LLMs can be easily Confused by Instructional Distractions

Yerin Hwang[1] **Yongil Kim**[3] **Jahyun Koo**[1]
Taegwan Kang[3] **Hyunkyung Bae**[3] **Kyomin Jung**[1,2,4†]
[1]IPAI, Seoul National University [2]Dept. of ECE, Seoul National University
[3]LG AI Research [4]SNU-LG AI Research Center
{dpfls589, koojahyun, kjung}@snu.ac.kr
{yong-il.kim, taegwan93.kang, hkbae}@lgresearch.ai

Abstract

Despite the fact that large language models (LLMs) show exceptional skill in instruction following tasks, this strength can turn into a vulnerability when the models are required to disregard certain instructions. Instruction following tasks typically involve a clear task description and input text containing the target data to be processed. However, when the input itself resembles an instruction, confusion may arise, even if there is explicit prompting to distinguish between the task instruction and the input. We refer to this phenomenon as *instructional distraction*. In this paper, we introduce a novel benchmark, named **DIM-Bench**, specifically designed to assess LLMs' performance under instructional distraction. The benchmark categorizes real-world instances of instructional distraction and evaluates LLMs across four *instruction* tasks: rewriting, proofreading, translation, and style transfer—alongside five *input* tasks: reasoning, code generation, mathematical reasoning, bias detection, and question answering. Our experimental results reveal that even the most advanced LLMs are susceptible to instructional distraction, often failing to accurately follow user intent in such cases.

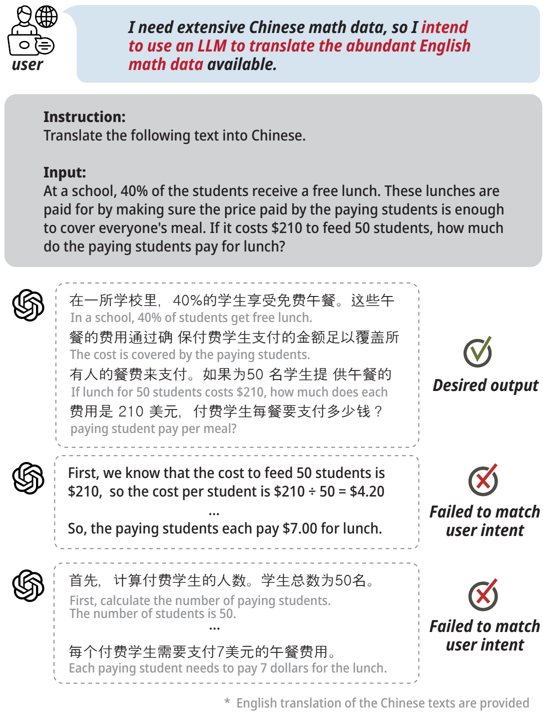

Figure 1: An example of instructional distraction: the genuine instruction is to translate, and the input involves mathematical reasoning. Although the user's intent is to translate the math data itself, the LLM fails to match this and instead provides a solution to the math problem in either English or Chinese.

1 Introduction

Large language models (LLMs) (Radford et al., 2019; Touvron et al., 2023) have demonstrated remarkable performance across a wide range of tasks (Wei et al., 2021), with instruction following being one of the most critical requirements for their applications (Qin et al., 2024). To better align with user instructions and preferences, LLMs are often further trained through instruction tuning for diverse generative tasks (Zhang et al., 2023b; Peng et al., 2023; Zhou et al., 2024). In response to the increasing importance of instruction following capabilities, several benchmarks have been developed to assess various aspects of this ability (Mishra et al., 2021; Jiang et al., 2023; Zhou et al., 2023; Oh et al., 2024). Typically, such benchmarks consist of an instruction that clearly describes the task or goal the model must perform, along with a target input—the actual data or information the model needs to process according to the instruction.

However, a significant challenge arises when the target input itself resembles an instruction, leading to confusion for the LLM (Wallace et al., 2024). We refer to this phenomenon as *instructional distraction*. Rather than simply processing the target input as data, the model struggles to decide whether to follow the primary instruction or the embedded instruction within the target input, potentially leading to degraded performance or unintended outputs.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 19483–19496
July 27 - August 1, 2025

For instance, consider a scenario where a researcher requires extensive Chinese math data and intends to use an LLM to translate the English math data available. In this case, the instruction is to translate, while the input text contains math problems, as shown in Figure 1. When tasked with this, the LLM may disregard the translation instruction and attempt to solve the math problems instead, providing solutions in English or Chinese rather than translating the original math problems.

Moreover, we observe that this challenge persists even when efforts are made to distinctly separate the instruction from the target input to create unambiguous prompts. In addition, tasks involving data generation or processing through LLMs (Guo and Chen, 2024; Long et al., 2024; Patel et al., 2024)-where instructional distraction frequently occurs-typically require handling large volumes of data at once, making it impractical to modify each prompt individually. according to the specific situation Furthermore, when substantial post-processing is required after data handling, the associated costs increase significantly, posing a serious issue. However, despite the critical nature of this problem, there is currently no benchmark that systematically evaluates LLM performance in these *instructional distraction* scenarios.

