	77.29	92.47	72.31	86.00	72.33	58.67	40.67	64.42
w/ GRPO	88.50	82.00	86.75	94.86	89.52	98.12	92.60	95.00	86.67	82.67	78.67	85.75
Δ	17.00	18.00	29.00	7.07	12.23	5.65	20.29	9.00	14.34	24.00	38.00	21.33

Table 9: Evaluation results on our custom test set.

Models	Prompt Str.	Instruction Str.	Prompt Loo.	Instruction Loo.
LLaMA3.1-Instruct-8B				
w/o GRPO	70.79	79.02	74.49	82.49
w/ GRPO	79.11	85.73	80.22	86.57
Δ	8.32	6.71	5.73	4.08
Qwen3-8B-Direct.				
w/o GRPO	82.44	88.25	85.95	90.65
w/ GRPO	86.69	91.13	89.09	92.69
Δ	4.25	2.88	3.14	2.04
Qwen3-14B-Direct.				
w/o GRPO	86.14	90.77	89.09	92.69
w/ GRPO	87.06	91.49	89.09	92.81
Δ	0.92	0.72	0.00	0.12
DeepSeek-R1-Distill-LLaMA-Instruct-8B				
w/o GRPO	61.55	71.82	67.65	76.50
w/ GRPO	79.30	85.13	82.07	87.41
Δ	17.75	13.31	14.42	10.91
Qwen3-8B-Reason.				
w/o GRPO	84.10	88.73	86.88	90.53
w/ GRPO	87.43	91.49	89.46	92.81
Δ	3.33	2.76	2.58	2.28
Qwen3-14B-Reason.				
w/o GRPO	86.88	91.01	89.65	93.05
w/ GRPO	86.88	91.25	89.83	93.17
Δ	0.00	0.24	0.18	0.12

Table 10: Evaluation results on IFEval.

Models	Italian	Spanish	Hindi	Portuguese	English	French	Chinese	Russian	Avg.
LLaMA3.1-Instruct-8B									
w/o GRPO	68.30	72.70	58.50	69.50	79.29	67.83	62.38	60.52	68.60
w/ GRPO	83.53	83.58	69.87	81.65	81.99	77.39	75.04	71.22	78.41
Δ	15.23	10.88	11.37	12.15	2.70	9.56	12.66	10.70	9.81
Qwen3-8B-Direct.									
w/o GRPO	86.58	89.46	78.20	86.16	87.25	85.54	83.62	82.23	85.12
w/ GRPO	90.27	91.03	82.44	91.23	91.65	88.81	87.49	85.57	88.84
Δ	3.69	1.57	4.24	5.07	4.40	3.27	3.87	3.34	3.72
Qwen3-14B-Direct.									
w/o GRPO	86.92	90.59	82.38	86.79	89.95	89.29	85.47	82.40	87.05
w/ GRPO	89.65	90.54	85.71	87.18	90.02	88.33	86.71	85.34	88.14
Δ	2.73	-0.05	3.33	0.39	0.07	-0.96	1.24	2.94	1.09
DeepSeek-R1-Distill-LLaMA-Instruct-8B									
w/o GRPO	53.82	56.23	43.15	53.22	60.44	51.44	64.01	49.24	54.37
w/ GRPO	76.08	72.62	50.86	75.44	79.92	67.61	70.49	61.55	70.21
Δ	22.26	16.39	7.71	22.22	19.48	16.17	6.48	12.31	15.84
Qwen3-8B-Reason.									
w/o GRPO	89.24	90.46	79.14	88.52	89.82	87.77	83.36	82.10	86.65
w/ GRPO	91.79	93.04	83.32	91.48	90.45	90.15	86.72	85.33	89.19
Δ	2.55	2.58	4.18	2.96	0.63	2.38	3.36	3.23	2.54
Qwen3-14B-Reason.									
w/o GRPO	91.10	91.29	82.44	90.58	89.88	92.05	85.17	83.98	88.48
w/ GRPO	91.51	92.45	86.29	90.48	90.12	89.20	87.71	87.81	89.51
Δ	0.41	1.16	3.85	-0.10	0.24	-2.85	2.54	3.83	1.03

Table 11: Evaluation results on Multi-IF for turn 1.

