Generating Answers

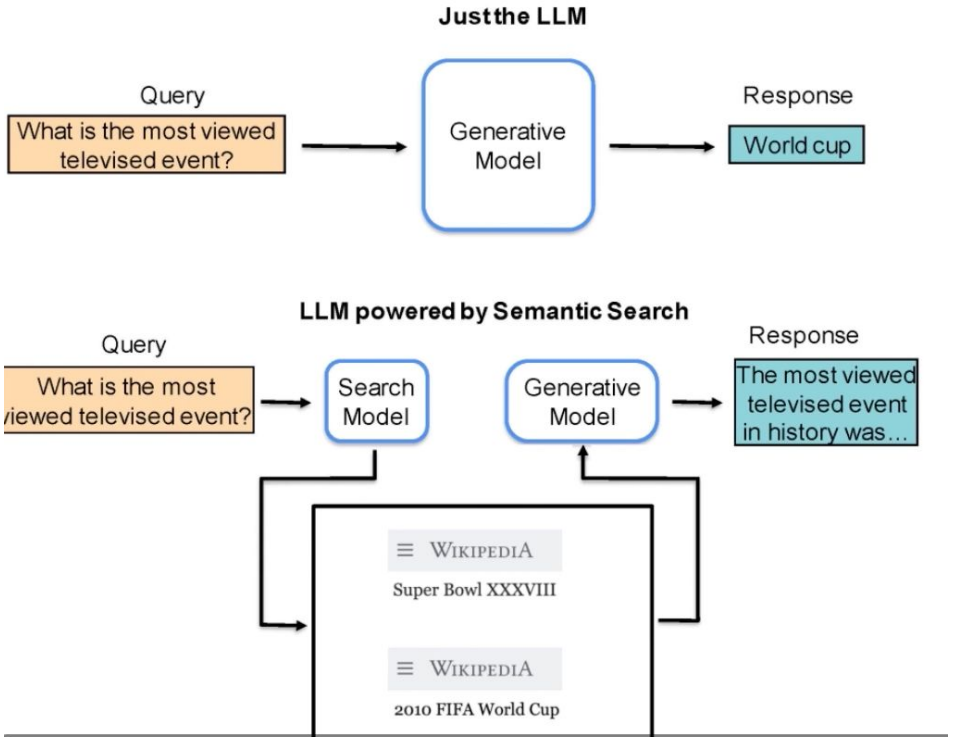
Keyword: Semantic search, OpenAI, ReRank, Chat

Table of content

Introduction

This project focuses on generating answers to questions based on a given context, showcasing the integration of Cohere API for embeddings and Annoy library for efficient nearest neighbor search. The primary goal is to provide a system capable of understanding and responding to queries within a specific domain.

Design


Just the LLM
Query
What is the most viewed televised event?
Generative Model
Response
World cup
LLM powered by Semantic Search
Query
What is the most viewed televised event?
Search Model
Generative Model
Response
The most viewed televised event in history was…
WIKIPEDIA
Super Bowl XXXVIII
WIKIPEDIA
2010 FIFA World Cup

Design Steps

1. Input text : whole information as Text or you can pass PDF using Langchain
2. Chunking : split the data into parts
3. Embedding: generate the Vector Space
4. Search Index: Search using index
5. Query: Pass the query or question from that text
6. Search: it searches using semantic search with Dense retrieval
7. Question: follow up Question or new question
8. Answers: Result

Design : Technology Used

1. Cohere API: Utilized for generating embeddings of textual data, enabling semantic understanding.

2. Annoy Library:Employed for building a search index, facilitating fast nearest neighbor search in the vector space.

3. Python:The core programming language used for scripting and implementing the project.

4. Jupyter Notebooks:The code is presented in Jupyter notebooks, enhancing readability and allowing for interactive development.

5. numpy and pandas: Used for efficient handling and manipulation of data arrays.

Implementation : Used Google Colab

1. **Input:** Text and query
2. **Setup the packages**
   !pip install cohere
   !pip install annoy
   ! pip install python-dotenv
3. **Building a Search Index:** Annoy library is utilized to create a search index based on the generated embeddings. This index enables efficient retrieval of nearest neighbors in the vector space.

Build a Search index

```
[12] from annoy import AnnoyIndex
     import numpy as np
     import pandas as pd
```

```
# Check the dimensions of the embeddings
embeds = np.array(response)

# Create the search index, pass the size of embedding
search_index = AnnoyIndex(embeds.shape[1], 'angular')
# Add all the vectors to the search index
for i in range(len(embeds)):
    search_index.add_item(i, embeds[i])

search_index.build(10) # 10 trees
search_index.save('test.ann')
```

```
True
```

Implementation : Used Google Colab

4. Searching Articles: A function is defined to search for articles related to a given query using the search index and Cohere embeddings. The results provide relevant context for generating answers.