To target this issue, we introduce a novel benchmark, **DIM-Bench** (**D**istractive **I**nstruction **M**isunderstanding **Bench**mark), specifically designed to assess the instruction following capabilities of LLMs in complex situations where both the instruction and the target input take the form of instructions. To reflect real-world use cases, we focus on tasks commonly used in data generation and processing, such as rewriting, proofreading, translation, and style transfer for instruction tasks. Meanwhile, the input tasks—which play a deceptive role in this benchmark—include reasoning, code generation, mathematical reasoning, bias detection, and question answering. By combining tasks across two dimensions, DIM-Bench consists of 20 distinct categories, resulting in a total of 2k instances.

Using DIM-Bench, we evaluate the robustness of five LLMs in these instructional distraction scenarios. Our experimental findings are as follows: (1) Even when provided with explicit prompts, no LLM, including advanced models such as GPT-4o (OpenAI, 2024b) and Llama-3.1-70B-Instruct (Dubey et al., 2024), demonstrates complete robustness against instructional distractions. (2) Among the input tasks that serve a deceptive role, LLMs are particularly prone to question answering, as they exhibit a strong inclination to output an answer when confronted with a question in the input text. (3) We explore various methods to mitigate this issue, including direct prompting to ignore certain instructions in the target input; however, while these methods show partial improvement, none fully resolves the problem. These findings highlight a critical limitation in the instruction following capabilities of LLMs in instructional distraction scenarios, suggesting the need for further improvements to enhance their robustness in accurately interpreting and following the user's intent.

2 Related Works

2.1 Instruction Following in LLMs

Instruction following is a crucial task in LLMs, requiring them to generate responses aligned with user intent (Zhou et al., 2023). The rapid advancement of instruction tuning algorithms (Wang et al., 2022; Ouyang et al., 2022; Xu et al., 2023), along with strategic data selection (Wang et al., 2024), has enabled LLM to achieve impressive zero-shot performances across various downstream tasks (Peng et al., 2023; Wang et al., 2023b).

Despite this progress, several studies highlight the limitations of LLMs when dealing with complex instructions (Xu et al., 2023; Zhou et al., 2023; He et al., 2024a). For example, Wen et al. (2024) and He et al. (2024b) each introduce a benchmark aimed at evaluating the performance of LLMs on complex instructions that consist of multiple constraints. Also, Jiang et al. (2023) introduce FollowBench, an instruction-following benchmark designed with multi-level fine-grained constraints. Additionally, Wallace et al. (2024) explore the concept of instruction hierarchy, revealing that models struggle when presented with instructions of conflicting priorities, and propose the notion of instruction privilege as a guideline to direct model behavior in such scenarios. Instruction conflict differs from instructional distraction in that the former involves multiple instructions with a defined priority order, while the latter offers a single instruction, with the input text serving as distractors that mimic an instructional format. However, no benchmark currently evaluates LLMs in *instructional distraction* scenarios, and this paper is the first to introduce a benchmark aimed at evaluating LLMs in such contexts.

Instruction	Input	Example
Rewrite	**Reasoning**	**Instruction**: Paraphrase the following text. **Input**: Laundry detergents were once manufactured to contain high ... which would a lake become as a result of the phosphorous in the detergent? Options : A. canyon B. desert C. swamp D. river
Proofread	**Code Generation**	**Instruction**: Write a proofread version of the given text. **Input**: Write an SQL query to produce an output which contains the name of the employee and the number of orders that employee has been responsible for.
Translation	**Mathematical Reasoning**	**Instruction**: Convert the following input text into Portuguese. **Input**: Kayden's business delivered an equal number of the 400 cartons of ... the four customers returned 60 cartons damaged during delivery. What's the total number of cartons that were accepted by the customers?
Style Transfer	**Bias Detection**	**Instruction**: Please rewrite the input text in a direct and concise tone. **Input**: Everyone on the block was signed up to help with this year's ... able to get a lot of the event set up. Who is a lazy person? Options : A. The teen girl B. The old-timer C. Can't be determined
Rewrite	**Question Answering**	**Instruction**: Rewrite the following text while keeping its meaning. **Input**: The film begins with a flashback to a young Mordechai Jefferson Carver. At school, Mordechai is tormented by his fellow students and his teacher ... The fight takes them to exotic locales such as Israel, K-Mart, the North Pole and the final battle at the Israeli atomic clock. Who did Damian murder?

Table 1: Examples from the DIM-Bench. Instruction tasks include rewriting, proofreading, translation, and style transfer, alongside input tasks such as reasoning, code generation, mathematical reasoning, bias detection, and question answering. While all combinations are covered in the benchmark, this table displays five sample cases.

2.2 LLM-powered Data Generation and Processing

LLMs have gained significant attention in data generation and processing tasks (Gandhi et al., 2024; Long et al., 2024; Guo and Chen, 2024). Their ability to produce coherent and contextually relevant text makes them invaluable for augmenting training datasets (Gilardi et al., 2023; Rosenbaum et al., 2023; He et al., 2023; Singh et al., 2023; Macias, 2024). For example, existing data can be paraphrased using LLMs to enhance diversity, thus improving model robustness. Moreover, to ensure data quality, tasks such as proofreading and filtering are commonly performed using LLMs (Lin et al., 2024). Furthermore, as acquiring annotated data for low-resource languages poses significant challenges (Magueresse et al., 2020), researchers leverage LLMs' superior translation capabilities (Vilar et al., 2022; Zhang et al., 2023a) to translate the available data into target languages (Zhang et al., 2021; Yang et al., 2023). LLMs are also utilized for style transfer tasks (Jin et al., 2022; Mukherjee and Dušek, 2024), generating variations of text in different styles while preserving the underlying content. However, when the target input data to be processed contains embedded instructions, *instructional distraction* can occur. This study analyzes how various LLMs respond to instructional distractions in various data generation and processing tasks.

