FRI Natural language processing course 2025
Project report

University of Ljubljana
Faculty of Computer and Information Science

Improving Prompt Sensitivity of LLMs

Gonçalo Cardoso, Gopika Krishnan, Ali Muhammad

Abstract

The abstract goes here. The abstract goes here.

Keywords
Keyword1, Keyword2, Keyword3 ...

Advisors:

Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across various natural language processing tasks. However, a significant challenge in their practical application is their sensitivity to subtle variations in user prompts. Even minor changes in phrasing, such as word order or the inclusion of seemingly innocuous words, can lead to drastically different outputs in terms of content, quality, and even correctness. This phenomenon, known as prompt sensitivity, undermines the reliability and consistency of LLMs, hindering their widespread adoption in critical applications. Several studies like Mizrahi et al [1] have highlighted this issue, demonstrating the instability of LLM performance under slight prompt perturbations, the work presented by Cao et al [2] shows a difference of up to 45.48% on performance for two paraphrased prompts. Understanding and mitigating prompt sensitivity is therefore crucial for building robust and dependable LLM-based systems.

Prompt sensitivity has garnered increasing attention in the research community, leading to the development of various evaluation metrics and mitigation strategies.

Chatterjee et al [3] introduced POSIX, a metric to evaluate the output invariance of LLMs across paraphrased prompts. ROBUSTALPACAEVAL (Cao et al [2]), provides a benchmark for evaluating the robustness of instruction-following models against adversarial prompt variations by comparing model performances. Similarly, PromptSensiScore (Zhuo et al [4]) offers a measure to assess the degree to which model outputs change with small prompt modifications. Mizrahi et al [1] further introduce metrics such as MaxP, AvgP, and CPS to quantify both the peak and consistency of model performance across prompt variations. These metrics provide valuable tools for analyzing and comparing the prompt sensitivity of different LLMs and prompting techniques.

Researchers have explored various approaches to reduce the sensitivity of LLMs to prompt variations. Techniques like voting, where multiple responses generated from slightly different prompts are aggregated to produce a more stable and reliable output and self-refinement techniques, where the model iteratively refines its own output based on different prompt variations have been tested by Cao et al [2] with non promising results although voting showcases a stable performance. Prompting strategies such as Few-Shot learning (Zhuo et al [4]), Chain-of-Thought Prompting (Lu et al [5]), and General Knowledge Prompting (Lu et al [5]) aim to guide the model towards more consistent and accurate responses by providing relevant context or reasoning steps within the prompt itself. The results of these strategies are more promising but there is a lack of generality for different models.

Planned Methods

To investigate and mitigate prompt sensitivity in LLMs, we will benchmark different prompting strategies, evaluate reasoning models for robustness, and explore fine-tuning for controlled text generation. We will conduct experiments on open-source models that can be manipulated locally.

The language models used in this study are:

- Meta AI's second-generation Llama model with 7 billion parameters, fine-tuned for chat and hosted on Hugging Face. **Technical Name:** Llama-2-7b-chat-hf
- A compact yet capable language model developed by MosaicML. **Technical Name:** mosaicml/phi2
- A 6.7 billion parameter model from DeepSeek AI, specifically designed and instruction-tuned for code-related tasks. **Technical Name:** deepseek-coder-6.7b-instruct
- Mistral AI's 7 billion parameter model that has been instruction-tuned. "v0.1" indicates the initial release. **Technical Name:** mistral-7b-instruct-v0.1

We will first test prompting strategies like Few-Shot Learning, Chain-of-Thought Prompting, General Knowledge Prompting, voting, and self-refinement. By comparing outputs across paraphrased prompts, we aim to assess their impact on response stability. Additionally, we will evaluate reasoning models, such as DeepSeek 7B and LLaMA 3, which have been shown to exhibit stronger logical consistency, to determine whether their structured reasoning capabilities contribute to lower sensitivity to prompt variations. We have not found literature of this type of work so we think it can be promising.

