Project Proposal: Prompt Recovery in LLMs

Advith Chegu ac1771 | **Will Chandler** wcc44 | **Hassan Raji** hr322 | **Rohan Tripathi** rt767 | **Vignesh Krishnan** vk505

Abstract

With the recent development and popularity of Large Language Models the ability to generate and manipulate text has been expanded to many people. This has led to the proliferation of LLM generated and manipulated text. This leads to the problem of who wrote these texts or how were they manipulated by LLMs. We have taken this general issue and have reduced it to the problem of prompt recovery.

1 Problem Statement

Can we identify how a text has been manipulated by an LLM. Given an input text we will have an LLM transform this text into a specified "style." Our goal is to recover the specific prompt used to transform this text.

1.1 Relation to AI Principles

In developing and deploying Prompt Recovery in LLMs, it is imperative to adhere to a comprehensive set of AI principles to ensure ethical and effective operation:

- **Transparency:** Transparency is paramount as we seek to understand how the model identifies prompt styles, allowing for insight into its decision-making processes.
- **Fairness:** We must also prioritize fairness to mitigate biases in the model's predictions, ensuring equitable treatment across prompt styles.
- **Avoid creating or reinforcing unfair bias:** We avoid creating or reinforcing unfair bias, making sure that the model's capabilities serve all individuals equally.
- **Accountability:** Holding the model responsible for reliably and safely identifying prompt styles with accuracy is crucial for ensuring its performance.
- **Awareness:** Awareness in this project involves understanding the model's ability to accurately identify and replicate different writing styles based on specified prompts, initially through fixed user prompts and later through scenarios where the prompts are unknown.
- **Explainability:** Our ability to clarify how our model identifies different prompt styles is crucial, highlighting the significance of Explainability in this project.
- **Value alignment:** Ensuring that the model's goals are in line with human values and objectives, which encompass the precise identification of prompt styles.

2 Objectives

Specific problem and relation to general AI principles as a topic that you propose
The main objectives of our project are:

- To create models that successfully recover a prompt given the generated output with reasonable accuracy.

Preprint. Under review.

- To compare the performance of different models and speculate over why one performs better than the other.
- To understand state-of-the-art works in the field of Large Language Models and implement the facets applicable to writing style analysis.

3 Methods

While our task is similar other NLP tasks such as semantic analysis, LLM prompt recovery is not as straightforward. This is due to the fact that we have **two** inputs, one is the block of text before going through the LLM prompt, and one block after the LLM. Therefore, unlike a task which requires looking at a tweet and trying to find certain buzzwords, it's not clear which words to weight the most in the new document compared to the original document when trying to figure out the the LLM prompt.

3.1 Relevant Methods

Text as a medium is non-intuitive to work with when it comes to ML tasks, it's often the case we need to learn some numerical representation of the text in order to feed it through an ML model. Below are some common approaches.

- *Bag of Words* - A Bag of Words is a tally of the number of times a particular word appears in some text. The intuition behind this approach is to understand the frequency of the words in some text such that we can classify them into different categories. For example, if the word car or bus appears multiple times throughout the text, we can safely assume that the text relates to vehicles. Harris (1954)
- *TF-IDF* - The problem of unimportant words such as "and" or "it" becomes a evident when classifying/summarizing text since we're not sure how important these are to the input text. To tackle this, early NLP used the term-frequency-inverse document frequency to figure out the weightage of certain words. All this did was to compare all word frequencies between a corpus of several different input texts to figure out how important a word is relevant to other texts. If the word frequently appears in all the other documents, it is safe to assume that it might not be so important to the particular document we are looking at. Sparck Jones (1988)
- *Continuous Bag-of-Words* - Enter deep learning. A continuous bag of words model relies on the idea of a context window, where given some surrounding words, we attempt to output a next word or some word that is most likely to be associated with the input. The associated neural network architecture is simple, we start with a random encoding for all the words in the corpus and feed the context window through our neural network with our result being the learned representation of the goal word. Mikolov *et al.* (2013b)

