Conversational Drive Through Restaurant Dataset Prompt Engineering with GPT & Gemini

Aaron Fletcher & Boxuan Shaun

December 30, 2023

1 Introduction

Prompt engineering is the process of modifying the input of a model to achieve optimal results. In the case of large language models (LLMs), the prompt is a textual sequence indicating a task the model should perform, such as "In computer science, what is priority inversion?" - see Figure 1. Given this initial prompt, the information recovered from the LLMs will generally complete the task. However, it may not be in line with the intended audience. Engineering this prompt with modifiers can ensure that the response matches this target audience, such as "Respond in a way that a 5-year-old would understand: In computer science, explain what priority inversion is?" - see Figure 2.

An emerging property of LLMS enables prompt engineering: in-context learning (ICL). ICL is the ability of a language model to make inferences based on information provided with the input, typically preceding or following it. In the above example, given the proceeding prompt "Respond in a way that a 5-year-old would understand," the LLM would not update its parameters and respond using its pre-trained model but the context to shape the response.

Recently, Google released a new generative model, reportedly superior to the GPT-4 architecture, the Gemini model, in November 2023. Little research exists to evaluate the effect of prompt engineering on Gemini architecture.

This research aims to understand whether prompt engineering improves the effects of downstream tasks presented within this mini-project and additionally establish some prompt-engineering performance comparison of GPT-4 to the Gemini model.

2 Related Works

3 Types of prompt engineering

1. One-Shot Prompts
2. Few-Shot Prompting

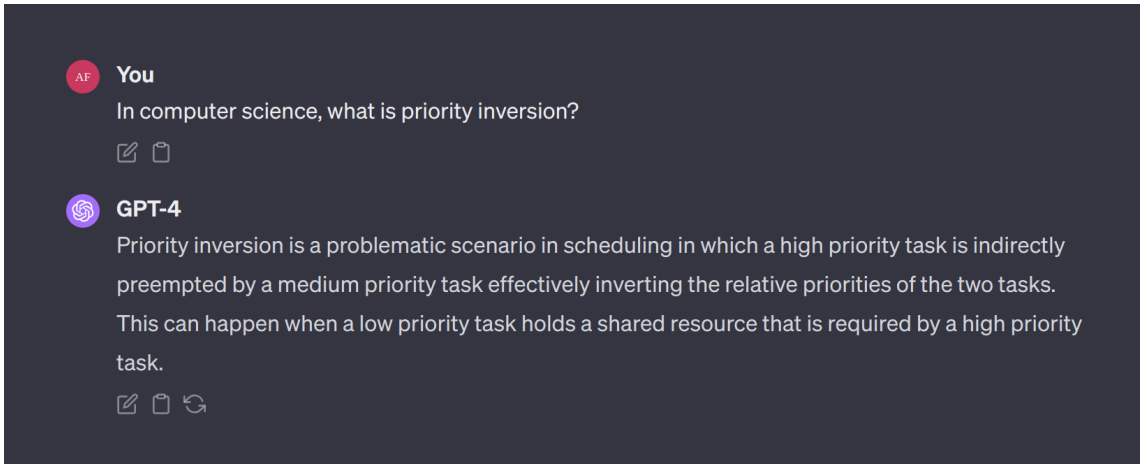

Figure 1: Chat GPT asked a standard computer science question, using GPT 4.5

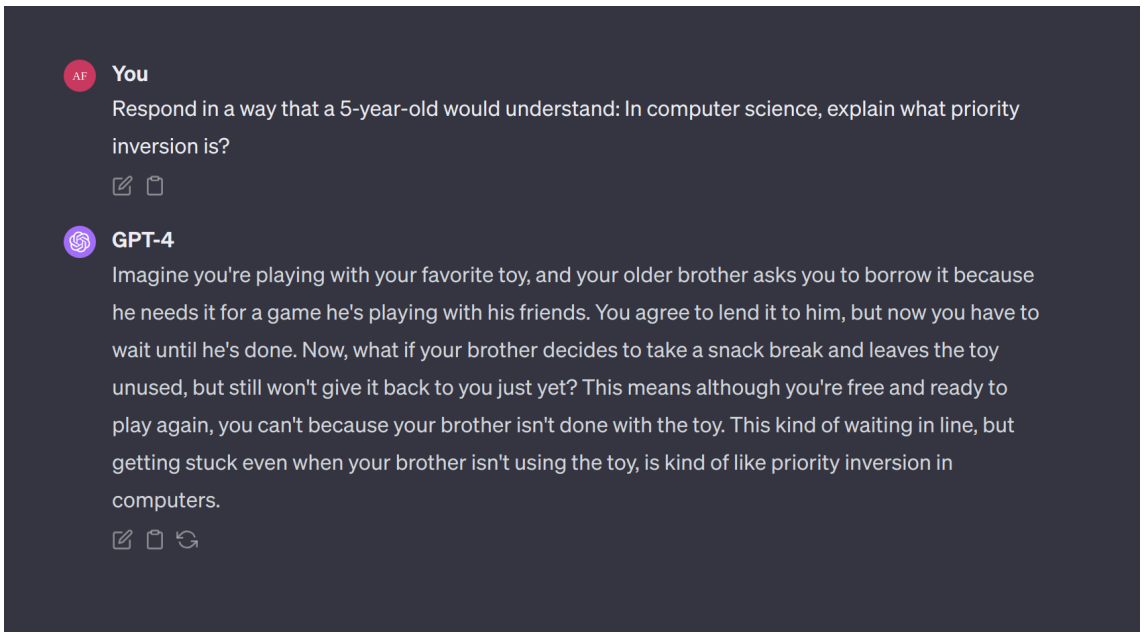

Figure 2: Chat GPT asked a standard computer science question, with prompt engineering, using GPT 4.5