Models	Italian	Spanish	Hindi	Portuguese	English	French	Chinese	Russian	Avg.
LLaMA3.1-Instruct-8B									
w/o GRPO	59.62	64.31	50.71	62.76	71.64	62.92	54.81	41.97	59.82
w/ GRPO	70.76	73.49	55.87	67.89	72.79	69.49	66.35	54.57	66.94
Δ	11.14	9.18	5.16	5.13	1.15	6.57	11.54	12.60	7.12
Qwen3-8B-Direct.									
w/o GRPO	81.18	80.85	68.46	77.95	80.92	79.71	75.22	69.06	77.05
w/ GRPO	81.21	80.42	73.17	79.97	81.09	81.72	76.49	70.21	78.33
Δ	0.03	-0.43	4.71	2.02	0.17	2.01	1.27	1.15	1.28
Qwen3-14B-Direct.									
w/o GRPO	80.76	83.57	74.22	81.07	84.28	84.56	77.67	70.44	80.03
w/ GRPO	82.50	81.32	76.64	79.05	83.65	82.97	77.58	70.56	79.70
Δ	1.74	-2.25	2.42	-2.02	-0.63	-1.59	-0.09	0.12	-0.33
DeepSeek-R1-Distill-LLaMA-Instruct-8B									
w/o GRPO	44.92	45.01	28.29	44.90	55.15	41.90	54.69	30.67	44.04
w/ GRPO	63.35	63.59	43.48	63.13	69.34	59.72	61.08	38.14	58.66
Δ	18.43	18.58	15.19	18.23	14.19	17.82	6.39	7.47	14.62
Qwen3-8B-Reason.									
w/o GRPO	83.12	80.58	72.67	81.32	82.40	83.87	74.78	70.52	79.06
w/ GRPO	85.65	84.51	75.55	83.86	84.35	85.57	79.67	74.30	81.96
Δ	2.53	3.93	2.88	2.54	1.95	1.70	4.89	3.78	2.90
Qwen3-14B-Reason.									
w/o GRPO	84.18	84.53	74.87	83.80	84.67	86.31	76.88	71.80	81.27
w/ GRPO	85.11	84.99	77.86	83.96	84.47	85.49	80.27	74.92	82.36
Δ	0.93	0.46	2.99	0.16	-0.20	-0.82	3.39	3.12	1.09

Table 12: Evaluation results on Multi-IF for turn 2.

Models	Italian	Spanish	Hindi	Portuguese	English	French	Chinese	Russian	Avg.
LLaMA3.1-Instruct-8B									
w/o GRPO	51.27	54.55	41.78	54.54	63.75	54.91	45.53	35.31	51.46
w/ GRPO	59.08	61.03	46.79	59.75	64.72	58.53	53.71	41.47	56.43
Δ	7.81	6.48	5.01	5.21	0.97	3.62	8.18	6.16	4.97
Qwen3-8B-Direct.									
w/o GRPO	74.08	71.17	58.29	71.43	73.19	72.25	67.99	58.40	68.73
w/ GRPO	73.33	71.77	66.57	73.31	74.42	72.57	69.43	59.50	70.49
Δ	-0.75	0.60	8.28	1.88	1.23	0.32	1.44	1.10	1.76
Qwen3-14B-Direct.									
w/o GRPO	76.50	76.74	65.06	75.23	78.20	77.76	71.56	60.14	73.14
w/ GRPO	76.41	74.51	70.33	73.73	78.62	76.21	71.29	58.97	73.05
Δ	-0.09	-2.23	5.27	-1.50	0.42	-1.55	-0.27	-1.17	-0.09
DeepSeek-R1-Distill-LLaMA-Instruct-8B									
w/o GRPO	34.95	32.51	19.85	35.67	41.91	32.33	44.42	23.06	33.67
w/ GRPO	50.37	48.43	29.32	52.51	58.75	46.58	50.12	31.16	46.94
Δ	15.42	15.92	9.47	16.84	16.84	14.25	5.70	8.10	13.27
Qwen3-8B-Reason.									
w/o GRPO	73.32	71.32	62.23	73.72	75.11	75.90	68.62	61.55	70.66
w/ GRPO	78.58	75.49	67.53	77.10	78.11	77.21	73.47	64.71	74.35
Δ	5.26	4.17	5.30	3.38	3.00	1.31	4.85	3.16	3.69
Qwen3-14B-Reason.									
w/o GRPO	77.45	77.08	66.47	77.39	78.01	79.59	70.54	62.54	74.04
w/ GRPO	80.78	76.69	70.47	79.33	79.59	78.80	73.23	64.53	75.80
Δ	3.33	-0.39	4.00	1.94	1.58	-0.79	2.69	1.99	1.76

Table 13: Evaluation results on Multi-IF for turn 3.

