Econimate

Midpoint Review Memo

Jason Yan
Kelly Hong
Victoria Ye
Yukta Butala
Zhu Yuchen

Project Description & Motivation

In our collaborative brainstorming session, the team converged on a unified goal: leveraging the capabilities of Large Language Models (LLMs) to develop an educational tool. Recognizing the potentially overwhelming nature of economics, we focus on developing "***Econimate***". This innovative interface seeks to address the economics-related queries through AI tutoring, catering to students, professionals, or anyone curious about the world of economics. Through the application of Econimate, we developers strive to make complex concepts more accessible through engaging and informative video tutoring.

Our cutting-edge AI video generator incorporates PaLM2 API, an innovative solution that transforms the users' questions into videos seamlessly. Users can simply input their economics-related query, and our advanced model will translate the text into a dynamic video along with an immersive and personalized visual learning experience.

Project Goals

Completed:

- Enhance the outputs from LLMs (PaLM2 API) by refining text and code outputs through prompt engineering.
- Transform textual information into audio and code into user-friendly visuals, combining them for enhanced comprehension.
- Generate videos by integrating the refined audio and visuals for an easy-to-follow output.

Next Steps:

- Integrate the GPT-3.5 LLM interface by replacing the PaLM2 API to improve overall performance and content quality.
- Further fine-tune the model through training with more examples, exploring and implementing advanced prompt engineering techniques.
- Develop a user-friendly interface for direct interaction with the LLM model, eliminating the need for users to independently execute code for video output.
- (Possibly) Create transitions in content – produce videos with multiple slides and audio.

Implementation Progress

As our group embarks on this project, our implementation progress primarily documents our work in the Google Hackathon Project and integration attempts with OpenAI.

Prompt Engineering

Our prompt engineering strategy involves creating prompts that consider simplicity, detail, and tone variation. We have also employed certain techniques, such as asking the AI to think "step by step" and to engage in "chain of thought" analysis (Gao, 2022). Using the PaLM2 API, we were able to obtain the best results for a simpler prompt with direct instructions. However, when using GPT-3.5, we get the best results for a prompt which is more complex, especially with the utilization of "step by step" technique. We attribute this difference to the limited processing power of PaLM2 API, which makes it harder to take in more complex instructions. The specificities of prompts utilized, together with the analysis, can be found in *Appendix A*.

AI Training

AI training under PaLM2 is conducted using Makersuite's data training function, where we input examples into the system. We utilized a total of 8200 tokens and 30 examples for the final training of our model. We also attempted to tune our model, but success is limited as we do not have sufficient examples for our model to learn (the recommended number is 50 examples). *Appendix B* shows how our model transforms across training iterations and its success with test cases.

Code Implementation & Synthesis

The implementation of our project's code is segmented into various components: segmenting results from the large language model, converting text to audio, transforming code into images, and ultimately merging the image and audio to create a video. During the segmentation of the LLM results, we faced a challenge: PaLM2, the model we used, failed to generate results in a uniform format after its training. Our initial strategy, which involved using a string split function on certain keywords to segment the results, was not successful because of this variability. To resolve this, we implemented a new method where we re-input the output from LLM back into itself. This process enabled us to isolate the code intended for image generation and the text portion that explains the concept or question posed to the model accurately.

After segmentation, we process the text using our text-to-audio conversion tool, while the code is input into our code-to-image generation function. The text-to-audio part is handled using the Google Text-to-Speech Python library. For generating images, we write the code into a temporary Python file, applying modifications that enable it to save the image to a specified local path, and then execute this code to create the image.

To create the final video, we employ both MoviePy and OpenCV2 libraries to combine the image and audio files. While the MoviePy library theoretically has the capability to perform this task alone, we kept encountering an error in its function that exports the video to a local folder – an issue we couldn't resolve regardless of the parameters used. To overcome this, we utilized OpenCV2 for the final export step of the video. *Appendix C* documents our implementation flowchart.

Existing Challenges

In this section, we address the current challenges of our project and outline plans for further improvement in the second half of this mini. During our implementation process, we encountered three major problems:

Segmentation of User Input

Our primary challenge revolved around segmenting the user input for the PaLM2 chatbot. To generate a text explanation for an economic concept, along with Python code and a complementary graph, we needed to divide the task into distinct parts. We explored various approaches, including using an NLP tokenizer to extract keywords and formatting the chatbot input. Additionally, we attempted to fine-tune the model to produce output in a specific format.

Initially, we used the spaCy library in Python to extract keywords from user input by extracting nouns. However, this approach had limitations, as nouns didn't necessarily translate to keywords, potentially leading to incorrect information. Furthermore, even with careful formatting, we couldn't ensure that the chatbot's output would consistently match our desired format, making it challenging to separate the explanation and the code.

Our subsequent approach involved training the generative AI model, PaLM2, to better understand context and generate answers in a specific format. Unfortunately, the model did not consistently produce output in the desired format, posing similar challenges in segmenting the response into audio and graph components.

Accuracy of PaLM2 Model Output

The second challenge centered on the accuracy of the PaLM2 model's output. It frequently generated incorrect Python code, resulting in the failure of image generation. This proved problematic as we needed accurate images to combine with audio for video production, leading to errors. Furthermore, PaLM2 struggled to adhere to specific instructions to produce responses using the specified format, particularly after fine-tuning our updating the model responses. We postulate that this is due to text-bison's limited processing ability.