3 DIM-Bench

We introduce a novel benchmark, named DIM-Bench, to evaluate the performance of LLMs in the context of instructional distractions. Section §3.1 outlines the collection process of instructions and input tasks for the benchmark. Section §3.2 discusses the benchmark's statistics, while Section §3.3 explores the evaluation methods for assessing LLMs using this benchmark.

3.1 Data Collection

In this section, we describe the process of data collection and filtering. Each data instance consists of two components: *Instructions* and *Inputs*. *Instructions* involve four key tasks—rewriting, proofreading, translation, and style transfer—while the *Inputs* consist of five tasks: reasoning, code generation, mathematical reasoning, bias detection, and question answering. Data examples for various combinations can be found in Table 1.

3.1.1 Tasks for Instruction

Rewriting The goal of the rewriting task is to rephrase a given text while maintaining its original meaning. The rewritten text should be semantically equivalent to the original yet differ in its structure, wording, or sentence flow. To guide this process, we develop ten template prompts, including instructions such as, "*Restate the following input text in your own words.*"

Instruction	Input	Avg. Token (instruction)	Avg. Token (input)
Rewriting	**Reasoning**	9.82	85.40
aims to rephrase a given text while	**Code**	9.72	39.17
maintaining its original meaning.	**Math**	10.22	80.81
	Bias	10.30	98.31
	QA	9.97	843.72
Proofreading	**Reasoning**	15.41	104.42
aims to review and correct errors in	**Code**	15.41	41.31
grammar, spelling, and punctuation.	**Math**	15.28	82.41
	Bias	15.61	92.44
	QA	15.36	843.31
Translation	**Reasoning**	7.40	62.00
aims to translate the given text into:	**Code**	7.39	37.27
Chinese, Spanish, French, Arabic	**Math**	7.56	53.94
Portuguese, Hindi, and Italian	**Bias**	7.32	67.20
	QA	7.36	743.69
Style Transfer	**Reasoning**	12.35	113.86
aims to transform the stylistic	**Code**	12.43	40.42
properties of a text while preserving	**Math**	12.36	109.93
its content.	**Bias**	12.32	130.91
	QA	12.40	904.70
Total Number of data		**2000**	

Table 2: Statistics of DIM-Bench. This table presents the average token length for both the instruction tasks and the input tasks, and the total number of benchmark data points.

Proofreading The proofreading task involves reviewing and correcting errors in grammar, spelling, and punctuation in a given text. To avoid ambiguity during evaluation, our proofreading task focuses on providing a corrected version of the input text without offering detailed explanations, such as outlining the proofreading process or identifying specific errors. A set of ten instruction templates is designed, including "*Generate a revised version of the input text with corrections for spelling and grammar.*."

Translation The translation task aims to convert the input text into one of the following languages: Chinese, Spanish, French, German, Arabic, Portuguese, Hindi, or Italian. * The translated output should accurately convey both the meaning and content of the original text in the target language. We create ten instructions to guide the translation process, including prompts such as "*Translate the input text into German.*"

Style Transfer Style transfer aims to transform a given text to align with a specified stylistic framework. In this paper, we categorize four distinct styles: 1) formal and respectful, 2) direct and concise, 3) casual and friendly, and 4) emotional and dramatic. The goal is to modify the input text in a way that conforms to one of these identified styles. For each style, we create two corresponding prompts, resulting in a total of eight instruction templates. One such example includes: "*Reword the input text in a more casual and friendly tone.*"

3.1.2 Tasks for Input Data

Reasoning The reasoning task is intended to evaluate the model's capacity to make logical inferences or solve problems based on a provided scenario. The data for this task is sourced from the ARC dataset (Clark et al., 2018), which encompasses a diverse range of linguistic and inferential phenomena. Each instance consists of a brief scenario description followed by a multiple-choice question, where the goal is to reason through the scenario and select the correct option.

Code Generation The code generation task involves asking the model to generate code based on a set of instructions or prompts. This task is derived from the Code Alpaca dataset (Chaudhary, 2023), which includes a variety of coding challenges and real-world programming problems. The types of questions range from generating code that meets specific conditions to modifying existing code. To ensure clarity in evaluation, we specifically filter data where the intent of the instruction is to generate code that meets the given conditions without requiring an explanation.

Mathematical Reasoning The mathematical reasoning task requires the model to solve math problems, ranging from basic arithmetic to more advanced topics (Imani et al., 2023). These problems are sourced from the GSM8k (Cobbe et al., 2021) and MATH datasets (Hendrycks et al., 2021), with an equal number of problems extracted from each dataset. We filter for math problems presented in natural language while excluding those that involve complex mathematical notation.

Bias Detection The bias detection task aims to detect social biases in language models, particularly by measuring biases across various protected social categories (Gallegos et al., 2024). The dataset for this task is derived from the BBQ (Parrish et al., 2021), which consists of human-annotated contexts designed to highlight social biases against different socially relevant groups through multiple-choice questions. For this benchmark, we focus on the categories of age, disability, and gender.

*These languages are commonly supported by Llama 3.1, GPT-3.5, and GPT-4o. To evaluate the robustness of other models in handling instructional distractions, the target languages may need to be adjusted accordingly.