  There's a few drawbacks to this approach, two big ones are the lack of order in the context window as well as long range dependencies. CBOW does a lousy job of learning relationships that are outside of the context window and thus it's easy to realize that long range dependencies between words suffer quite a bit.
- *Skip-Gram* - This approach is similar to CBOW except the opposite. Confused? Well the skip-gram just takes in the context of words and aims to predict the subject word. This might be more similar to our goal since we have some document and want to learn a representation of it. Mikolov *et al.* (2013a)
- *Recurrent Neural Networks* - Remember the problem of learning meanings of words in the order that they are presented? Recurrent neural networks aim to remedy that problem. In simple terms, RNNs use the output of previous timesteps as the input of the future ones. This helps the model "remember" patterns in sequences that came before the current word and it uses this memory to inform future predictions. Schmidt (2019)
- *LSTMs* - An LSTM is a type of RNN that uses different gates to form something called "attention" in which the model is trained to learn which information in some sequence is important and which is not. Therefore, we're able to retain more of the important information and remedy the gradient vanishing problem present in RNNs. Gers *et al.* (1999)

- *Transformers* - Lastly transformers are an improvement over LSTMs which involve "self attention". Unlike taking inputs as a sequence, transformers relate different positions of a sequence against each other to learn relationships between all the words in a sentence, this allows transformers to be trained in paralell. Vaswani *et al.* (2023)

3.2 Data Generation

First, a hundred paragraphs having random data are generated by hitting the GPT API. There is a fixed list of 100 people, based on whom the paragraphs are rewritten. Each of the paragraphs is rewritten in the style of all 100 people. This is also done by firing requests using the GPT API. The prompt used here is to rewrite the text in the style of a person. In this way, the dataset having an original text column, a rewritten text column and a text prompt column is generated.

3.3 Input Parameters

The input to our model will be a single system prompt LLM output along with the user prompt LLM outputs. The goal of our model is to identify the user prompts associated with each LLM output.

3.4 Model Creation

- *Bag of Words + TF-IDF + Neural Network* - We have a set of documents that have been fed into an LLM with various user prompts, and we have the original (system prompt) one as well. All we need to do is run through the documents and figure out which words appear more often in certain documents when compared to the others from the user prompts. Then feed the top N term representations into a neural network to learn the mappings between the popular encodings for each user prompt.
- *spaCy + Set Difference + Neural Network* - We encode our inputs using spaCy embeddings and then take the difference of the stylized output with the original output and train a neural network to map this difference with the style. Honnibal and Montani (2017)
- *LSTM + Set Difference* - We just use a transformer to learn the representations from scratch such that when the difference between document representations is taken, it maximizes the probability of mapping to the correct user prompt. We would create our own loss function in this scenario, but would likely use some level of transfer learning for the LSTM.

3.5 Model Evaluation

We evaluate the model on how well it is able to identify the style of the writing based on the two inputs. For example, if our model gets two documents, one about horses and the other about horses written in the style of Shakespeare, we want to build a model that correctly identifies the style of Shakespeare. The model can simply be evaluated on an accuracy metric of how well it is able to identify the style. We can also use classic metrics such as F1 score, precision, and recall.

4 Timeline

We have divided our project into 4 phases based on the expected workload. An overall tentative timeline is provided below:

