ChatGPT Notes

<u>July 3, 2023</u>

Pricing

- OpenAI API is based on token use.
- Input + output tokens are taken into account.
- Eg. Input is 10 words (10 tokens), response is 30 words (30 tokens), pricing for that api call would be: 40 tokens * **price per token**
- Need to create an account and attach billing details to use the API.

Capabilities

- Give intelligent answers based on certain given information. By giving it a list of information, for example a sheet of rules and relevant phone numbers, the bot can answer FAQs and also refer the user to contact info.

- Function Calling
    - Use custom functions that are called
    - Eg. If you have functions that can access data from a database, passing it into ChatGPT will intelligently get the required arguments for the function. ChatGPT gives you the arguments in a json output, which then can be passed into the function in the code.
    - With this, with a database, you can access the data of an order by passing, for example, your orderID and userID into the chatbot.
    - Could possibly make alerts based on user input.

Potential Issues

- The pricing is based on usage, malicious attack and spam could increase costs
    - Limit token usage per user
- Responses can be unpredictable, and can vary with the same input.
- Inputting a large document (such as information and rules) each time the chatbot is initiated can rack up costs over time as many tokens will have to be used.
- Model might change over time.
- Rate limits: Only 3 requests can be make a minute
    - Put a delay for putting things in chat, once every 15 - 20 seconds

TO-DO

- Setup an account to start using the API
- Discuss on what kind of integration are we looking for for the tech support chatbot
    - Database integration?
- Discuss what features are needed.
- What is the existing support base?

July 4, 2023

The Tool

Create an ai based model that can take data from an existing knowledge base and make inferences/analysis based on a question asked by the user. This is not meant for the customer, but more to aid in technical support as it takes a long time to find relevant information in the documents.

Problems

The OpenAI API has many restrictions. There is a per minute restriction of 3 API calls, and also a max token per call of 4000, which is not very much. This will also rack up costs because the pricing is based on tokens inputted. Inputting tokens would have to be done each time the tool is used.

OpenAI only takes in strings, so conversion from pdf/docx/txt into a string has to be done. String limits would be a problem, but it could be converted into a list of strings.

Looking at the document, there seems to be a lot of images. This means that it cannot be passed into a language model like ChatGPT, and there would be a lot of missing and incomplete information if only language is used.

This is also a problem for tabular data. It is hard to parse tabular data with an LLM like ChatGPT as it can't visualize the positioning of the rows and columns with just language alone, eg.
(https://en.wikipedia.org/wiki/List_of_WLAN_channels#mw-head)
(https://documentation.meraki.com/MR/Radio_Settings/Channel_Bonding)

HTML files provided do not have images, and are missing stuff like styling and sources.

Potential Solutions

Chunk the data into smaller sizes and use only relevant info based on the queries.
Use another API such as elasticsearch.
Manually put in information based on the customer's needs.
https://bdtechtalks.com/2023/05/01/customize-chatgpt-llm-embeddings/
https://www.docsumo.com/blog/pdf-reading-with-gpt4
https://openai.com/customer-stories/morgan-stanley

Can't compare 2 documents
Instead of coming up with solutions, find relevant documents.

TO-DO

Work on making the wikipedia document usable by chatgpt. Make it so that the images can be opened when needed. Larger documents.

July 5, 2023

ManualGPT Steps

- Format manuals into pdf/txt that makes it easily parseable by PyPDF2, ChatGPT and the OpenAI embedding API.
    - This requires more work for manuals with photos and tables.
    - The manuals have to be formatted logically, for example sectioning of different sections, and adding filepaths to relevant images.
    - This means converting everything into a string, and storing images in relevant directories.
- Use OpenAI embeddings to convert all manual strings into a vector format database.
- Store this in a hashable format or SQL database.
- Based on the user's query, convert the relevant contents of the inquiry into a vector with the same OpenAI embedding model.
- Use this query embedding to retrieve the most similar documents by comparing the query embedding vector and the document embedding vector.
    - This will be done using cosine similarity: Using k most similar documents, or having a similarity threshold, or both.

- Feed the relevant documents into ChatGPT to enable it to answer questions based on the manuals.

With this workflow, we can get the most relevant data from documents, while also minimizing the token usage.

Two Key Parts

- Make a document recommender based on an input
    - Requires OpenAI embedding model or word2vec
    - Requires making the document vector database
    - Figure out a quick way to access these vectors quickly
    - Need to get as many manuals as possible from Alex, along with example questions for testing.
    - This can be tuned to maximize relevant knowledge and minimize token use.

- Make the chatbot that takes in the documents.
    - Already done for the most part
    - Needs fine tuning to elicit more useful and desired responses (Prompt Engineering)

Embeddings

- Experimented with GPT embeddings and tested it with PCA
- Some clustering of visually similar things can be seen when using countries as the text.
- Countries with the name “republic of” would be clustered together
- No other relation was apparent.
- Further testing needs to be done to see if it can be relied on to find suitable manuals.

TO-DO

- Get Manuals from Alex
- Talk to him about my idea
- Actually do it
- Figure out a way to deploy both of these into one whole package
    - Make it so new manuals can be inputted.
    - You can also input a manual in by yourself if the recommendation system is not satisfactory.

For Websites, potentially try using beautiful soup to scrape text and images. This won’t work with tabular data though.