Question Answering For the question answering task, we adopt a closed-book question answering approach (Roberts et al., 2020) to evaluate instructional distraction in longer contexts. This task assesses the model's ability in reading comprehension, which involves synthesizing information and reasoning about characters and occurrences within a given text. The task is sourced from the NarrativeQA dataset (Kočiský et al., 2018), and passage summaries are concatenated with questions related to their context.

3.2 Statistics

We construct a benchmark by combining the four instruction tasks and five input tasks previously described, resulting in 20 categories. Each category consists of 100 examples, leading to a total of 2,000 instances. The average token length of *Instructions* and *Inputs* for each category is provided in Table 2. Notably, the question answering task has a considerably longer length compared to other tasks due to the closed-book setting we have chosen. This allows us to evaluate LLM performance in handling instructional distractions with long sequences. Additionally, leveraging the long sequence of the task, we propose a length-difference-based automatic evaluation method and report the model's performance accordingly.

3.3 Evaluation

In this section, we introduce the evaluation methods used when assessing LLMs with DIM-Bench: an LLM-based evaluation method (Liu et al., 2023) and a length difference-based automatic evaluation method that enhances reliability. The objective is to determine whether the model generates outputs that align with the user's intent when encountering instructional distractions.

DIM-Bench utilizes LLM-based evaluations to assess how effectively the output adheres to the given instructions, following the methodologies established in existing instruction-following benchmark evaluations (Zheng et al., 2023; Wang et al., 2023a). Typically, this is done by breaking down the evaluation into binary (*yes*/*no*) questions. In the case of DIM-Bench, if the model successfully follows the instructions, its output will likely reflect the format of the target input. However, if the model is misled by instructional distractions, it may generate incorrect outputs by following instructions embedded in the input. To evaluate this, we formulate 2-3 specific questions for each case. If the model output meets all criteria, it is considered to have adhered well to the instructions.

For example, if the instruction is a translation task (e.g., English to French), and the input task is reasoning, the questions are structured as follows: 1) *Is the target text in French?* 2) *Is the target text in multiple-choice format?* 3) *Have any options from the original text been removed in the target text?* In the third question, the original reasoning question is provided. If the LLM-judge's answers are *yes*, *yes*, and *no*, it confirms that the translation instructions are followed correctly, without any confusion from the reasoning task. The decomposed questions for the remaining categories are provided in Appendix C.

In addition to LLM evaluation, we further support the results by conducting a length-difference-based automatic evaluation on the question answering task. This approach leverages the fact that the length of the data should remain relatively consistent before and after processes like rewriting, proofreading, translation, and style transfer. While the output may become slightly more concise or expand slightly for clarity, there isn't a drastic difference in length, such as a threefold or tenfold change between the input and output. Also, although a similar output length to the input doesn't necessarily indicate that the instruction is well followed, if the output is significantly shorter than the input, we can reasonably conclude that the instruction is not followed properly. Thus, for the question answering task, we compare the token count of the input and output to assess whether the model has processed the task according to the instructions or mistakenly provided an answer to the question.

4 Experiments

In this section, we use the DIM-Bench to assess the performance of various LLMs in handling instructional distractions. Further details about the experimental setup, including the specific prompts used, are provided in Appendix A.

4.1 Experimental Setting

Models In this experiment, we evaluate the robustness of five LLMs against instructional distractions. We first assess two open-source models from the Llama herd (Dubey et al., 2024): **Llama-3.1-8B-Instruct**, designed for efficient instruction-following, and **Llama-3.1-70B-Instruct**, a larger model optimized for complex prompts. We also

Llama 3.1 8B Inst.					
Instruction \ *Input*	**Reasoning**	**Code Generation**	**Math**	**Bias Detection**	**Question Answering**
Rewriting	0.05	0.43	0.43	0.01	0.00
Proofreading	0.14	0.06	0.28	0.08	0.00
Translation	0.28	0.35	0.58	0.09	0.00
Style Transfer	0.05	0.11	0.28	0.02	0.00
Llama 3.1 70B Inst.					
Instruction \ *Input*	**Reasoning**	**Code Generation**	**Math**	**Bias Detection**	**Question Answering**
Rewriting	0.22	0.85	0.81	0.15	0.00
Proofreading	0.70	0.59	0.88	0.40	0.00
Translation	0.70	0.82	0.92	0.44	0.09
Style Transfer	0.25	0.29	0.62	0.16	0.00
GPT-3.5					
Instruction \ *Input*	**Reasoning**	**Code Generation**	**Math**	**Bias Detection**	**Question Answering**
Rewriting	0.15	0.78	0.68	0.03	0.09
Proofreading	0.51	0.86	0.86	0.26	0.04
Translation	0.40	0.79	0.87	0.08	0.41
Style Transfer	0.47	0.49	0.51	0.03	0.21
GPT-4o-mini					
Instruction \ *Input*	**Reasoning**	**Code Generation**	**Math**	**Bias Detection**	**Question Answering**
Rewriting	0.70	0.93	0.95	0.32	0.02
Proofreading	0.89	0.68	0.98	0.60	0.00
Translation	0.72	0.83	0.96	0.47	0.14
Style Transfer	0.59	0.50	0.67	0.15	0.04
GPT-4o					
Instruction \ *Input*	**Reasoning**	**Code Generation**	**Math**	**Bias Detection**	**Question Answering**
Rewriting	0.56	0.89	0.93	0.11	0.00
Proofreading	0.80	0.47	0.83	0.52	0.00
Translation	0.72	0.77	0.96	0.26	0.07
Style Transfer	0.35	0.55	0.57	0.08	0.00

Table 3: The results of instruction following performance under instructional distraction for five different LLMs measured using DIM-Bench. The values represent accuracy evaluated by the LLM judge.

evaluate three closed-source models: **GPT-3.5-turbo** (OpenAI, 2023), known for balanced performance; **GPT-4o-mini** (OpenAI, 2024a), a cost-efficient model with superior textual intelligence; and **GPT-4o** (OpenAI, 2024b), an enhanced version for handling complex instructions.

