Resilience of Large Language Models for Noisy Instructions

Bin Wang[♡,◇], Chengwei Wei[§], Zhengyuan Liu[♡,◇], Geyu Lin[♡], Nancy F. Chen[♡,◇,†]

[♡]Institute for Infocomm Research (I[2]R), A*STAR, Singapore
[◇]CNRS@CREATE, Singapore   [§]University of Southern California, USA
[†]Centre for Frontier AI Research (CFAR), A*STAR, Singapore
wang_bin@i2r.a-star.edu.sg

Abstract

As the rapidly advancing domain of natural language processing (NLP), large language models (LLMs) have emerged as powerful tools for interpreting human commands and generating text across various tasks. Nonetheless, the resilience of LLMs to handle text containing inherent errors, stemming from human interactions and collaborative systems, has not been thoroughly explored. Our study investigates the resilience of LLMs against five common types of disruptions including 1) ASR (Automatic Speech Recognition) errors, 2) OCR (Optical Character Recognition) errors, 3) grammatical mistakes, 4) typographical errors, and 5) distractive content. We aim to investigate how these models react by deliberately embedding these errors into instructions. Our findings reveal that while some LLMs show a degree of resistance to certain types of noise, their overall performance significantly suffers. This emphasizes the importance of further investigation into enhancing model resilience. In response to the observed decline in performance, our study also evaluates a "re-pass" strategy, designed to purify the instructions of noise before the LLMs process them. Our analysis indicates that correcting noisy instructions, particularly for open-source LLMs, presents significant challenges.

1 Introduction

Large language models offer unprecedented capabilities in understanding and generating human-like text (Touvron et al., 2023; Tunstall et al., 2023). Built upon the foundation of pre-trained language models (PLMs) (Wei et al., 2023), large language models inherit and significantly extend the capabilities of their predecessors by following human-readable instructions, enabling a broad spectrum of applications that were previously challenging or infeasible with unavailable training samples (Kojima et al., 2022).
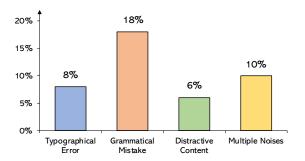

Figure 1: Our analysis scrutinized 500 inputs from real users, focusing on three distinct types of noise. The findings reveal that more than 40% of the inputs to the model are affected by noise.

Meantime, the capability of LLMs to process noisy instructions is a critical feature that enables their applications in real-world scenarios, where data contains imperfections. To validate the extent of such occurrences, we analyzed the noise within user instructions to a chatbot. Specifically, these instructions were evaluated using GPT-4 (Achiam et al., 2023) to detect the presence of the specific noise types. Our statistical analysis, illustrated in Figure 1, indicates that over 40% of user inputs contain typographical errors, grammatical mistakes, or unrelated content in addition to their primary query[1]. Previous research also reveals that human users are inclined to commit errors when interacting with chatbot (James, 2013). It is also treated as an evident social cue for human communications (Bührke et al., 2021). Therefore, examining LLM's proficiency in managing noisy text inputs is critical to practical applications.

In our study on deploying Large Language Models (LLMs) across various applications, we categorized the kinds of noisy instructions from three primary sources. First, from a linguistic standpoint, our focus is on grammatical mistakes and typo-

---

[1]The dataset for this study comprises user inputs sourced from the ShareGPT dataset, as referenced in Chiang et al. (2023).

graphical errors. Second, we explore noise stemming from system integration, specifically errors originating from Optical Character Recognition (OCR) and Automatic Speech Recognition (ASR) technologies. Lastly, we investigate the impact of destructive content from previous interactions or extended contexts. This part of our study aims to assess the models' proficiency in isolating current queries from past interactions, evaluating their effectiveness in disregarding irrelevant content.

We observe distinct performance impacts across three open and closed-sourced models, when faced with different noise types. First, we find that a higher resilience of models to grammatical mistakes, likely because these errors are also present in data used for pre-training and supervised fine-tuning as also revealed in Figure 1. This familiarity enables models to more accurately interpret the intended meaning despite such inaccuracies. In contrast, errors from ASR and OCR systems, which are less common in training datasets, present more significant challenges for the models. Furthermore, our study highlights that models are susceptible to being influenced by previous instructions in both cooperative and non-cooperative manner, which can lead to deviations in responses in subsequent interactions. This suggests a limitation in the models' ability to filter out irrelevant or distracting content from past exchanges.