   (a) Contextual prompting is an example of these

3. Role Prompting

   (a) Translating the transcript into a finalised order would be an example of this

4. Chain of thought prompting

4 Experimental Approach

At its core, this project aims to use audio data to represent ordered items accurately. The experimental design generates three forms of data: the scenario audio, a human-validated transcribed representation of the scenario, and the finalised ordered items.

In this context, prompt engineering represents a type of hyperparameter optimisation.

Prompt engineering can occur any time data are being passed to a model. Audio prompting is enabled by using OpenAI's whisper model's API, which allows textual information to be supplied to the model to improve its performance. Notably, only 224 tokens can be provided to the model, with any preceding tokens ignored. Textual prompt engineering can be achieved by modifying the textual input through prepending, extending, or modifying the input.

Due to API limitations, the audio files were pre-processed into 25 MB or smaller files and fed into the transcription API.

4.1 Audio (GPT-4 only): Prompting affects the transcribed word error rate

The two experimental conditions will use standardised settings to transcribe the scenario. As a baseline, no textual prompt will be supplied to the whisper model, and in another experimental condition, a textual prompt will be provided with the scenario data. The textual prompt for the whisper API will consist of any neologistic food items from the menu (i.e., food menu items that have been created for this scenario, such as "Mambo Combo" or "ShefBurger").

The prompt will be formed from the frequency of food entity neologisms within the overall transcript corpora. They will be ordered from lowest to highest, with the lowest frequency neologisms up to the token limit of 224 being provided as the prompt to all transcriptions in this experimental condition.

These conditions will then be compared to human-validated ground-truth transcripts of the audio scenarios, with a word error rate used to measure either approach's effectiveness.

This approach determines if model performance can be improved by selecting this prompt list without retraining the entire model. The potential benefit of this approach is that energy consumption could be reduced using prompt engineering to tailor the model to different menu items in restaurants.

Notably, as of writing, Gemini API does not enable audio transcriptions.

4.2 Audio (GPT-4 only): The textual prompt for the longer-term dependencies reduce WER

Type of Prompt Engineering Tested: One/Few Shot Prompting

Due to the API limitations, which allow only 25 MB or smaller files, the previous longer-term contextual dependencies are lost.

Prompting with the conversation's previous transcription will hopefully improve the word error rate.

4.3 Text (GPT-4 & Gemini): Generating finalised orders from human-validated transcripts.

Type of prompt engineering tested: Role Prompting

The two experiential conditions for this will use standardised settings to translate the scenario into finalised orders. The transcript of the text will be prepended with "Generate a finalised order for the following transcript: TRANSCRIPT." Both models will be compared with the finalised, human-verified gold orders generated through data generation.

It should be noted that the LLM output may not precisely replicate the desired outcome. The output of the generated text will be standardised into a table containing the quantity and the item ordered that matches the expected menu structure to enable comparison. Then, they will be ordered alphabetically by the item name.

The word error rate could then be calculated from the gold standard formatted lists on the gold standard.

4.4 Text (GPT-4 & Gemini): Context effect on WER.

Type of Prompt Engineering Tested: One/Few Shot Prompting

The two experimental conditions for this will use the standardised settings to transcribe the scenario; however, each model request (GPT-4 and Gemini's) will be prepended with text to provide context to the LLM about the data that will be received. This text will be "The following text contains an order from a restaurant drive-through order. Generate a finalised order for the following transcript: TRANSCRIPT.

It should be noted that the LLM output may not precisely replicate the desired outcome. The output of the generated text will be standardised into a table containing the quantity and the item ordered that matches the expected menu structure to enable comparison. Then, they will be ordered alphabetically by the item name.

The word error rate could then be calculated from the gold standard formatted lists on the gold standard.

4.5 Text (GPT-4 & Gemini): Delineation effect on WER

Type of Prompt Engineering Tested: One/Few Shot Prompting

https://user-images.githubusercontent.com/89960/233509865-4f3e7265-6645-4d43-8644-ecac5c0ca4a7.png

4.6 Text (GPT-4 & Gemini): Specifying the desired output of the text

Supply the model with an exact output; no post-processing is required.

5 Evaluation Metrics

-	**Whisper**	**Gemini**	**GPT**
Version	whisper-1	Gemini Pro	gpt-4-1106-preview
Size/Token Limit	25mb	32000 Tokens	128000 Tokens
Training Data	-	-	April 2023
Tokenizer	-	-	April 2023
API Endpoint	https://api.lemonfox.ai/v1/audio/transcriptions	-	-

Table 1: Model References used for experiments