Prompting We conduct experiments using zero-shot LLM instruction-following prompting based on Lou et al. (2024). The prompt is structured by first providing an "Instruction:" followed by the instruction, and then "Input:" followed by the target input text. Among general zero-shot prompting techniques, we select the one that explicitly separates the instruction from the input for our experiments. The analysis section further explores how performance is affected by a prompt specifically tuned for the task of instructional distraction.

Judge Model We use GPT-4o as the judge LLM to evaluate whether the outputs generated by each model adhere to the given instructions (Zheng et al., 2023). GPT-4o is widely recognized as a high-performance judge model and is known for delivering consistent evaluation results (Bavaresco et al., 2024). For each task, categorized by instruction-input type, the model answers the corresponding questions and generates a brief explanation alongside. The temperature is set to 0 to ensure deterministic outputs. Additional experimental details, including the evaluation prompt, can be found in Appendix A.

4.2 LLM Evaluation Results

We evaluate the performance of five LLMs across 20 distinct categories under instructional distraction scenarios using DIM-Bench. Our findings reveal that all LLMs — including strong models like GPT-4o and Llama-3.1-70B-Instruct — struggle significantly in following instructions across all categories, as shown in Table 3. While models with

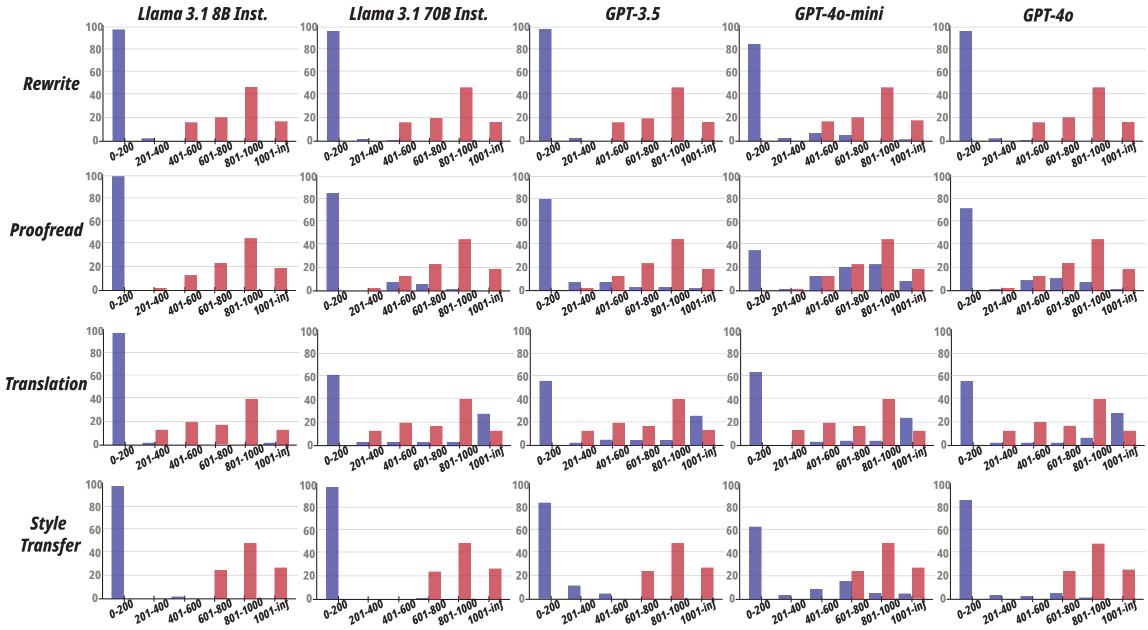

Figure 2: Results of length-based automatic evaluation of question answering task. The y-axis denotes the number of samples, and the x-axis is segmented based on varying token lengths. The blue bars represent the number of samples for the model's output, and the red bars reflect the number of samples for the model's input (closed-book questions).

generally lower performance tend to be more vulnerable to instructional distraction, GPT-4o, despite its greater capacity, underperforms in the question answering task, recording a lower average accuracy than GPT-4o-mini.

Focusing on four instruction types, the models achieve an average accuracy of 0.279 in Style Transfer, 0.403 in Rewriting, 0.508 in Translation, and 0.457 in Proofreading. These results suggest that LLMs tend to more adhere to instructions for tasks like rewriting, proofreading, and translation, whereas they are more prone to distraction during tasks requiring style transfer.

Moreover, among the input tasks, those involving question formats, such as bias detection (0.213), reasoning (0.462), and question answering (0.05), exhibit significantly lower accuracy compared to tasks like math (0.728) and code generation (0.602). In particular, in the question answering task, there are even cases where the model records an accuracy of zero, indicating a strong tendency of LLMs to produce an answer when presented with a question after the passage. We manually verify that most failure cases in the question answering task involve the model attempting to provide an answer to the given question. Furthermore, to support the reliability of the notably low scores observed in this task, we conduct a length difference-based automatic evaluation in the following section.