As noisy instructions can be harmful to model perfomrnace, we investigate the potential of leveraging LLMs to mitigate the impact of noisy instructions through a "re-pass" strategy. This approach involves a two-step process: initially, we employ an LLM to conduct zero-shot text normalization to purify the noisy instructions. Next, we prompt the model to process upon the cleaned instruction. Our findings reveal that not all models are adept at fulfilling this role of data normalization. The exception is ChatGPT, which demonstrates a comprehensive understanding of the text and can recover the instruction with different types of noises.

2 Related Work

The progression of general-purpose Large Language Models (LLMs) such as ChatGPT (Achiam et al., 2023), Gemini (Team et al., 2023), LLaMa (Touvron et al., 2023), Mistral (Jiang et al., 2023a), and Gemma (Mesnard et al., 2024) has facilitated a myriad of real-world applications. These advancements are attributed to their capabilities in managing long-range textual dependencies, enhancing contextual comprehension, and displaying a remarkable ability to adapt to a wide array of tasks with minimal need for detailed, task-specific training. Meantime, several recent studies have demonstrated that the user prompt significantly influences task performance, highlighting its indispensable role in the process (Wang et al., 2023; Zhu et al., 2023). The following will introduce studies on prompt sensitivity and noisy text reconstruction as related work.

2.1 Instruction Sensitivity

Pre-trained large language models exhibit performance variability even with semantically similar inputs. SeaEval (Wang et al., 2023) demonstrates that across five different input templates, performance can fluctuate between 5% and 10%, depending on the dataset. Similar observations are reported in Sclar et al. (2023), highlighting this instability. Moreover, introducing brief sentences such as "Let's think step by step" can significantly enhance performance on reasoning tasks, further underscoring the LLMs' sensitivity (Kojima et al., 2022). Leveraging this characteristic, various studies concentrate on decomposing and crafting improved prompts to efficiently tackle tasks. Zhou et al. (2023) proposes an additional LLM as a prompt engineer to automatically create prompts that enhance performance. Alternatively, some approaches advocate for the use of search (Prasad et al., 2023) or optimization techniques (Khot et al., 2022; Hao et al., 2022; Prasad et al., 2023) to identify superior instructions, replacing those that are less effective.

As an extensive benchmark for adversarial prompts, PromptBench (Zhu et al., 2023) offers a platform to evaluate the resilience of LLMs against attacks across four levels: character, word, sentence, and semantic. In contrast to their research which uses models like DeepWordBug (Gao et al., 2018) and TextBugger (Li et al., 2018), our work does not aim to introduce adversarial prompts to mislead the model into making errors. Instead, we seek to replicate real-use scenarios where errors might naturally occur and be harmless. Additionally, we categorize different instruction noises, offering a more comprehensive analysis.

2.2 Reconstruction of Noisy Instructions

Our study extends beyond merely assessing the model's resilience to textual noise; it delves into deploying a comprehensive model designed to rec-
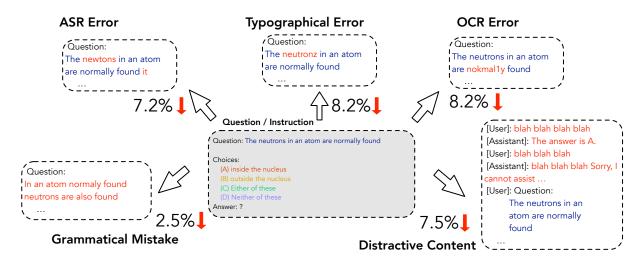
Figure 2: Our study identifies and assesses the impact of five distinct categories of textual disruptions on the ChatGPT-3.5 model's effectiveness. We noted a reduction in accuracy between 2.5% to 8.2% across the MMLU dataset, a phenomenon directly linked to these varied types of noisy instructions.

tify a wide spectrum of input errors. The literature review uncovers a variety of approaches specifically devised to address the multifaceted errors highlighted in our research, each tailored to its unique context. These methodologies span from Automatic Speech Recognition (ASR) Error Correction (Mani et al., 2020; Leng et al., 2021; Jiang et al., 2023b), which aims to amend errors in speech-to-text transcriptions, to Grammar Error Correction (Yuan and Briscoe, 2016; Bryant et al., 2023), focusing on rectifying grammatical inaccuracies in written text. Additionally, Typographical Error Correction (Church and Gale, 1991; Zhang et al., 2020) methods are explored to fix misspellings and typographical mistakes, while Optical Character Recognition (OCR) Error Correction (Tong and Evans, 1996; Soper et al., 2021; Nguyen et al., 2021) seeks to correct errors introduced during the digitization of printed texts.

