RAG system for answering questions related to CMU

Akshay Badagabettu [*], **Sai Sravan Yarlagadda** [†], **Wenjin Fu** [‡]

Abstract

As a part of this assignment we worked on building a Retrieval Augmented Generation system that is capable of answering questions related to various facts about Carnegie Mellon University(CMU) and Language Technologies Institute (LTI). This is one particular topic that requires domain knowledge related to CMU and LTI for answering questions. Our RAG system addresses LLM's domain knowledge limitations by compiling documents relevant to the questions at hand. The core of our endeavor involved extensive data preparation and annotation phase, employing both automated tools and manual efforts. Incorporating both a multiquery retriever and a reranker into the RAG system yielded statistically significant results when compared to utilizing only a retriever in the RAG framework. This model had given a F1 score of 0.4161, which is 1.57 times better than the F1 score of a retriever only model.

1 Introduction

In the field of Natural Language Processing, developing systems that can answer domain-specific questions from diverse and complex sources presents a challenging task. This project report outlines our journey in constructing a Retrieval Augmented Generation (RAG) (Lewis et al., 2020) system tasked with answering questions given domain-specific knowledge about CMU and LTI. Our RAG system consisted of three main components: a document embedder, a document retriever, and a question-answering system. We experimented with various models and techniques to optimize each component, ultimately settling on a dense retriever approach for its ability to handle long texts and retrieve contextually relevant documents. The report details our experimentation with different models and methodologies, including the use of reranker to refine the selection of documents, and multiquery module to broaden the scope of document retrieval. Through this project, we contributed valuable insights into the RAG framework and its potential in leveraging structured knowledge to enhance LLM performance in specialized domains.

2 Data Creation

2.1 Compiling knowledge resources

We have scraped all the websites, PDFs, and research papers. We then have developed various Python scripts that can parse this information into a text file. All the scraped text files can be found on GitHub[1].

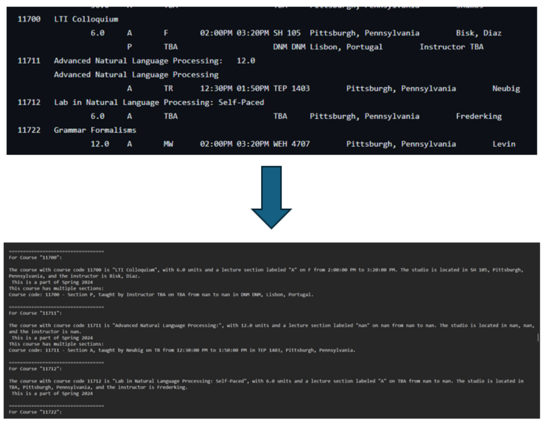
2.2 Tools used to extract data

The pdf sources were parsed through libraries like pypdf and fitz. OCR has also been performed on images in some pdf's. The details regarding research papers were obtained with the help of Semantic Scholar Academic API (sem), and the HTML websites were scraped with the help of Beautiful Soup Library (Richardson, 2007). Certain PDF files and websites had complex structures (text and tables together); some examples of such PDFs were the schedule of classes and academic calendars. In our exploration of processing research papers and PDFs that contain multiple columns, we evaluated various libraries, including PyPDF and PyPDF-Plumber. However, only PyMuPDF (git, b) demonstrated the capability to read the document in its natural reading order, effectively handling documents with multiple columns. We initially tried retrieving documents without taking into consideration the structure of the document. In other words, we scraped the tables into a txt file and embedded it directly. We then tested out the retriever by asking