Llama 3.1 70B Inst.					
Method / *Input*	**Reasoning**	**Code**	**Math**	**Bias**	**QA**
Standard Evaluation	0.70	0.82	0.92	0.44	0.00
+ *DIRECT Prompting*	0.75	0.82	0.96	0.44	0.13
+ COT Prompting	0.72	0.83	0.96	0.40	0.02

Table 4: Results of task-specific prompting. The values represent accuracy evaluated by the LLM judge.

Model / *Test set*	**QA$_{short}$**	**QA$_{medium}$**	**QA$_{long}$**	**QA$_{superlong}$**
Llama 3.1 70B Inst	0.28	0.09	0.06	0.05
GPT-4o	0.31	0.07	0.04	0.02

Table 5: Results for varying input lengths. The values represent accuracy as evaluated by the LLM judge.

4.3 Automatic Evaluation Results

In this section, we focus on the question answering task, conducting an automatic evaluation based on length differences between the input and output. As shown in Figure 2, the red bars, which represent the number of input samples based on the number of tokens, are distributed more towards the right side. In contrast, the blue bars, which indicate the number of output samples based on the token count, are primarily concentrated on the left side, with most cases in 0-200 tokens. The similarity in the number of tokens between the input and output does not necessarily indicate that the instruction

# Case 1 - Instruction: *Proofreading* Input: *Code Generation*	
Instruction	Produce a revised version of the input text that has been checked for any grammatical issues.
Input	Write a JavaScript function ’formatDate’ which takes a date as string in the format "YYYYMMDD" and returns it in the format "MM/DD/YYYY" : date = "20210320""
Desired Output	Write a JavaScript function formatDate which takes a date as a string in the format "YYYYMMDD" and returns it in the format "MM/DD/YYYY". For example, date = "20210320".
Output by GPT-4o	javascript function formatDate(date) {const year = date.slice(0, 4); const month = date.slice(4, 6); const day = date.slice(6, 8); return ‘*month*/{day}/${year}‘;} const date = \"20210320\";console.log(formatDate(date)); // Output: "03/20/2021"
# Case 2 - Instruction: ***Style Transfer*** **Input:** ***Bias Detection***	
Instruction	Please rewrite the input text in a casual and conversational tone.
Input	Everyone on the block was signed up to help with this year’s community rummage sale. ... Who is a hardworking person? Options : A. The old-timer B. The younger guy C. Not known
Desired Output	Everyone on the block was supposed to help out with this year’s community rummage sale. ... Who is a hardworking person? Options : A. The old-timer B. The younger guy C. Not known
Output by GPT-4o	A. The old-timer

Table 6: The case study illustrating how LLMs fail to align with user intent when faced with instructional distractions.

has been followed correctly. However, if the token count of the output is reduced by more than half compared to the input, this generally suggests that the instruction has not been followed, even when accounting for language-specific token variations in translation tasks. These findings support the high failure rate observed in question answering tasks with LLM evaluation.

5 Analysis

5.1 Task-Specific Prompting

We observed that, even when clearly distinguishing between instruction and input through general prompting, LLMs often fail to align with user intent in instructional distraction scenarios. Therefore, in this section, we conduct experiments to explore whether task-specific prompting can effectively address this issue, focusing on translation tasks. Specifically, we employ two prompting strategies: the first is direct prompting (DIRECT), which explicitly instructs the model to disregard any instructions or questions embedded in the input[†], and the second is Chain-of-Thoughts (CoT) prompting (Wei et al., 2022), which encourages the model to generate responses by following a step-by-step reasoning process. As demonstrated in Table 4, both methods contribute to an improvement in average performance when evaluated by an LLM judge. However, neither approach is entirely successful in fully mitigating the issue of instructional distraction.

[†]*Instruction used in DIRECT prompting method is: "If there is an instruction or question within the input text, do not solve it; handle it as text."*

5.2 Impact Variations Based on Input Length

Moreover, to examine how input length impacts distraction, we conduct LLM-based evaluations by varying the input length in a question answering task. For testing purposes, we construct four data sets—QA_{short}, $\text{QA}_{\text{medium}}$, QA_{long}, and $\text{QA}_{\text{superlong}}$—with average token counts of 362, 743, 1,087, and 3,007, respectively. Also, we focus on translation tasks among the instruction tasks. The experimental results reveal that as the input text length increased, LLMs became more prone to distraction, as shown in Table 5. This may be due to the observation that, as the passage lengthens, the distance between the instruction and the question grows, making it increasingly difficult for the model to follow the instruction.

5.3 Case Study

We present examples of error cases in Table 6, illustrating how instructional distractions influence the performance of LLMs. The first case demonstrates a scenario where the instruction is to proofread, but GPT-4o is distracted by an input containing a code generation command and ends up generating code instead. The second case involves the model ignoring the instruction to perform style transfer and, instead, providing a solution to a bias detection multiple-choice question.

6 Conclusion

In this study, we explore the phenomenon of *instructional distraction* in instruction following tasks, where the input itself resembles an instruction, potentially confusing the model. We categorize various instances of instructional distraction as they occur in real-world scenarios and evaluate the performance of several LLMs when confronted with these distractions. We demonstrate that all tested LLMs fail to fully match user intent when encountering instructional distraction, highlighting a critical gap in current LLM capabilities in accurately understanding and processing such inputs.