3 Noisy Instruction and Analysis

In this section, we describe our methodology for integrating five types of noise into the MMLU benchmark (Hendrycks et al., 2021), where the original text is devoid of any noise. We employ hybrid rule-based techniques to introduce noise for OCR and Typographical errors. For ASR and Grammatical errors, we leverage a language model to capture error patterns and simulate these errors through a generative process. To simulate distractive content, we embed actual dialogues as irrelevant background information. An illustration of each type of noisy dataset is provided in Figure 2 and a summarization

Noise Type	Sources
ASR	LibriSpeech (Panayotov et al., 2015) CommonVoice-15 (Ardila et al., 2020)
OCR	NLPAug (Ma, 2019) OCR Engine
Grammatical	JELEG (Napoles et al., 2017) C4-200M (Stahlberg and Kumar, 2021)
Typographical	NLPAug (Ma, 2019) Keyboard, Spelling, Random
Distractive Content	ShareGPT (Chiang et al., 2023)

Table 1: A summary of the techniques and datasets.

is shown in Table 1.

3.1 Automatic Speech Recognition (ASR)

3.1.1 Method

Given that the original texts are not available in audio format, we propose employing a generative model to effectively replicate the patterns of ASR errors. This method enables us to inject realistic ASR errors into pre-existing textual materials. Specifically, we utilize one of the premier ASR models, Whisper-Tiny (Radford et al., 2023), as our ASR engine (M_{ASR}). We utilize the CommonVoice-15 (Ardila et al., 2020) dataset and the noisy test set from the LibriSpeech (Panayotov et al., 2015) dataset as the source data. These datasets together offer over 1,000 hours of speech data, all of which are processed by our ASR engine without prior exposure (zero-shot). We then generate the ASR output texts (T_n), which include ASR-induced errors, by processing the simulated audio through our ASR engine $T_n = M_{ASR}(T_c)$,
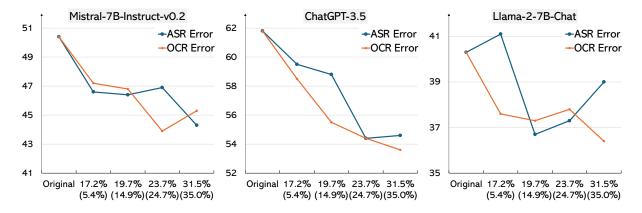
Figure 3: Evaluation of the performance of three Large Language Models (LLMs) using the adapted MMLU dataset, emphasizing different error ratios, as measured by Word Error Rate (WER). The x-axis represents the WER values for Automatic Speech Recognition (ASR) and Optical Character Recognition (OCR), indicated within brackets. The performance declines with noisy instructions.

resulting in outputs that diverge from the original clean transcripts (T_c).

Using the provided dataset of clean and predicted transcripts, we divide them into four distinct categories based on their Word-Error-Rate (WER): less than 10%, 10%-20%, 20%-30%, and 30%-40%, allocating 80,000 samples to each category. By leveraging this paired data, we finetune Tiny-Llama-1.1B-Chat (Zhang et al., 2024) models for each category to learn the underlying patterns of ASR errors. We use the trained error generation model to introduce ASR errors into each question sentence from the MMLU dataset independently. This approach produces a varied WER (Word Error Rate), as illustrated in Figure 3. This noise injection method is consistently applied across all types of noise, except for the destructive content. The same set of 1,000 questions is selected for five noisy instructions to benchmark the performance.

3.1.2 Discussion

Figure 3 presents the noisy instructions alongside the corresponding accuracy. ASR noise embedded in the instructions is harmful to all models. As the WER increases, the magnitude of performance accuracy drops accordingly. This trend highlights a critical vulnerability in current models when dealing with speech recognition errors. It is worth noticing that the close-sourced ChatGPT-3.5 model is as vulnerable as open-sourced models like Mistral and Llama.

Although LLMs have numerous applications in processing spoken content, they lack robustness against errors introduced by ASR systems. Consequently, there is a critical need for the development of LLMs that are resilient to ASR errors, as well as the creation of comprehensive speech-to-text foundation models that can directly handle speech inputs (Chu et al., 2023; Tang et al., 2024).

