**Considerations: -**

1. **Scraping –** In financial report order and structure is important, especially with the tables as a deterministic answer is always expected.

I chose to scrap using hr\_tags.

hr\_tag

A picture containing text, font, screenshot, letter

Description automatically generated

hr\_tag

1. Scrap over first hr\_tag.
2. Scrap text between hr\_tags.
3. Scrap titles and tables between hr\_tags with best structure possible with some detailed html code.

**Advantage** -Text files with ordered and structured chunks. [1a]

**Disadvantage -** Not all documents might have patterns, time consuming compared to using libraries to do sentence wise split or smaller chunks. It is also a rare possibility that some parts of the texts might have more tokens than model permits.

Note: SEC does not allow scrapping the webpage and therefore, I downloaded it to the local system. I tried their API, it includes 8-K but not the Press Release. [1b]

1. **Preprocessing, Modelling –** Standard steps recommended by Open AI with some changes:
2. Append the scrapped txt files into a directory and create a DataFrame where each row is a txt file in the order it was scrapped. [1a]
3. Tokenize the text and plot histogram to check if the allowed threshold for the model used is crossed. In this case the threshold for text-davinci-003 model is 4,097 tokens. [6]
4. Convert text into numerical/vector representation using Open AI text-embedding-ada-002. [12]
5. Convert the question text into embeddings using the same model and get cosine distance between scrapped text embeddings and question text embeddings. Sort the distances in ascending order and choose a threshold for length of text based on tokens. [7]
6. Try different hyperparameters (max\_len, size, max\_token and prompts) and definite hyperparameters (temperature = 0 and top\_p = 0.1 - Generates data scripts that are more likely to be correct and efficient. Output is more deterministic and focused) accordingly to refine model and use text-davinci-003 model to answers questions. [8]

**Assumptions: -**

1. I tried text-davinci-003 and gpt-3.5-turbo but the latter did not give definite answers. [2]
2. I also tried to induce memory in that chat using ConversationalRetrievalChain from LangChain but it seems it has some bug which is either not resolved or I could not resolve it. [3]
3. To change numbers - Example from (15,423) to “$-15,423,000” I tried Regular Expression (re) on the text just after scrapping and then using Prompt Engineering. The former did not work at all, the latter worked well.
4. For prompt engineering, I followed the guidelines based on DeepLearning.AI - ChatGPT Prompt Engineering for Developers Course. I also tried a structured JSON format output, but the model started hallucinating with answers. I believe I should be able to do it with a few more Prompt Engineering attempts. [4]

Guidelines followed:

- Write Clear and Specific Instructions: Clear! = Short.

- Give the model time to think.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. **Innovation -** Solving mathematical problems - both quantitative and qualitative, doing data analysis and visualization and, converting files between formats using LangChain Python Agent. [5]

This can allow your clients to do calculations or visualizations. For example, quickly calculate Liquidation Value.

Example - Percentage Change:

A screenshot of a computer

Description automatically generated with medium confidence

Example – Visualization of Percentage Change:

Rate of change for number of restaurants

A picture containing text, screenshot, rectangle, diagram

Description automatically generated

**References: -**

* 1. <https://platform.openai.com/docs/guides/fine-tuning/case-study-entity-extraction>
  2. <https://www.sec.gov/edgar/sec-api-documentation>
  3. <https://platform.openai.com/examples/default-factual-answering>
  4. <https://scale.com/blog/chatgpt-vs-davinci#Classification%C2%A0>
  5. <https://github.com/hwchase17/langchain/issues/2133>
  6. <https://towardsdatascience.com/4-ways-of-question-answering-in-langchain-188c6707cc5a>

<https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/>

* 1. <https://youtu.be/aywZrzNaKjs?list=PLialE9a4poWub7ZDIY2FF5JguPu3nHvcg&t=703>
  2. <https://python.langchain.com/en/latest/index.html>
  3. <https://openai.com/blog/chatgpt-plugins>

1 token ~= 4 chars in English

1 token ~= ¾ words

100 tokens ~= 75 words

Or

1-2 sentence ~= 30 tokens

1 paragraph ~= 100 tokens

1,500 words ~= 2048 tokens

1. <https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them>
2. <https://github.com/openai/openai-cookbook/blob/main/examples/How_to_count_tokens_with_tiktoken.ipynb>

Which distance function should I use?

It is recommended using cosine similarity. While the choice of distance function typically doesn’t matter much, cosine similarity can be computed slightly faster using just a dot product. Additionally, OpenAI embeddings are normalized to length 1, meaning that cosine similarity and Euclidean distance will result in identical rankings.

<https://medium.com/@pankaj_pandey/openai-embeddings-frequently-asked-questions-afac07f38317>

<https://community.openai.com/t/cheat-sheet-mastering-temperature-and-top-p-in-chatgpt-api-a-few-tips-and-tricks-on-controlling-the-creativity-deterministic-output-of-prompt-responses/172683>

* 1. <https://www.mlq.ai/fine-tuning-gpt-3-question-answer-bot/>
  2. <https://www.mlq.ai/fine-tuning-gpt-3-earnings-call-assistant/>
  3. <https://github.com/openai/openai-cookbook/blob/main/apps/web-crawl-q-and-a/web-qa.ipynb>
  4. <https://github.com/openai/openai-cookbook/blob/main/examples/Question_answering_using_embeddings.ipynb>

<https://community.openai.com/t/difference-between-frequency-and-presence-penalties/2777>

<https://openai.com/blog/introducing-text-and-code-embeddings>

<https://platform.openai.com/docs/guides/embeddings>