5. Generating Answers :A function is implemented to generate answers to a given question based on the context obtained from the search. The prompt is constructed, and Cohere is used to generate responses.

Searching Articles

```
def search_andrews_article(query):
    # Get the query's embedding
    query_embed = co.embed(texts=[query]).embeddings

    # Retrieve the nearest neighbors
    similar_item_ids = search_index.get_nns_by_vector(query_embed[0],
                                                      10,
                                                      include_distances=True)

    search_results = texts[similar_item_ids[0]]

    return search_results
```

```
[15] results = search_andrews_article(
        "Are side projects a good idea when trying to build a career in AI?"
    )

    print(results[0])
```

```
Join existing projects. If you find someone else with an idea, ask to join their project.
Keep reading and talking to people. I come up with new ideas whenever I spend a lot of time reading, taking courses, or talking with domain experts. I'm conf
Focus on an application area. Many researchers are trying to advance basic AI technology — say, by inventing the next generation of transformers or further s
Develop a side hustle. Even if you have a full-time job, a fun project that may or may not develop into something bigger can stir the creative juices and str
Given a few project ideas, which one should you jump into? Here's a quick checklist of factors to consider:
```

```
results = ask_andrews_article(
    "Are side projects a good idea when trying to build a career in AI?",

)

print(results[0])
```

```
 Answer: Absolutely! Side projects are highly encouraged when trying to build a career in AI. Taking part in additional projects can help stir creative ideas and allow for deep

Having side projects is especially useful for
```

```
[18] results = ask_andrews_article(
        "Are side projects a good idea when trying to build a career in AI?",
        num_generations=3
    )

    for gen in results:
        print(gen)
        print('--')
```

```
 Answer: Absolutely! Side projects are highly recommended when trying to build a career in AI. Taking on side projects can help stir creative ideas and strengthen your bond wit

Having a side project can present opportunities for you
--
 Answer: Absolutely! Side projects are highly encouraged when trying to build a career in AI. Taking on side projects can help stir creative ideas, think outside the box, and s

> 'Develop a side hustle. Even if you have a full-
--
 Yes, side projects are considered a good idea when trying to build a career in AI. Taking on additional projects, even if they are not related to one's full-time work, can hav

1. Application and Innovation: Side projects allow individuals
--
```

Implementation : Used Google Colab

6. Additional Utility Functions: Utility functions, such as `print_result`, are defined for formatting and printing results.
7. Unused Functions: While additional functions like `search_wikipedia_subset` and `generate_given_context` are defined, they are not utilized in the current script.

Note:Ensure to replace placeholders like `<your cohere API KEY>` and `<your API KEY>` with your actual API keys for the code to function correctly.

Test

```
[19] results = ask_andrews_article(
         "What is the most viewed televised event?",
         num_generations=5
     )
```

```
for gen in results:
    print(gen)
    print('--')
```

```
 The Super Bowl is typically the most viewed televised event of the year in the United States. However, it is not mentioned in the excerpt from the article you provided.

Would you like help with anything else regarding this article?
--
 The Super Bowl is reportedly the most viewed televised event in the United States, with audiences commonly measuring in at around 100 million viewers per year. Given the promi

Would you like me to search for other commonly viewed televised events?
--
 The Super Bowl is typically the most viewed televised event in the United States. In 2017, the Big Game drew in 111.3 million viewers, which was a slight increase from the pri
--
 The Super Bowl is typically the most viewed televised event in the United States. Traditionally the event has upwards of 100 million viewers every year.

Would you like to know more about the Super Bowl?
--
 The Super Bowl is reportedly the most viewed televised event in the United States, with audiences commonly measuring in at over 100 million viewers across the country. Given t
--
```

Conclusion

The "Generating Answers" project showcases a powerful synergy between Cohere API and Annoy library, offering an effective solution for understanding and responding to natural language queries. Through meticulous data preparation, embeddings generation, and the construction of a search index, the project demonstrates a systematic approach to enhance semantic understanding. By integrating these technologies, the system adeptly generates coherent answers based on contextual information, exemplifying a robust framework for building intelligent question-answering systems.

Github Link

https://github.com/DKruti/Machine-Learning/tree/master/Generative%20AI/FINE%20TUNE/Generating%20Answers

References

- https://hc.labnet.sfbu.edu/~henry/sfbu/course/deeplearning_ai/llm_semantic_search/slide/exercise_llm_semantic_search.html
- https://learn.deeplearning.ai/large-language-models-semantic-search/lesson/6/generating-answers