3.2 Optical Character Recognition (OCR)

3.2.1 Method

OCR technology is prone to specific types of errors, often misclassifying items that appear visually similar, especially on word-level. To simulate OCR errors, we employed the OcrAug engine (Ma, 2019), enhancing it with broader OCR mapping dictionaries to inject errors into clean text. We expanded the initial OCR error dictionary from 12 to 36 groups of characters prone to misclassification. Each word is altered by replacing it with versions that include easily misclassified characters, introducing OCR errors with variations of 1 to 3 characters. To simulate varying degrees of OCR error severity, we adjust the number of words altered, categorizing them into four distinct groups following the above convention.

3.2.2 Discussion

The findings, in Figure 3, reveal that all models demonstrate a lack of robustness to OCR errors, showing a higher performance decline compared to similar WER from ASR errors. This may stem from the characteristics of LLMs. First, LLMs use BPE tokens (Gage, 1994) for pre-processing, meaning the corrupted words will not follow the original tokenization scheme. Such discrepancies can result in words being split into multiple tokens, significantly disrupting the original semantic representation. In contrast, ASR errors tend to per-
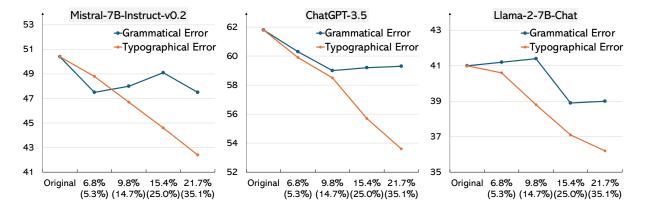
Figure 4: Evaluation of the performance of three Large Language Models (LLMs). The x-axis represents the WER values for grammatical mistakes and typographical errors, indicated within brackets.

verse word integrity. Additionally, the pre-training phase for these models seldom includes text with OCR-induced errors, which are rooted in visual effects. Enhancing OCR robustness of LLMs should include both pre-training exposure and tokenization strategies. However, it is important to note that character-level tokenization (Xue et al., 2022), despite its potential benefits, is still inferior to common subword tokenization methods.

3.3 Grammatical Mistakes

3.3.1 Method

To replicate grammatical errors, similar to those produced for ASR systems, we conducted training on a generative model to emulate this pattern. Our approach involves utilizing two primary sources to gather pairs for both clean and noisy text containing grammatical errors: JELEG (Napoles et al., 2017) and C4-200M (Stahlberg and Kumar, 2021). Both datasets serve the purpose of grammatical error corrections, and we employ their pairs in reverse sequence, thereby transitioning from grammatical error correction to error injection. Four models are trained to learn error patterns with four distinct WER ranges and subsequently applied to each question sentence from the MMLU dataset to simulate grammatical mistakes.

3.3.2 Discussion

The performance on the MMLU dataset with grammatical errors is shown in Figure 4. We observe that LLMs exhibit a more resilient performance in handling grammatical mistakes. Specifically, the performance deterioration of LLM when dealing with grammatical errors is less severe compared with other types of errors. This suggests that LLMs

possess a certain degree of robustness to grammatical mistakes, indicating their ability to process contextualized information even with grammatical deficiencies. We expect that the LLM pre-training and fine-tuning stages have been exposed to a reasonable amount of content with grammatical mistakes, which aligns with our findings shown in Figure 1.

3.4 Typographical Errors

3.4.1 Method

To address typographical errors, we utilize a hybrid approach that combines three character-level modifications, as implemented in the NLPAug package (Ma, 2019). The modifications are derived from three primary sources to construct text with typographical errors: 1) Spelling errors, comprising 13,000+ groups of commonly misspelled words. 2) Keyboard errors, which simulate errors arising from mistyping characters that are physically close to each other on the keyboard. 3) Random errors, where characters are arbitrarily replaced by others. Each type of error is equally likely to occur, ensuring a diverse representation of typographical errors in the generated noisy text. The word error rate is improved by adjusting the number of words that are altered and each word can have a maximum of 3 altered characters for spelling and random errors. The number of adapted words is changed to be categorized into four distinct WER groups.

3.4.2 Discussion

Figure 4 presents the results with the WER specified in parentheses for typographical errors. From the results, we can see that the performance of LLM is severely influenced by typographical errors. The analysis in Figure 1 reveals that a fraction of text
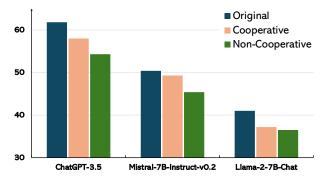
Figure 5: The performance of LLMs with both cooperative and non-cooperative distactive content. Both lead to performance declines while non-cooperative distractions have a more disruptive impact.

data contains typographical errors. These errors often result in tokenization issues similar to those observed with OCR errors, contributing to the reduction in performance.