Models	MMLU	GSM8K	MATH	HumanEval	MBPP
LLaMA-3.1-8B-Instruct					
w/o GRPO	71.40	80.44	48.94	70.73	69.65
w/ GRPO	71.72	81.12	47.74	70.12	67.70
Δ	0.32	0.68	-1.20	-0.61	-1.95
Qwen3-8B-Direct.					
w/o GRPO	71.68	83.70	82.06	85.37	89.11
w/ GRPO	79.39	86.81	82.24	87.80	78.99
Δ	7.71	3.11	0.18	2.43	-10.12
Qwen3-14B-Direct.					
w/o GRPO	82.05	85.52	84.24	91.46	83.66
w/ GRPO	83.36	84.31	85.10	89.63	82.10
Δ	1.31	-1.21	0.86	-1.83	-1.56
DeepSeek-R1-Distill-Llama-8B					
w/o GRPO	67.37	84.61	82.30	77.44	73.93
w/ GRPO	69.39	85.82	84.54	84.15	80.93
Δ	2.02	1.21	2.24	6.71	6.99
Qwen3-8B-Reason.					
w/o GRPO	84.46	92.27	85.08	72.56	82.49
w/ GRPO	85.28	93.48	83.18	80.49	78.60
Δ	0.82	1.21	-1.90	7.93	-3.89
Qwen3-14B-Reason.					
w/o GRPO	87.10	92.87	84.92	76.22	88.72
w/ GRPO	87.05	93.33	86.44	79.27	93.39
Δ	-0.05	0.46	1.52	3.05	4.67

Table 14: Evaluation results on general domains.

Prompt for Constraint Expand

```
You are an expert in instruction-following data construction. Your task is to
↪  generate corresponding data as required.

You must carefully analyze and select specific constraints from the [New
↪  Constraint List]. Then, based on the original question in the provided
↪  [Data], generate new data that adheres to the [Data Generation Requirements].
↪  Finally, respond in the format specified in the [Response Format].

[New Constraint List]: {new_constrint_list}

[Data Generation Requirements]:

    [Core Requirements]:

        1. Ensure only {c1} added, that is, {c2}. The word following [Main
        ↪  Category] should be the main category.

        2. Based on this analysis, select {c3} from the [New Constraint List]
        ↪  and construct an appropriate "Specific Constraint Content". Add it
        ↪  to the [Original Constraint List] in the provided data, and return
        ↪  the [Updated Constraint List].

        3. Modify the content of the [Original Question] in the provided data to
        ↪  **explicitly and clearly specify all the constraints** in the
        ↪  [Updated Constraint List]. The modified question must clearly
        ↪  describe each constraint in natural language, ensuring that the
        ↪  constraints are fully integrated into the question text. For
        ↪  example:
           - Original Question: "Tell me about machine learning."
          - Constraint: "The answer must use capitalized letters for each word."
           - Modified Question: "Tell me about machine learning. The answer must
            ↪  use capitalized letters for each word."

        4. Ensure that the Specific Constraint in each constraint triplet is
        ↪  detailed and specific, containing concrete information or examples
        ↪  (e.g., instead of "Must include", specify "Must include the keyword
        ↪  'machine learning'").

    [Notes]:

        1. The new constraint cannot conflict with the constraints in the
        ↪  [Original Constraint List].

        2. The modified [Question with the New Constraint] must **explicitly
        ↪  describe all the constraints** in natural language, ensuring that
        ↪  the constraints are fully integrated into the question text.
        ↪  Constraints should not be implicitly applied to the answer without
        ↪  being explicitly stated in the question.
```

```
        3. Make sure the Specific Constraint in each constraint triplet is as
        ↪  specific as possible, including concrete details or examples.

        4. **Important**: The response must strictly follow the [Response
        ↪  Format] exactly as specified. Do not include any numbering, bullet
        ↪  points, or additional formatting. The [Updated Constraint List] must
        ↪  be outputted as a single list of tuples in the exact format shown,
        ↪  without any additional characters or line breaks between the tuples.

        5. When generating the modified [Question with the New Constraint],
        ↪  ensure that the language is natural and well-polished. Enrich the
        ↪  phrasing of constraints to avoid redundancy and monotony.

[Response Format]:

    [Thinking Process]: xxx

    [Updated Constraint List]: [(Main Category, Subcategory, Specific
    ↪  Constraint), (Main Category, Subcategory, Specific Constraint), ...]
    ↪  (The main category is the word after [Main Category], and the
    ↪  constraints we provide are just broad scopes. You need to find suitable
    ↪  specific constraints based on the question and its answers. The
    ↪  Specific Constraint should be detailed and specific.)

    [Question with the New Constraint]: xxx

[Data]:

    [Original Constraint List]: [{original_constraint_list}]

    [Original Question]: {original_question}
```

Figure 5: The prompt used for constraint expansion.