4.1 flowchart

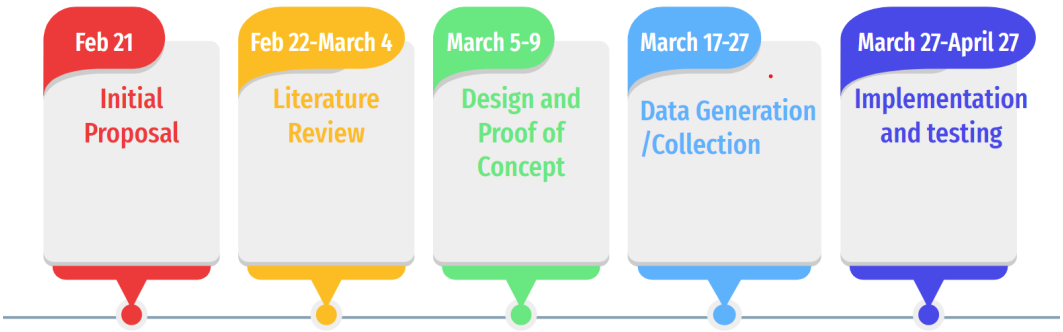

Figure 1: Time Line for different phases of project

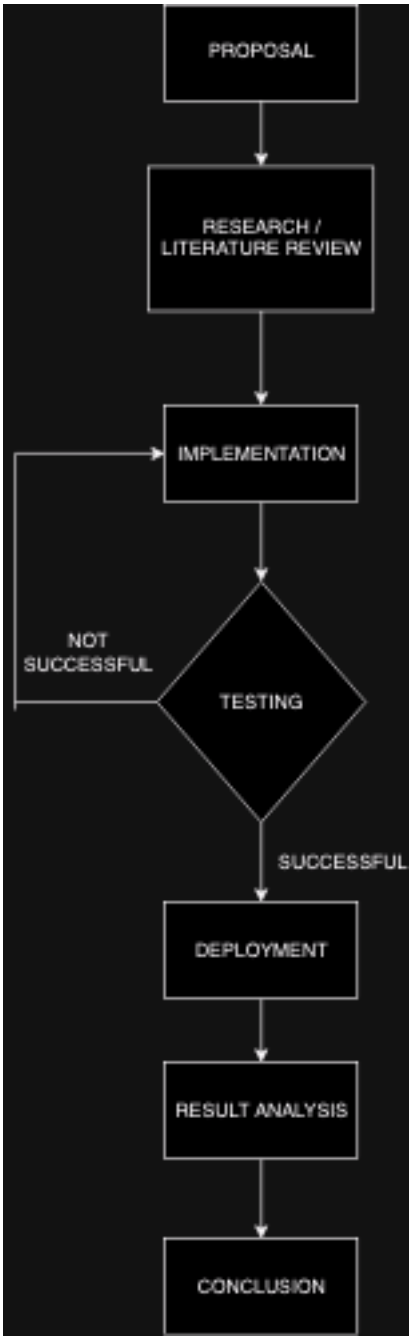

Figure 2: FFlowchart of work

5 Expected Conclusions

The expected outcomes of the project are:

- Comparative analysis of different model architectures will be conducted to highlight their respective performances.
- The effectiveness of employing a predetermined set of user prompts is evaluated under standardized testing conditions, ensuring consistency and comparability across different experiments and models. In this way, we can control variables that might otherwise introduce bias or variability into the evaluation process.
- Conclusions will be drawn based on the model's effectiveness in discerning the stylistic differences between two given inputs, along with the consideration of classic evaluation metrics
- The efficacy of the pre-established threshold in determining the correctness of model predictions will be assessed.
- Efforts will be made to tackle the challenges of accurately recreating the original user prompts from the outputs generated by the models.

6 Roles Breakdown

Project Manager and Implementer - Rohan
Implementer - Advith
Co-implementer - Vignesh
Plotter - Hassan
Final Report Writer - Will

References

Gers, F., Schmidhuber, J., and Cummins, F. (1999). Learning to forget: continual prediction with lstm. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, volume 2, pages 850–855 vol.2.

Harris, Z. S. (1954). Distributional structure. *WORD*, **10**(2-3), 146–162.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013a). Distributed representations of words and phrases and their compositionality.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013b). Efficient estimation of word representations in vector space.

Schmidt, R. M. (2019). Recurrent neural networks (rnns): A gentle introduction and overview.

Sparck Jones, K. (1988). *A statistical interpretation of term specificity and its application in retrieval*, page 132–142. Taylor Graham Publishing, GBR.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention is all you need.