3.5 Distractive Content

3.5.1 Methods

When interacting with large language models, users may introduce irrelevant information into their input for multifaceted reasons. This can occur due to a lack of clarity about previous interactions, retrieval of unrelated documents in Retrieval-Augmented Generation (RAG) systems, or simply by accident. With historical content, LLM can grasp the context more effectively and deliver responses that are more tailored to the context. However, the impact of irrelevant content on the performance related to the most recent instructions remains uncertain. Therefore, we study the effect of irrelevant content on its influence on the current instruction. Specifically, we add one turn of irrelevant dialogue content sampled from the ShareGPT dataset (Chiang et al., 2023), which consists of real-user interactions with other LLMs. The utterance speaker information is injected as shown in Figure 2. Here we study two scenarios of user interactions.

Cooperative distraction indicates that the user follows multi-turn dialogue patterns provided by respective models. It can be viewed that the user forgot to clean chat histories while initiating requests, which occurs frequently in human-chatbot interactions. The model must possess the ability to discern the lack of relevance between current instructions and historical information. This capability is essential for ensuring the model does not

mistakenly integrate past interactions into current responses, leading to inaccurate responses.

Non-cooperative distraction indicates that the irrelevant content is concatenated directly with the current inquiries without following the designed template of the particular chatbot model. In instances where RAG systems are employed, retrieving content irrelevant to the current instruction is possible. When concatenated with the instruction, such unrelated content can adversely affect the responses.

3.5.2 Discussion

Figure 5 demonstrates the impact of both cooperative and non-cooperative distractive content on the performance of three models. It reveals that introducing distractions can result in performance decline across all models. As expected, the non-cooperative distractive content exhibits a more significant impact. More specifically, the analysis indicates a performance decline of *ChatGPT-3.5* by 3.8% and 7.5% for cooperative and non-cooperative distractions, respectively. This trend is consistent across other models such as *Mistral-7B-Instruct-v0.2* and *Llama-2-7B-Chat*.

In cooperative settings, this decline indicates the models' inability to completely disregard irrelevant dialogue history while processing current requests, as responses tend to be context-dependent. While the capability to generate context-dependent responses based on dialogue history can be advantageous, our findings suggest that it becomes harmful when the history consists of irrelevant distractions. This may be because the models are commonly tuned for multi-turn dialogue instructions, where context dependency is emphasized and irrelevant context is rarely introduced. Therefore, enhancing models' ability to discern relevant from irrelevant content is crucial in further model development to show higher robustness in handling distractions. On the other hand, non-cooperative settings present even higher challenges for the model in terms of isolating irrelevant content. It is particularly crucial for systems augmented with Retrieval-Augmented Generation (RAG), where retrieving irrelevant information from the database can lead to performance decline. Consequently, dynamic retrieval strategies and filtering techniques are necessary to enhance the robustness of models towards distractions and maintain optimal functionality as discussed in Asai et al. (2023).

Harmonizer	WER	Base Acc	ChatGPT-3.5	Mistral-7B-Instruct-v0.2	Llama-2-7B-Chat
Clean	0%	50.4%	+0.4% (50.8%)	-3.8% (46.6%)	-5.3% (45.1%)
ASR Error	17.2%	46.6%	+2.9% (49.5%)	-1.4% (48.0%)	-1.2% (45.4%)
	19.7%	46.4%	+3.3% (49.7%)	+0.8% (47.2%)	-0.8% (45.6%)
	23.7%	46.9%	+3.2% (50.1%)	+1.9% (48.8%)	-2.2% (44.7%)
	31.5%	44.3%	+2.3% (46.6%)	+2.2% (46.5%)	-1.6% (42.7%)
OCR Error	5.4%	47.2%	+3.1% (50.3%)	-0.1% (47.1%)	-0.6% (46.6%)
	14.9%	46.8%	+1.7% (48.5%)	+0.9% (47.7%)	-0.7% (46.1%)
	24.7%	43.9%	+5.0% (48.9%)	+3.0% (46.9%)	+0.3% (44.2%)
	35.0%	45.3%	+0.6% (45.9%)	+1.7% (47.0%)	-3.3% (42.0%)
Grammatical Error	6.8%	47.5%	+2.3% (49.8%)	+2.4% (49.9%)	-3.0% (44.5%)
	9.8%	48.0%	+1.7% (49.7%)	+0.4% (48.4%)	-2.1% (45.9%)
	15.4%	49.1%	+1.2% (50.3%)	-1.6% (47.5%)	-5.0% (44.1%)
	21.7%	47.5%	+2.2% (49.7%)	+0.1% (47.6%)	-4.6% (42.9%)
Typographical Error	5.3%	48.8%	+1.3% (50.1%)	+0.5% (49.3%)	-2.6% (46.2%)
	14.7%	46.7%	+3.0% (49.7%)	+1.0% (47.7%)	-2.5% (44.2%)
	25.0%	44.6%	+6.4% (51.0%)	+3.0% (47.6%)	-0.5% (44.1%)
	35.1%	42.4%	+5.0% (47.4%)	+1.9% (44.3%)	-0.1% (42.3%)