Access Issues with Fine-Tuned Palm2 Model

Our final challenge was related to accessing the fine-tuned PaLM2 model. We encountered intermittent connectivity issues with the model, causing significant errors during our implementation.

Future Plans

For our final project, we are considering adopting a GPT model. Initial testing indicates that a GPT model provides better answers more consistently and excels at extracting key economic concepts from user input while adhering to our intended formatting. We anticipate that most of our challenges will be resolved by implementing this different generative model. Beyond that, we might try a variety of different prompting methods as well, including zero-shot adaptive prompting (Wan and Sun, 2023) in the training of the model.

Work Cited

Gao, A. (2023, July 17). *Prompt engineering for large language models*. SSRN. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4504303

Thursday, N. 02, & Large Language ModelsMachine LearningNatural Language Understanding. (2023, November 2). *Zero-shot adaptive prompting of large language models*. – Google Research Blog. https://blog.research.google/2023/11/zero-shot-adaptive-prompting-of-large.html

Appendix A - Prompts Engineering Efficiency

<table>
<tr><td colspan="2"><b>Prompt 1:</b>

In the following conversation, I will mention concepts. Firstly, identify if the concepts mentioned are Economics concepts. If they are not Economic concepts, respond with: “This is not an Economic concept. Ask me anything about Economics instead!”

If there are Economic concepts present, respond in two sections. In the first section, produce Python codes that code the graph/ table/ matrix/ equation which explains the concept. Name this section [Figure]. In the second section, explain the concept with references to the figures as well as real-life scenarios, name this section [Explanation].</td></tr>
<tr><td><b>PaLM2 API:</b> Failure to follow format most of the time. Completely ignores the command when unable to identify economic concepts.</td><td><b>GPT-3.5:</b> Very well structured response that completely follows the format given.</td></tr>
</table>

<table>
<tr><td colspan="2"><b>Prompt 2:</b>

In the following conversation, explain to me concepts in two sections, [Figure] and [Explanation]. In the [Figure] section, include a Python code that codes several figures (graphs/ tables/ matrix/ equations) which explain the concept. In the [Explanation] section, explain the concept with references to the figures produced as well as real-life scenarios. If no Economic concepts are mentioned, respond with “This is not an Economic concept. Ask me anything about Economics instead!”.</td></tr>
<tr><td><b>PaLM2 API:</b> Failure to follow format at times by not describing the figures produced. Mostly able to identify concepts’ nature.</td><td><b>GPT-3.5:</b> Gives too structured response that explicitly states [Reference to figure] section and [Real life scenario] despite not asked.</td></tr>
</table>

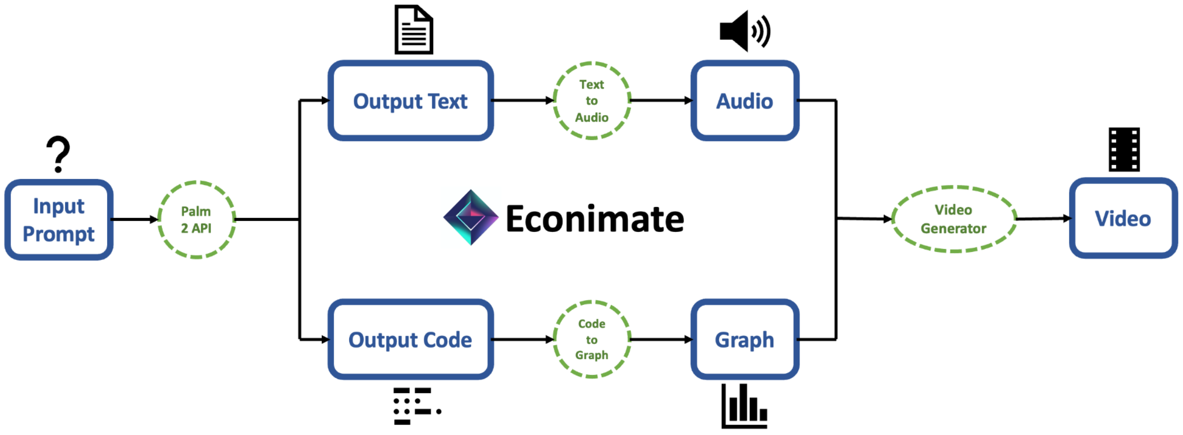
<table>
<tr><td colspan="2"><b>Prompt 3:</b>

In the following conversation, explain the following concepts in two sections, [Figure] and [Explanation]. In the [Figure] section, return me a Python code that codes a figure (graphs/ tables/ matrix/ equations) which explains the concept. In the [Explanation] section, explain the concept with references to the figure produced. If the concept is not an Economics concept, respond with “This is not an Economic concept. Ask me anything about Economics instead!”.</td></tr>
<tr><td><b>PaLM2 API:</b> Able to follow instructure most of the time.</td><td><b>GPT-3.5:</b> Able to follow instructions all the time.</td></tr>
</table>

Appendix B - AI Model Versions

Model Generation	Prompt Ver.	Number of Examples	Example Quality	Prompt Type	Test Success
Gen 1	1	10	Self-produced	Chat	3/9
Gen 2	2	20	Chat-GPT	Chat	5/9
Gen 3	2	25	Bard	Data	6/9
Gen 4	3	30	Bard	Data	9/9
Gen 5	3	30	Bard + Formatted	Data	9/9

Appendix C - Implementation Flowchart


Input Prompt
Palm 2 API
Output Text
Text to Audio
Audio
Econimate
Output Code
Code to Graph
Graph
Video Generator
Video