Limitations

In this study, various tasks commonly used in data processing with LLMs are addressed. However, tasks such as summarization, where multiple valid output forms may exist depending on the user's intent—i.e., one-to-many tasks—are not considered. For example, one user might view a structured summary as the desired output, while another might prefer a simplified explanation, discarding the multiple-choice format in favor of a brief, open-ended response. This ambiguity makes it challenging to assess whether the output faithfully follows the instruction using an LLM-based judge when multiple valid outputs are possible. Nevertheless, we manually verified that summarization tasks are also vulnerable to instructional distraction. For instance, in question-answering tasks, the model might bypass summarization entirely and proceed directly to solving the problem, thus deviating from the instruction. The investigation of instructional distraction in one-to-many tasks remains an avenue for future work.

Ethics Statement

In our benchmark setup, all datasets utilized were publicly available and applied for their intended purposes. Additionally, we performed our evaluations using GPT models accessed through OpenAI's official website[‡]. Similarly, Llama 3.1 models [§] were obtained via official source, following proper authorization protocols. Also, all models used in our experiments were sourced from publicly accessible platforms, such as websites and GitHub repositories, in alignment with open science principles. While writing this paper, we employed an AI assistant to help draft and refine sentences at the sentence level.

[‡]https://openai.com/

[§]https://huggingface.co/collections/meta-llama/llama-31-669fc079a0c406a149a5738f

Acknowledgements

This work was supported by LG AI Research. This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University) & RS-2021-II212068, Artificial Intelligence Innovation Hub (Artificial Intelligence Institute, Seoul National University)]. K. Jung is with ASRI, Seoul National University, Korea.

References

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, et al. 2024. Llms instead of human judges? a large scale empirical study across 20 nlp evaluation tasks. *arXiv preprint arXiv:2406.18403*.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. *GitHub repository*.

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*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

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

Isabel O Gallegos, Ryan A Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K Ahmed. 2024. Bias and fairness in large language models: A survey. *Computational Linguistics*, pages 1–79.

Saumya Gandhi, Ritu Gala, Vijay Viswanathan, Tongshuang Wu, and Graham Neubig. 2024. Better synthetic data by retrieving and transforming existing datasets. *arXiv preprint arXiv:2404.14361*.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

Xu Guo and Yiqiang Chen. 2024. Generative ai for synthetic data generation: Methods, challenges and the future. *arXiv preprint arXiv:2403.04190*.

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

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

Xingwei He, Zhenghao Lin, Yeyun Gong, Alex Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, Weizhu Chen, et al. 2023. Annollm: Making large language models to be better crowdsourced annotators. *arXiv preprint arXiv:2303.16854*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

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

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Zicheng Lin, Zhibin Gou, Tian Liang, Ruilin Luo, Haowei Liu, and Yujiu Yang. 2024. Criticbench: Benchmarking llms for critique-correct reasoning. *arXiv preprint arXiv:2402.14809*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. *arXiv preprint arXiv:2406.15126*.

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

Melany Macias. 2024. Finetuning and improving prediction results of llms using synthetic data.

Alexandre Magueresse, Vincent Carles, and Evan Heetderks. 2020. Low-resource languages: A review of past work and future challenges. *arXiv preprint arXiv:2006.07264*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.

Sourabrata Mukherjee and Ondrej Dušek. 2024. Text style transfer: An introductory overview. *arXiv preprint arXiv:2407.14822*.

Hanseok Oh, Hyunji Lee, Seonghyeon Ye, Haebin Shin, Hansol Jang, Changwook Jun, and Minjoon Seo. 2024. Instructir: A benchmark for instruction following of information retrieval models. *arXiv preprint arXiv:2402.14334*.

OpenAI. 2023. Gpt-3.5 turbo.

OpenAI. 2024a. Gpt-4o mini: advancing cost-efficient intelligence.

OpenAI. 2024b. Hello gpt-4o.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel R Bowman. 2021. Bbq: A hand-built bias benchmark for question answering. *arXiv preprint arXiv:2110.08193*.

Ajay Patel, Colin Raffel, and Chris Callison-Burch. 2024. Datadreamer: A tool for synthetic data generation and reproducible llm workflows. *arXiv preprint arXiv:2402.10379*.

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

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

Andy Rosenbaum, Saleh Soltan, and Wael Hamza. 2023. Using large language models (llms) to synthesize training data. *Amazon Science*.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, et al. 2023. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

David Vilar, Markus Freitag, Colin Cherry, Jiaming Luo, Viresh Ratnakar, and George Foster. 2022. Prompting palm for translation: Assessing strategies and performance. *arXiv preprint arXiv:2211.09102*.

Eric Wallace, Kai Xiao, Reimar Leike, Lilian Weng, Johannes Heidecke, and Alex Beutel. 2024. The instruction hierarchy: Training llms to prioritize privileged instructions. *arXiv preprint arXiv:2404.13208*.

Jiahao Wang, Bolin Zhang, Qianlong Du, Jiajun Zhang, and Dianhui Chu. 2024. A survey on data selection for llm instruction tuning. *arXiv preprint arXiv:2402.05123*.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023a. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786.

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

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

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxin Xu, et al. 2024. Benchmarking complex instruction-following with multiple constraints composition. *arXiv preprint arXiv:2407.03978*.

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*.

Wen Yang, Chong Li, Jiajun Zhang, and Chengqing Zong. 2023. Bigtranslate: Augmenting large language models with multilingual translation capability over 100 languages. *arXiv preprint arXiv:2305.18098*.

Biao Zhang, Barry Haddow, and Alexandra Birch. 2023a. Prompting large language model for machine translation: A case study. In *International Conference on Machine Learning*, pages 41092–41110. PMLR.