Table 2: Performance evaluation of *Mistral-7B-Instruct-v0.2* on modified noisy MMLU datasets corrected by three different Large Language Models (LLMs). "WER" represents the word-error-rate and "Base Acc" refers to the initial accuracy of noisy dataset prior to any corrections applied using LLMs.

4 Recovery of Noisy Instructions

Previous research has demonstrated the capability of language models to amend specific errors (Ma et al., 2023; Mai and Carson-Berndsen, 2024). In this section, we explore the effectiveness of utilizing Large Language Models (LLMs) for zero-shot correction of four previously identified types of noise.

4.1 Methods

We employ a "re-pass" strategy to investigate whether LLMs can be used to recover clean instruction from noisy counterparts. As shown in Figure 6, the noisy instructions are processed with a large language model (e.g. *ChatGPT-3.5*, *Mistral-7B-Instruct-v0.2*) to correct errors contained in the instructions. After that, the revised instruction is fed into the task-solving LLM to perform the desired task.

4.2 Results and Analysis

In Table 2, we show the evaluation results of *Mistral-7B-Instruct-v0.2* model with the instructions being cleaned by three LLMs including *ChatGPT-3.5*, *Mistral-7B-Instruct-v0.2* (self-correction) and *Llama-2-7B-Chat*.

First, we witness that *Llama-2-7B-Chat* does not show a good capability in error correction of noisy instructions. It even leads to performance drop even after the correction process. We witness that
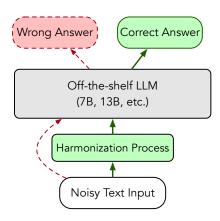

Figure 6: The "re-pass" strategy involves a preliminary step where noisy instructions undergo a harmonization process to refine before input into the model for generating responses.

the model could not follow instruction as well as other models and the revised instructions can be modified with hallucinations added, which makes the final answer unanswerable. Therefore, even with clean instruction as input, the performance on corrected instructions drops up to 5.3%.

Second, in comparison to *Mistral-7B-Instruct-v0.2*, *ChatGPT-3.5* demonstrates superior performance in detecting and amending errors. Utilizing the re-pass strategy enables the recovery of most mistakes, particularly in samples with a Word Error Rate (WER) of up to 30%. It is anticipated that a WER exceeding 30% may often result in damage to the model that cannot be easily reversed.

	Example #1
Clean Instruction	An American firm moves a manufacturing plant from the United States to Brazil. How will this affect gross domestic product (GDP) in the United States and in Brazil?
Noisy Instruction (ASR)	and american firm moved manufacturing plant from the united states to brazil. how well this affect gross domestic product g p in the united states and (void) brazil
ChatGPT-3.5	An American firm moved its manufacturing plant from the United States to Brazil. How will this affect the gross domestic product (GDP) in the United States and (void) Brazil?
Mistral-7B-Instruct-v0.2	An American firm moved its manufacturing plant from the United States to Brazil. How will this affect gross domestic product (GDP) in the United States and (void) Brazil?
Llama-2-7b-Chat	(void) American firm moved manufacturing plant from the United States to Brazil. (– void –)
	Example #2
Clean Instruction	Darwin's mechanism of natural selection required long time spans in order to modify species. From whom did Darwin get the concept of Earth's ancient age?
Noisy Instruction (Grammar)	Darwin's mechanism of natural selection require long time spans in order to modify species. From whom Darwin got the concept of Earth's old age?
ChatGPT-3.5	Darwin's mechanism of natural selection requires long time spans in order to modify species. From whom did Darwin get the concept of (void) Earth's old age?
Mistral-7B-Instruct-v0.2	Darwin's theory of natural selection requires long time spans to modify species. From whom did Darwin get the concept of the Earth's old age?
Llama-2-7b-Chat	Darwin's mechanism of natural selection requires long time spans in order to modify species. (– void –)

Table 3: Two qualitative examples of noisy instruction correction using LLMs.