As another of the tools to bring insights into prompt sensitivity, we plan to use adversarial prompting to systematically vary input phrasing while preserving intent. Recent work has framed this brittleness in terms of adversarial robustness, showing that large language models are highly sensitive to carefully crafted prompt variations in attack scenarios (Yang et al [6]). By drawing from this field, we aim to surface inconsistencies in model behavior and use these insights to inform our broader methodology for improving prompt stability.

LLM-as-a-Judge Scoring Method

We used Mistral-7B-Instruct to judge outputs with the following prompt template:

> Compare the following two texts and rate their similarity on a scale from 0.0 to 1.0.
> Return only the number, as a decimal with at least one digit after the point.
>
> **Reference:**
> `{gold}`
>
> **Candidate:**
> `{model output}`
>
> **Similarity Score:**

Finally, we will explore fine tuning techniques, such as the ones presented on Zhou et al [7] to understand the impact of controlled text generation on prompt sensitivity.

Datasets

For our project on improving prompt sensitivity in LLMs, we have collected diverse prompt datasets from three key repositories that offer rich, data-centric resources:

Multi-Prompt LLM Evaluation Dataset: This repository contains an extensive collection of automatically generated and manually validated instruction paraphrases. The dataset is organized into CSV files each corresponding to specific tasks from benchmarks such as LMentry and BBH. These files include detailed metrics such as model accuracies, correctness assessments, and ranking information for each paraphrase. Additionally, JSON files with aggregated results are provided, allowing for statistical analysis of how different prompt phrasings impact model performance. This dataset is introduced in [1].
Link: https://github.com/SLAB-NLP/Multi-Prompt-LLM-Evaluation

POSIX Dataset: In addition to its code, the POSIX repository supplies datasets comprising sets of intent-aligned prompts paired with model-generated responses. Each dataset includes multiple variations of the same query, covering modifications in wording, spelling, and template structure, which enable us to compute key metrics such as response diversity, distribution entropy, semantic coherence, and confidence variance. This dataset forms the backbone of the quantitative analysis presented in [3].
Link: https://github.com/kowndinya-renduchintala/POSIX

RobustAlpacaEval Dataset: This repository provides a benchmark dataset specifically designed to assess worst-case prompt performance. Based on a refined subset of the TinyAlpacaEval benchmark, the dataset features 10 manually refined paraphrases per query. This dataset is presented in [2].
Link: https://github.com/bwcao/RobustAlpacaEval

Collectively, these datasets provide a robust foundation for analyzing the variability in LLM performance across diverse prompt formulations, a critical component for our study on reducing prompt sensitivity.

Results (Intermediate Status)

Our initial implementation focused on evaluating prompt sensitivity using a custom version of the Prompt Sensitivity Index (POSIX) [3]. The official GitHub repository provided limited utility, so we opted for a clean and modular implementation tailored to our setup. Our version closely adheres to the original POSIX computation logic, measuring the log-probability divergence of model responses across paraphrased prompts. We ran experiments on the RobustAlpacaEval dataset (4996 prompt sets) using the `tiiuae/falcon-rw-1b` model. The POSIX pipeline was executed via SLURM array jobs on the HPC cluster, with 46 parallel jobs processing 100 sets each.

We successfully generated baseline responses and calculated POSIX scores for each prompt set. Preliminary analysis shows substantial variability across prompt sets, with some exhibiting high sensitivity to rewording. To address this, we implemented a Chain-of-Thought (CoT) variant by modifying each prompt with the suffix "*Let's think step by step.*"

POSIX Improvements via Chain-of-Thought Prompting

Quantitative results show a clear improvement in output stability. The average POSIX score dropped from 1.25 in the baseline to 0.59 with CoT prompting (a 47% reduction in sensitivity). Figure 1 shows the full distribution of POSIX scores for both the baseline and CoT variants. The distribution for CoT is sharply left-shifted, indicating more consistent and stable responses across paraphrased prompts. CoT also eliminates several extreme outliers present in the baseline.