Prompt for Conflict Detection

```
You are an expert in data structure following instructions. You need to perform
↪  a series of checks on the given [Data] according to the [Check Requirements]
↪  and finally respond in the format specified in the [Response Format].

[Check Requirements]:
    1. Check if there is any constraint conflict in the "Constraint List" in the
    ↪  provided data. Explain first and then conclude.
    2. Check if the "Question" in the provided data clearly specifies all the
    ↪  constraint requirements in the "Constraint List". Explain first and
    ↪  then conclude.
    3. The response format should follow the requirements specified in the
    ↪  [Response Format] below.

[Response Format]:
```

```
    # Constraint Conflict Check #
    [Specific Explanation]:
    [Is there any constraint conflict in the constraints of the data]: [Yes/No]

    # Does the Question clearly specify all constraints in the Constraint List
    ↪  Check #
    [Specific Explanation]: [Explanation]
    [Does the question include all constraints from the constraint list]:
    ↪  [Yes/No]

[Data]:
    [Constraint List]: [{constraint_list}]
    [Question]: {quetsion}
```

Figure 6: The prompt used for conflict detection.

Prompt for Instruction Rewriting (Listing)

```
You are an expert in constructing data based on instructions. You need to
↪  generate the corresponding data as required.
You should modify the given [Original Question] according to the [Core
↪  Requirements] without changing the original meaning of the question. Then,
↪  respond in the format specified in the [Reply Format].

[Core Requirements]:
    1. Fully understand the [Original Question] and the constraints listed in
    ↪  the [Constraint List].
    2. Change the expression of the [Original Question]. First, extract the core
    ↪  question from the [Original Question] that is not bound by constraints,
    ↪  then list the constraints corresponding to the [Constraint List] at the
    ↪  end of the sentence. Start with "The output must follow the following
    ↪  rules:" and list the constraints from the [Original Question] clearly
    ↪  after understanding the constraints.
    3. The modified question must remain consistent with the [Original Question]
    ↪  in terms of meaning and constraints.

[Reply Format]:
    [Constraint List Data]: Core question (does not include constraint
    ↪  descriptions in the constraint list),
    The output must follow the following rules:
    1.xxx 2.xxx

[Data]:
    [Original Question]:{original_question}
    [Constraint List]:{constraint_list}
```

Figure 7: The prompt used for instruction rewriting (Listing).

Prompt for Instruction Rewriting (Incorporation)

```
You are an expert in data construction based on instructions. You need to
↪  generate the corresponding data as required.
You should modify the given [Data] according to the [Core Requirements] without
↪  changing the original meaning of the question. Then, respond in the format
↪  specified in the [Reply Format].

[Core Requirements]:
    1. Do not alter the question to directly satisfy the constraints.
    2. Fully understand the [Original Question] and the constraints within it.
    3. Modify the expression of the constraints in the [Original Question] by
    ↪  clearly describing them in the question, so that the question
    ↪  explicitly indicates the constraints, without changing its structure to
    ↪  meet those constraints directly.
    4. The modified question should keep the original meaning and intent, while
    ↪  the constraints are introduced as descriptive explanations or
    ↪  clarifications in the question.
    5. Ensure that the constraints are explicitly described in the question,
    ↪  making it clear that they need to be considered when answering, without
    ↪  altering the question to directly satisfy them.

[Reply Format]:
    [Constraint Integration Format Data]: xxx

[Data]:
    [Original Question]:{original_question}
    [Constraint List]:{constraint_list}
'''
```

Figure 8: The prompt used for instruction rewriting (Incorporation).

Category	**New Constraint List**

Content	Main Category : Content
	Subcategory : {
	Keywords: Must include, Repeated, Avoid
	Identifiers: Start identifier, End identifier, Delimiting identifier
	Punctuation: Ending punctuation, Exclude punctuation
	}

Format	Main Category : Format
	Subcategory : {
	Markdown: Heading levels, Block quotes
	Json: Object nesting levels
	XML: Number of attributes
	Table: Row limit, Column limit
	}

Language	Main Category : Language
	Subcategory : {
	Chinese: Simpfied, Traditional
	English: All Uppercase, Capitalized, Lowercase
	}

Length	Main Category : Length
	Subcategory : {
	Words: At most, At least, Range
	Sentences: At most, At least, Range
	Paragraphs: At most, At least, Range
	}

Table 15: New constraint list for different constraint categories.

c1	**c2**	**c3**
one new constraint is	a single (Primary category, secondary category, specific constraint) triplet	one constraint
two new constraints are	two (Primary category, secondary category, specific constraint) triplets	two constraints

Table 16: ‘c1,’ ‘c2,’ and ‘c3’ for the prompt of constraint expansion.