Third, the self-correction mechanism of *Mistral* exhibits limited effectiveness. While capable of rectifying certain mistakes, it may inadvertently introduce new errors when handling clean instructions, resulting in a performance decline of up to 3.8%. Consequently, this creates an unavoidable barrier to deploying such models in real-world applications unless in an environment where noise is guaranteed.

Regarding types of noise, grammatical errors are typically easier to correct. Such errors are critical to comprehension and do not interfere with the tokenization process much. Therefore, they exert minimal impact on overall performance and are the least challenging noise type to be corrected.

Qualititive Study. Table 3 presents two examples comparing the effectiveness of various models. The results demonstrate that while *ChatGPT-3.5* may not fully restore the original instruction, the resulting instructions are more comprehensible. The worst case is *Llama-2-7B-Chat* which often results in additional information loss from the original instructions.

4.3 Discussion on Efficacy

In this section, we explore how effective Large Language Models (LLMs) are at mitigating the impact of noisy instructions. However, there are two major drawbacks: 1) open-sourced models generally perform poorly in this task and 2) there is an extra computational cost associated with processing re-quests. Therefore, there is a need for a lightweight model that is task-agnostic for noisy instruction correction. During our research, we explored fine-tuning an LLM with 1.1B model (Zhang et al., 2024) sizes using synthesized data to recover clean instructions from noisy ones. However, we found it is challenging for the model to grasp the real intent behind instructions and as a result, unable to performance error corrections accurately. This difficulty is attributed to the constraints imposed by the model size. Therefore, efficient correction of instructions require further investigation, which holds significant use cases like defending adversarial attacks and system integration (e.g. ASR and OCR).

5 Conclusion

In this study, we delve into the resilience of LLMs against noise in instructions from human interactions and system integration. This highlights the complex challenge of processing and recovering accurate information from noisy inputs. Further, we investigate into the "re-pass" strategy and spot the limitations of current open-sourced models in handling noise corrections. Our findings reemphasise that stronger noisy correction and resilience capabilities are required for LLMs, especially for system integration like ASR and OCR, and process various user requests under both cooperative and no-cooperative settings.

Limitations

First, injecting real noise patterns into the evaluation process poses a significant limitation. Simulating authentic noise that accurately reflects the varied and complex errors encountered in real-world data is challenging. This difficulty arises because noise can stem from numerous sources, such as human errors, system glitches, or environmental interference. In this study, we leverage real sample with error pairs, enabling LLM to simulate the error pattern as much as possible. The grasped knowledge is then applied to introduce noise in the aspects of Automatic Speech Recognition (ASR) and grammatical errors. However, it's important to acknowledge that this process may lead to potential information loss.

Second, our analysis and error types are limited to English benchmarks without extension to multilingual scenarios. The problem becomes more complex as each language has its own uniqueness. Moreover, coding-switching noise introduce further complexities. The assessment of LLM's resilience to noisy instructions in multilingual scenarios is an area needs future explorations.

Last, in this study, we focus on five types of noise rooted from system integration (ASR, OCR) and user interactions (Typographical, Grammatical, and Distraction Content). While comprehensive within its defined scope, our work does not encompass all possible sources of noise that could affect LLM performance. For instance, semantic ambiguities or stylistic variations, which could significantly impact the interpretation and processing capabilities of LLMs, are not investigated in details.

References

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

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France. European Language Resources Association.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe. 2023. Grammatical error correction: A survey of the state of the art. *Computational Linguistics*, 49(3):643–701.

Johannes Bührke, Alfred Benedikt Brendel, Sascha Lichtenberg, Maike Greve, and Milad Mirbabaie. 2021. Is making mistakes human? on the perception of typing errors in chatbot communication. *Proceedings of the 54th Hawaii International Conference on System Sciences*.

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

Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. 2023. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*.

Kenneth W Church and William A Gale. 1991. Probability scoring for spelling correction. *Statistics and Computing*, 1:93–103.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Yaru Hao, Zewen Chi, Li Dong, and Furu Wei. 2022. Optimizing prompts for text-to-image generation. *arXiv preprint arXiv:2212.09611*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Carl James. 2013. *Errors in language learning and use: Exploring error analysis*. Routledge.