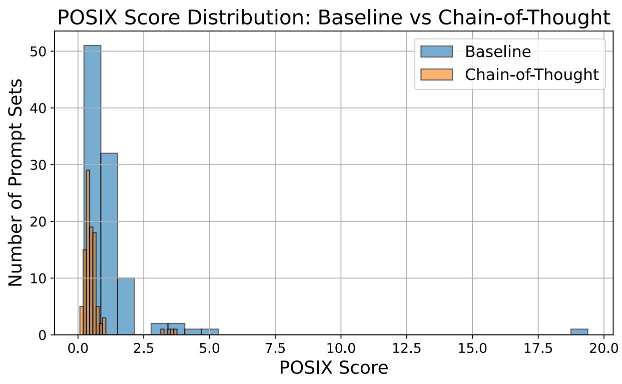

Figure 1. Histogram of POSIX score changes (CoT – Baseline). The majority of values are negative, indicating increased output stability under Chain-of-Thought prompting.

Sensitivity to Dataset Formatting Conventions

A qualitative examination of our prompt sets revealed that even the original dataset formatting (featuring explicit role indicators like `Q:`, `A:`, `question:`, or `answer:`) frequently confused the models. Rather than guiding the model toward cleaner completions, these structural cues often led to repetitive or shallow outputs, particularly in the baseline setting.

Adding Chain-of-Thought (CoT) prompting introduced structural regularity, often shifting responses to a numbered list format. This improved factuality and variety in cases where the prompt was only lightly paraphrased. However, when prompts included deeper noise, such as misspellings (e.g., `lisit okf tean interesting thingd`) or stylized tokens like `Q:::` or `||`, CoT completions remained brittle. In several cases, outputs hallucinated irrelevant entities or defaulted to unrelated QA-style answers (e.g., "What is the name of the file?").

These findings suggest that while CoT prompting increases stability under superficial rewording, it does not provide robustness against noisier or syntactically corrupted inputs. The model's behavior is sensitive not only to meaning but also to formatting patterns that disrupt its internal structure parsing.

Model Output Similarity Evaluation via LLM-as-a-Judge

To complement our POSIX-based sensitivity analysis, we implemented an independent pipeline using an LLM-as-a-Judge approach to evaluate semantic similarity between candidate responses and reference outputs (gold completions). For this, we employed `mistralai/Mistral-7B-Instruct-v0.1` to compute a similarity score between each generated output and its corresponding gold output.

Each score ranged from 0.0 to 1.0 and represented how well a model's response matched the expected semantic content. We evaluated three open-source language models: `GPT2-medium`, `EleutherAI/pythia-410m`, and `Falcon-7b-instruct`, across 40 prompts with 10 paraphrased variants each.

We parallelized this process using SLURM and evaluated results across four jobs, each responsible for a different slice of the data. The similarity scores were then aggregated into a single CSV and visualized through several plots.

Findings:

- **Falcon-7b-instruct** achieved the highest mean similarity score (0.50), significantly outperforming the other two models.
- **Pythia-410m** and **GPT2-medium** trailed behind with average scores of 0.18 and 0.13, respectively.
- Heatmap and boxplot visualizations revealed that while Falcon-7b maintained high scores on most examples, there were still a few cases where similarity dropped to 0, indicating brittleness on certain prompts.
- The lower-performing models showed much more variability and a clustering near zero, suggesting limited robustness to prompt variation.

Implications: This evaluation method offers a complementary lens to metrics like POSIX by directly assessing the semantic similarity of outputs, rather than comparing log-probability distributions. Our findings validate the use of larger, instruction-tuned models like Falcon-7b for tasks requiring higher prompt consistency.

