11-711 Assignment 2 : Retrieval Augmented Generation System

Rajeev Veeraraghavan *

rveerara@andrew.cmu.edu

Abstract

This report summarizes the work done to build a stand alone Retrieval augmented generation system to answers questions on CMU (Carnegie Mellon University). The system uses publicly available data to build a knowledge base, and LLama2-7b model to generate answers to questions.

1 Introduction

2 Data Creation

2.1 Raw Data Collection

2.1.1 Which Data was Collected ?

All collected raw data were from the publicly available sources mentioned on the assignment's github repository (Neubig, 2024). All the documents listed on the page were used because they broadly covered the topics that would be asked in the question set.

2.1.2 Data Extraction

LTI Faculty The LTI faculty page was scraped using BeautifulSoup to get the names of the faculty members and their research interests. The scraped text was stored in raw fashion with one file per faculty member. These files were transformed into a single JSON file and also into a descriptive text file where each sentenced contained exactly one faculty member's information.

LTI Faculty Research Semantic Scholar was used to get the research papers (and abstract, metadata) of the faculty members. The data was parsed into a JSON file which had a list where each member contained information about one paper.

Academics @ LTI The old LTI website landing page was scraped using BeautifulSoup to get gists of the courses offered by LTI. The scraped text was stored in raw fashion with one file per course, and fed directly to the embedder. The program handbooks were downloaded as pdfs and were then parsed using PyPdf to generate text files. The PDFs were directly used in one set of embeddings while the text files were used in another set of other embeddings.

CMU Courses The CMU course schedule catalog was scraped using BeautifulSoup to get the schedules of all courses. The scraped data was converted into a JSON file and into a combined text file where each sentence contained a worded form of one course's information. The academic calendars for 2022-23 and 2023-24 were downloaded as PDFs and parsed into text files. Both formats were fed into different sets of embedders.

History, Athletics and Events The relevant pages were scraped with BeautifulSoup, and PDFs were downloaded where available. As the documents here had different inherent structures, most of the data was manually copied into text files and parsed with separate scripts to convert the raw data into sentence form.

2.2 Annotations

2.2.1 Type and Quantity of Annotations

A total of 115 test annotation pairs were created manually - 31 for the history and events category, 15 for the LTI program handbook category, 30 for the CMU courses category, 20 for the LTI faculty category, and 19 for the LTI faculty research category. This split covered the different types of data collected, and the different types of questions that could be asked on them. No train annotations were created, since there was no fine tuning or few shot prompting being performed. The annotations

*This work was done as part of the 11-711 course at Carnegie Mellon University. The assignment started off with a group of 3 members, but after 21st February, I had to complete the assignment individually. Permission to work alone was taken from the course instructor. Permission to use material that the group had worked on together, prior to splitting up, was also taken from the course instructor. Shared work has been attributed to contributors.

were used to select the best models and embeddings. All the reference answers were kept down to 5 words with multiple variations being accepted (eg - 11711; 11-711; 11711 course; 11-711 course; 11711 course at CMU; 11-711 course at CMU etc). Reference answers were kept as short as possible with the goal of selecting a model that gave concise answers (to maintain the F1 score), while maintaining recall. This likely resulted in low f1 scores, but the idea behind this is to select a model that is concise. As I worked alone on this assignment, I could not calculate an inter annotator agreement score.

3 Model Details

3.1 Constraints

The RAG system was built on a GCP compute instance - n1-highmem-4 with 4 core CPU, 1 NVIDIA T4 GPU, 26 GB RAM, and 160 GB SSD. This influenced the choice of inference and embedding models. Langchain (Docs) was used to build the embedder and retriever modules due to its ease of use and integrations available.

3.2 Embedding Models and Vector Store

I chose to use the bge-large-en-v1.5 BGE embedding model as it is an established model in the top 15 of the MTEB leader board. Further, the v1.5 model update embedding model: release bge-en-v1.5 embedding model alleviates the issue of the similarity distribution and has ehanced retrieval ability without instruction, compared to earlier bge models.