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

Ridong Jiang, Wei Shi, Bin Wang, Chen Zhang, Yan Zhang, Chunlei Pan, Jung Jae Kim, and Haizhou Li. 2023b. Speech-aware multi-domain dialogue state generation with asr error correction modules. In *Proceedings of The Eleventh Dialog System Technology Challenge*, pages 105–112.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiangyang Li, Edward Lin, and Tie-Yan Liu. 2021. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition. *Advances in Neural Information Processing Systems*, 34:21708–21719.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Rao Ma, Mengjie Qian, Potsawee Manakul, Mark Gales, and Kate Knill. 2023. Can generative large language models perform asr error correction? *arXiv preprint arXiv:2307.04172*.

Long Mai and Julie Carson-Berndsen. 2024. Enhancing conversation smoothness in language learning chatbots: An evaluation of gpt4 for asr error correction. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11001–11005. IEEE.

Anirudh Mani, Shruti Palaskar, Nimshi Venkat Meripo, Sandeep Konam, and Florian Metze. 2020. Asr error correction and domain adaptation using machine translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6344–6348. IEEE.

Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, and et al. 2024. Gemma. *Kaggle*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

Thi Tuyet Hai Nguyen, Adam Jatowt, Mickael Coustaty, and Antoine Doucet. 2021. Survey of post-ocr processing approaches. *ACM Computing Surveys (CSUR)*, 54(6):1–37.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3845–3864, Dubrovnik, Croatia. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*.

Elizabeth Soper, Stanley Fujimoto, and Yen-Yun Yu. 2021. BART for post-correction of OCR newspaper text. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 284–290, Online. Association for Computational Linguistics.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47, Online. Association for Computational Linguistics.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun MA, and Chao Zhang. 2024. SALMONN: Towards generic hearing abilities for large language models. In *The Twelfth International Conference on Learning Representations*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Xiang Tong and David A. Evans. 1996. A statistical approach to automatic OCR error correction in context. In *Fourth Workshop on Very Large Corpora*, Herstmonceux Castle, Sussex, UK. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288.*

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944.*

Bin Wang, Zhengyuan Liu, Xin Huang, Fangkai Jiao, Yang Ding, Ai Ti Aw, and Nancy F Chen. 2023. Seaeval for multilingual foundation models: From cross-lingual alignment to cultural reasoning. *arXiv preprint arXiv:2309.04766.*

Chengwei Wei, Yun-Cheng Wang, Bin Wang, and C-C Jay Kuo. 2023. An overview on language models: Recent developments and outlook. *arXiv preprint arXiv:2303.05759.*

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–386, San Diego, California. Association for Computational Linguistics.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 882–890, Online. Association for Computational Linguistics.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. *ICLR.*

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Neil Zhenqiang Gong, Yue Zhang, et al. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv preprint arXiv:2306.04528.*

A Instruction Templates

Task	Prompt Template
Noise Simulation	`[Uttrance_1] Please help me generate errors in the sentence:` `« ${Instruction} » [/Uttrance_1]`

Error Corrections
 (*ChatGPT-3.5*)
	`[Uttrance_1] You are an error correction assistant. Do not output`
	 `additional explanations besides the corrected instruction. [/Uttrance_1]`
	 `[Uttrance_2] Please help me correct the instruction if it contains any error.`
	 `Instruction: ${Instruction}. Corrected Instruction: [/Uttrance_2]`

Error Corrections
 (*Mistral-7B-Instruct-v0.2*)
	`[Uttrance_1] You are a chatbot who always responds with corrected instructions.`
	 `[/Uttrance_1] [Uttrance_2] No problem! I will just correct the errors in the`
	 `content without any other output. Let's get started! [/Uttrance_2] [Uttrance_3]`
	 `Please help me correct possible errors in the instruction. Do not output anything`
	 `else. Instruction: ${Instruction} Corrected Instruction: [/Uttrance_3]`

Error Corrections
 (*Llama-2-7B-Chat*)
	`[Uttrance_1] You are a chatbot who always responds with corrected instructions.`
	 `[/Uttrance_1] [Uttrance_2] No problem! I will just correct the errors in`
	 `the content and output the corrected content without any other outputs.`
	 `[/Uttrance_2] [Uttrance_3] Please help me correct possible errors in`
	 `the instruction. Do not output anything else. Instruction: ${Instruction}`
	 `Corrected Instruction: [/Uttrance_3]`

Table 4: Templates for simulating noise and correcting errors. The dialogue template adheres to the format specified by the respective models.