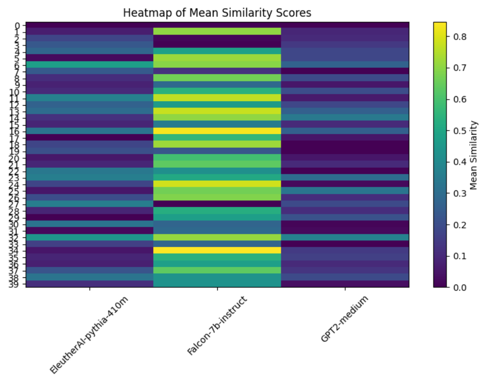

Figure 2. Heatmap of similarity scores across all prompts and models. Falcon-7b consistently shows high similarity (yellow-green bands), while others perform weakly.

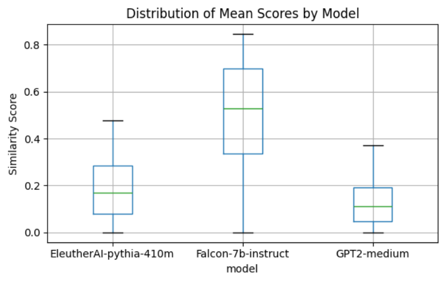

Figure 3. Boxplot showing distribution of similarity scores. Falcon-7b-instruct displays a higher median and broader score range.

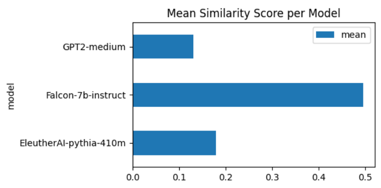

Figure 4. Mean similarity score per model. Falcon-7b-instruct significantly outperforms others.

Discussion and Next Steps

Our POSIX-based evaluation confirms that prompt phrasing significantly influences model behavior. Chain-of-Thought (CoT) prompting reduced sensitivity in many cases, but failures persisted under noisy or unconventional prompt structures.

To build on this, we plan to:

- **Explore iterative self-checking:** Re-prompt the model over k steps, each time including its previous output, to encourage internal consistency. We'll analyze performance-vs-cost trade-offs.
- **Test self-refinement:** Let the model rewrite prompts before answering, using instructions like *"Rephrase the question and solve it."*
- **Evaluate stronger models:** Run the same experiments on Mistral and LLaMA to assess how model scale affects sensitivity.
- **Incorporate additional evaluation metrics:** Alongside POSIX, we plan to experiment with other evaluation methods presented in earlier works to provide a more comprehensive picture of model robustness.
- **Introduce adversarial variants:** Generate controlled corruptions (e.g., typos, misplaced punctuation, odd structure) to explicitly stress-test robustness across methods.

These steps will help us better understand both the limitations of current prompting strategies and which combinations of techniques and models best improve response stability and meaning retention.

Acknowledgments

Here you can thank other persons (advisors, colleagues ...) that contributed to the successful completion of your project.

References

[1] Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. State of what art? a call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949, 2024.

[2] Bowen Cao, Deng Cai, Zhisong Zhang, Yuexian Zou, and Wai Lam. On the worst prompt performance of large language models, 2024.

[3] Anwoy Chatterjee, H S V N S Kowndinya Renduchintala, Sumit Bhatia, and Tanmoy Chakraborty. POSIX: A prompt sensitivity index for large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14550–14565, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[4] Jingming Zhuo, Songyang Zhang, Xinyu Fang, Haodong Duan, Dahua Lin, and Kai Chen. Prosa: Assessing and understanding the prompt sensitivity of llms. *arXiv preprint arXiv:2410.12405*, 2024.

[5] Sheng Lu, Hendrik Schuff, and Iryna Gurevych. How are prompts different in terms of sensitivity?, 2024.

[6] Zeyu Yang, Zhao Meng, Xiaochen Zheng, and Roger Wattenhofer. Assessing adversarial robustness of large language models: An empirical study, 2024.

[7] Wangchunshu Zhou, Yuchen Eleanor Jiang, Ethan Wilcox, Ryan Cotterell, and Mrinmaya Sachan. Controlled text generation with natural language instructions, 2023.