CMSC828J Project Proposal (Fall 2024)
Tricking Generative Language Models

Mike Ledford
Department of Computer Science
University of Maryland, College Park
mledfor@umd.edu

Nate Kadawedduwa
Department of Computer Science
University of Maryland, College Park
nkadawed@terpmail.umd.edu

Abstract

Generative language models (LMs) all rely on some representation of real-world knowledge to create narratives or accomplish reasoning tasks. Whether this is in the form of an explicit knowledge graph or embedded in the model's parameters, its limitations are always tied to the breadth of its training data.

The symptoms of limited implicit knowledge can show up in cases where the distribution of probabilities in the local learned space do not match with their occurrences in real life. For example, a prompt like "Jo put a lasagna in the oven but forgot to turn it on. When she opened the oven door..." may lead the model to continue a story about a burnt lasagna rather than a cold, un-cooked one because opening oven doors is more frequently linked to burnt food in the training data regardless of the physical reactions.

This issue often arises because large language models (LLMs) like GPT are trained to predict the next word based on data patterns rather than a genuine understanding of the real world. A primary contributor to this problem is reporting bias, which systematically distorts the frequency of certain events, outcomes, and properties in text (Gordon and Van Durme, 2013). Such misunderstandings were far more common in GPT-3 than they are in GPT-4 (Achiam et al., 2023). With vastly more parameters and a significantly larger training dataset, modern LLMs are now much more resilient to being misled. However, these state-of-the-art models still suffer from inherent biases in training data leading to ambiguous, erroneous, and biased outputs (Raiaan et al., 2024).

Motivation

Conditional language models like OpenAI's GPT, Meta's LLaMA, and Google's BERT generate text based on specific inputs, or "conditions." By aligning their output with a given input, these models can produce more controlled and relevant responses. Identifying consistent ways to mislead these language models would be valuable, as it could help determine whether errors arise from issues such as coreference resolution or contextual ambiguity at lexical, syntactic, or semantic levels, along with other underlying phenomena. Consequently, these insights could inform design and training strategies to enhance the robustness of language models as they become increasingly versatile and widely used across NLP applications.

Systematic Approach

To systematically find ways to lead conditional LLMs astray, we propose a four phase approach: problem identification, trend analysis, concept verification, and solution experimentation.

Problem Identification

In this phase of the experiments, we will systematically explore prompts and tests under controlled conditions to identify potential vulnerabilities in the LM. Our strategy is to create a set of 10-15 challenging examples either by creating prompts similar to "Jo put a lasagna in the oven..." or pulling examples from a myriad of larger datasets known to be difficult to state-of-the-art models (e.g., WinoGrande, PIQA, etc.). An example prompt would like Table 1. Ultimately, our emphasis will be to design a set of prompts that may be difficult for LMs to interpret due to suspected reporting biases or necessary background knowledge.

Our primary focus will be the LlaMa 3 model (8 billion parameters) from Meta due to its static weights, which simplifies consistent testing compared to more dynamic models like GPT or Claude. For each prompt, we will instruct the model to perform each task in a "zero-shot" setup, without any examples provided. In addition, we will extend

testing to include GPT-4, Claude 3.5 Sonnet, and Sonar Huge (based on LlaMa 3.1), allowing us to explore any interesting comparisons across models.

Trend Analysis

In this phase, we aim to analyze the results from the prompts used during the problem identification phase, focusing on pinpointing categories where language models (e.g., GPT-4, Claude 3.5 Sonnet, and Sonar Huge) generated misleading or incorrect responses. Drawing on relevant literature discussed in class, these categories may include ambiguous prompts, uncommon knowledge requirements, deficits in commonsense reasoning within specific domains (e.g., physical, social, cultural), or challenges with hypotheticals and counterfactuals. Our objective is to identify and determine whether there are syntactical patterns in these prompts that consistently pose difficulties for all models or for specific ones. To do this, we will compare performance between examples and models using truth tables, similar to Table 1, and qualitative analysis.

Model	Output	Tricked?	Notes
GPT-4o mini	The lawyer asked the witness a question, but he was reluctant to repeat it . Who was reluctant to repeat the question?	Yes	

Table 1: Tricking GPT-4o mini model with the prompt *"The lawyer asked the witness a question, but he was reluctant to repeat it . Who was reluctant to repeat the question?"* This example was provided by Winograd Schema (Levesque et al., 2012)

Concept Verification

In order to assure that the concepts we found above are separate of any model specific artifacts or situational cues (and find a pipeline to easily generate more examples), we test if a model is more accurate on examples which it creates based on our given one.

By combining the most difficult properties from our previous step, we create a single example that would be difficult for the models and ask LLaMa to generate 10 examples similar to it. If the model performs better on the model-generated set then we can go on to change parts of the examples which many contain artifacts. If, after changing components like the setting, subjects, or actions, the model performs poorly, then we can feel more confident about our initial example (It does not contain such cues) and we would also use this procedure as a basis for generate multiple examples for training large models in the future.

Solution Experimentation

In this final phase, we will converge all the findings to select a difficult domain (physical, social, temporal, etc.) and create a set of 10 examples that can be used for in-context learning. According to relevant literature, a minimum of 10 samples is needed for effective ICL (Chen et al., 2023) so our 10 examples will be derived from combinations of all the properties which were found to be particularly difficult.

To test the effectiveness of ICL, we will measure performance of the LLaMa model compared to the control set of zero-shot examples from the first step. We generally expect that the performance of the model after ICL will be significantly better than the zero-shot answers. Accuracy will be measured qualitatively or in a binary manner depending on the format of the example (i.e. Winograd or entailment).

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*.

Jiuhai Chen, Lichang Chen, Chen Zhu, and Tianyi Zhou. 2023. How many demonstrations do you need for in-context learning? *Preprint*, arXiv:2303.08119.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, AKBC '13, page 25–30, New York, NY, USA. Association for Computing Machinery.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Mohaimenul Azam Khan Raiaan, Md. Saddam Hossain Mukta, Kaniz Fatema, Nur Mohammad Fahad, Sadman Sakib, Most Marufatul Jannat Mim, Jubaer Ahmad, Mohammed Eunus Ali, and Sami Azam. 2024. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, 12:26839–26874.