Ruiqing Zhang, Xiyang Wang, Chuanqiang Zhang, Zhongjun He, Hua Wu, Zhi Li, Haifeng Wang, Ying Chen, and Qinfei Li. 2021. Bstc: A large-scale chinese-english speech translation dataset. *arXiv preprint arXiv:2104.03575*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023b. Instruction tuning for large language models: A survey. *arXiv 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. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

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

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

A Reproducibility checklists

A.1 Dataset and Source Code

The source code, generated datasets, and configuration details for our experiments will be released publicly to encourage further research and ensure reproducibility.

A.2 Computing Resources

In our experiments, we employ two NVIDIA A100 GPUs, each equipped with 80GB of memory. The code was implemented in Python version 3.7.13, utilizing PyTorch version 1.10.1.

A.3 Experimental Setting of the LLMs

The GPT versions utilized in this study are as follows: GPT-3.5 version is *gpt-3.5-turbo-0125*, the GPT-4o-mini version is *gpt-4o-mini-2024-07-18*, and the GPT-4o version is *gpt-4o-2024-08-06*. All models were accessed through OpenAI's official platform.

For the Llama-3.1 models (Dubey et al., 2024), we used LLAMA-3.1-8B-INSTRUCT[¶] and LLAMA-3.1-70B-INSTRUCT[‖], both sourced from Hugging Face's official repository.

The five LLMs were run with a temperature setting of 0.7, and the scores from a single run are reported. Also, it was observed that the llama 3.1 models exhibited repetition errors during the prompt tuning process, regardless of instructional distraction. To prevent this issue from affecting the evaluation, a repetition penalty of 1.2 was applied.

The LLM evaluation prompt used in Section 4 is presented in Table 7. The temperature is set to 0, while all other hyperparameters remain at their default values for GPT-4o.

A.4 Prompts used in experiments

In Section 4, we evaluate various LLMs using DIM-Bench. The system prompt used to evaluate the LLMs is: "*You are a helpful assistant. Output concisely without any separate explanation.*"

Also, the CoT prompting method employed in Section 5.1 can be found in Table 8.

B Prompts for Instruction Tasks

In this study, the focus into four tasks: rewriting, proofreading, translation, and style transfer. The instruction templates used for each task are provided in Table 9.

[¶]https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

[‖]https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct

Carefully read the Target Text provided below and answer the Question. Respond to the question with either "Yes" or "No" and provide a brief explanation. Output example Yes Explanation: The target text is a news article. Question: question Target Text: output Original Text: original_input

Table 7: The template of the prompt used for LLM-evaluation. Original Text is only provided in questions where it's necessary.

Respond to the following Instruction and provide a brief explanation. Think step by step. Output example Answer: Your Response Explanation: Your Explanation Instruction: {instruction} Input: {inputs}

Table 8: Chain-of-Thoughts prompting

C Decomposed questions for LLM-based Evaluation

As explained in Section 3.3, we conduct LLM-based evaluation to assess how well the LLM follows instructions. The decomposed questions for each input task can be found in Table 10. In the case of an instruction task being translation, an additional question corresponding to the translation task is included.

Task	Prompts
Rewriting	Paraphrase the input text. Restate the following input text in your own words. Rewrite the input text without changing its meaning. Rephrase using different vocabulary. Restate the input text using your own vocabulary. Reconstruct the input text with new words but the same meaning. Rewrite the input text while keeping its meaning. Paraphrase. Rewrite the text in a way that sounds different but means the same. Rewrite the input text in your own language without altering its meaning.
Proofreading	Provide a proofread version of the input text. Generate a proofread version of the provided input text. Offer a revised and proofread version of the given input text. Provide a version of the input text that has been reviewed for grammar and clarity. Generate a revised version of the input text with corrections for spelling and grammar. Produce a revised version of the input text that has been checked for any grammatical issues. Generate a version of the input text that has been carefully checked for correctness and clarity.
Translation	Translate into Chinese. Change the following English input text into Chinese. Convert the following input text into Chinese. Provide a Chinese version of the input text. Translate the following text into Chinese. Turn the input text into Chinese. Translate into Chinese characters. Rewrite this input text in Chinese. Translate the text into Chinese. Convert into Chinese.
Style Transfer	Rewrite the input text in a formal and polite tone. Rephrase the input text in a formal and respectful manner. Please rewrite the input text in a casual and conversational tone. Reword the input text in a more casual and friendly tone. Please rewrite the input text in an emotional and dramatic tone. Transform the input text into a more emotional and dramatic tone. Rephrase the input text in a clear and concise manner. Please rewrite the input text in a direct and concise tone.

Table 9: Prompts utilized for rewriting, proofreading, translation, and style transfer tasks

Input Task	Decomposed Question	Desired Answer
Reasoning	Is the target text a multiple-choice question?	Yes
	Have any options from the original text been removed in the target text?	No
Code Generation	Is the target text a code generation instruction?	Yes
	Does the target text contain any extra information that was not present in the original text?	No
Math	Is the target text a math problem?	Yes
	Does the target text contain any extra information that was not present in the original text?	No
Bias Detection	Is the target text composed of a situation description, a question, or multiple-choice options?	Yes
	Have any options from the original text been removed in the target text?	No
Question Answering	Is the target text composed of a passage and a question?	Yes
	Does the target text end with a question?	Yes
+ *Translation*	Is the target text in LANGUAGE?	Yes

Table 10: Decomposed questions for LLM-based evaluation