I experimented with the chunk sizes for both embeddings to find out the which size fits best. I used recursive chunking for greater granularity and variety of text, varying the chunk overlap between 0.1 and 0.4 times the size of the chunk size to see what effect the overlap had on the retrieval results. The embeddings were stored in a Chroma vector database due to its support for multiple formats and ease of use for a first time user.

I also evaluated llama2-7b embeddings and compared the two sets of embeddings to see which one gave better results. My hypothesis was that the llama2-7b embeddings along with the llama2-7b model for inference would provide comparable results to using the BGE embedding with llama2-7b and similar sized models.

Model	Chunk Size	F1 Score %	Recall
Llama2	500	5.6	
Openchat	500	7.8	
Neural Chat	500	7.8	
Llama2	1000	5.6	
Openchat	1000	7.8	
Neural Chat	1000	7.8	

Table 1: Evaluation results with Llama2 embeddings

Model	Chunk Size	F1 Score %	Recall
Llama2	500	5.6	
Openchat	500	7.8	
Neural Chat	500	7.8	
Llama2	1000	5.6	
Openchat	1000	7.8	
Neural Chat	1000	7.8	

Table 2: Evaluation results with BGE embeddings

3.3 Inference Models

I used the following suite of open source models available with Ollama: llama2-7b, mistral, openchat and neural-chat. I expected the 4 models to give different performances across the metrics measured while all being light weight enough to run locally with a reasonable infernce time. Among the 4 models, I expected to choose 2 for the final unseen test set submission, based on their relative peformance F1, recall and exact match. I did not expect any of these models to do very well at exact matching as they are all chat models.

4 Results

4.1 Embedding Models

I compared performance across 2 embeddings to choose an embedding model - **bge-large-en-v1.5 and llama2-7b**. I used F1 score as the primary metric to compare the embeddings, along with recall, precision, rogue score and exact match. The results are shown in tables 1 and 2.

Which embedding was better across evaluation metrics, chunk sizes and models ? Was one embeddings signifactly better than the other ?

4.2 Comparison of Models

After selecting the embedding, I compared the performance of the 4 models across the same metrics as the embeddings. The results are shown in tables 3.

How do the models comapre across evaluation

Model	Chunk Size	Retriever	F1 Score %
Llama2	500		
Openchat	500		
Neural Chat	500		
Llama2	1000		
Openchat	1000		
Neural Chat	1000		

Table 3: Evaluation results with Llama2 embeddings

metrics, chunk sizes and models ? Was one model signifactly better than the rest ?

5 Analysis

The annotations contained questions that required different types and numbers of documents to answer them. For example, most questions from the history category could be found from the context within one document. Similarly, questions from the schedules category that simply asked where or when a class was held were expected to be easier for the RAG system. Questions from the LTI faculty and LTI research categories were also of two types - ones whose answers could be found within one document or ones that needed collection of information from multiple documents. By document, I refer to the chunked documents here. Questions such as *Which author of the paper titled " Unsupervised Dense Retrieval Training with Web Anchors" did not publish any other paper in 2023 ?* required information from all documents corresponding to LTI papers. This question was not expected to be answere correctly by any of the baseline models. I categorized the question based on a hypothetical difficulty level, in terms of how many documents would be needed to capture the answer, and evaluated the peformance of the models I tsted across these difficulty levels. The results of this evaluation are given below.

5.1 Single Document Questions

When was the Carnegie Plan initiated ?

5.2 Simple Multi Document Questions

5.3 Complex Multi Document Questions

6 Conclusion

Acknowledgements

I acklowledge the contributions of the following people to this assignment and report:
Sudeep Agarwal (sudeepag@andreew.cmu.edu) ***contributed to the scraping of raw data for academic calendar schedule, LTI faculty and LTI faculty research***
Hugo C (hcontant@andrw.cmu.edu) ***contributed to the scraping of raw data for Kiltie band, Scottie and Tartans***

References

Lang Chain Docs. LangChain. [link].

Graham Neubig. 2024. nlp-from-scratch-assignment-spring2024. github repository.