[*]abadagab@andrew.cmu.edu
[†]saisravy@andrew.cmu.edu
[‡]wenjinf@andrew.cmu.edu

[1]https://github.com/akshay140601/End-to-End-NLP-System

questions whose answers lie in the middle of this huge table. The retriever was not able to retrieve the correct chunk. To solve this problem related to the structure of the tabular data, we have converted all the tabular data into a natural language format. An example of the results of this conversion is shown in Figure 1.



```
11700   LTI Colloquium
                6.0     A       F       02:00PM 03:20PM SH 105  Pittsburgh, Pennsylvania        Bisk, Diaz
                        P       TBA                     DNM DNM Lisbon, Portugal        Instructor TBA
11711   Advanced Natural Language Processing:   12.0
        Advanced Natural Language Processing
                        A       TR      12:30PM 01:50PM TEP 1403        Pittsburgh, Pennsylvania        Neubig
11712   Lab in Natural Language Processing: Self-Paced
                6.0     A       TBA                     TBA     Pittsburgh, Pennsylvania        Frederking
11722   Grammar Formalisms
                12.0    A       MW      02:00PM 03:20PM WEH 4707        Pittsburgh, Pennsylvania        Levin
```

```
==================================
For Course "11700":

The course with course code 11700 is "LTI Colloquium", with 6.0 units and a lecture section labeled "A" on F from 2:00:00 PM to 3:20:00 PM. The studio is located in SH 105, Pittsburgh,
Pennsylvania, and the instructor is Bisk, Diaz.
 This is a part of Spring 2024
This course has multiple sections:
Course code: 11700 - Section P, taught by Instructor TBA on TBA from nan to nan in DNM DNM, Lisbon, Portugal.

==================================
For Course "11711":

The course with course code 11711 is "Advanced Natural Language Processing:", with 12.0 units and a lecture section labeled "nan" on nan from nan to nan. The studio is located in nan, nan,
and the instructor is nan.
 This is a part of Spring 2024
This course has multiple sections:
Course code: 11711 - Section A, taught by Neubig on TR from 12:30:00 PM to 1:50:00 PM in TEP 1403, Pittsburgh, Pennsylvania.

==================================
For Course "11712":

The course with course code 11712 is "Lab in Natural Language Processing: Self-Paced", with 6.0 units and a lecture section labeled "A" on TBA from nan to nan. The studio is located in
TBA, Pittsburgh, Pennsylvania, and the instructor is Frederking.
 This is a part of Spring 2024

==================================
For Course "11722":
```

Figure 1: Example showing how structured data was converted to natural language

2.3 Annotated data for testing

We have placed more emphasis on generating quality questions. This emphasis on quality ensured that we did not end up with redundant questions. We also set strong annotation guidelines, such as keeping the answers concise, and not annotating similar kinds of questions (for example: if one question is "What is Graham Neubig's job title?", we did not add more questions that tested the retriever in the same area such as "What is Yonatan Bisk's job title?")

We have annotated 176 questions from all the sources we have scraped. We did try out multiple ways to generate question-answer pairs. We have also proved that 176 questions are enough to identify differences in our various RAG systems by performing a statistical significance test which is discussed in later sections. Initially, we took the scraped text file, chunked it, and passed it as a prompt to an LLM (System-based annotations). An instruction was also added in the prompt to generate a question-answer pair from the passed chunk. We then manually selected the good questions from the generated chunk of the question and answer pairs.

The LLM used for generating these annotations was the 4-bit quantized version of Google Flan T5-xl (3B parameters) (Chung et al., 2022). We have also tried using Mistral 7B instruct, but the inference time with Mistral was very high, hence we ended up using the Flan T5. The main reason for using this model was because it was instruction-tuned specifically on QA datasets. Later, we also manually generated more QA pairs by going through the documents.

2.4 Estimation of quality of annotations - Inter Annotator Agreement (IAA)

Two members of the team independently annotated a subset of our test consisting of 40 questions randomly sampled from the test set and achieved an exact match for 38 questions out of 40. The F1 score, precision, and recall between the two annotators were 0.9833, 0.98125 and 0.9875 respectively. The high scores after doing IAA prove that our annotations are more uniform, and all the annotators followed the annotation guidelines properly.

3 Model Details

We have developed 4 RAG systems. Each system is built on top of the previous one. We decided what method/strategy to adopt for each new system after thoroughly analyzing the shortcomings of the previous system. The models and methods used, and the analysis of the shortcomings are discussed in this section. It should also be noted that the reader model chosen for all 4 systems is Llama2-70B-chat. The justification for this is given at the end of this section.

3.1 Retriever only system

First, we have to embed all our scraped and preprocessed documents. To do this, we used the model 'BAAI/bge-large-en-v1.5' (Chen et al., 2024) This model is 13th on the MTEB leaderboard (Muennighoff et al., 2022). The main reason we chose this model is because it supports longer texts. This is a dense retriever method. The main limitation that comes with sparse retriever is that it does not retrieve context where words do not match exactly(Izacard et al., 2021). In other words, sparse retrievers rely on exact term matches to retrieve documents. From (Arabzadeh et al., 2021), we see that dense retrievers provide substantial performance improvements when compared to traditional sparse retrievers, such as BM25. The only downside with dense retrievers is that they are computationally more expensive. But, for this particular scenario, we value performance over

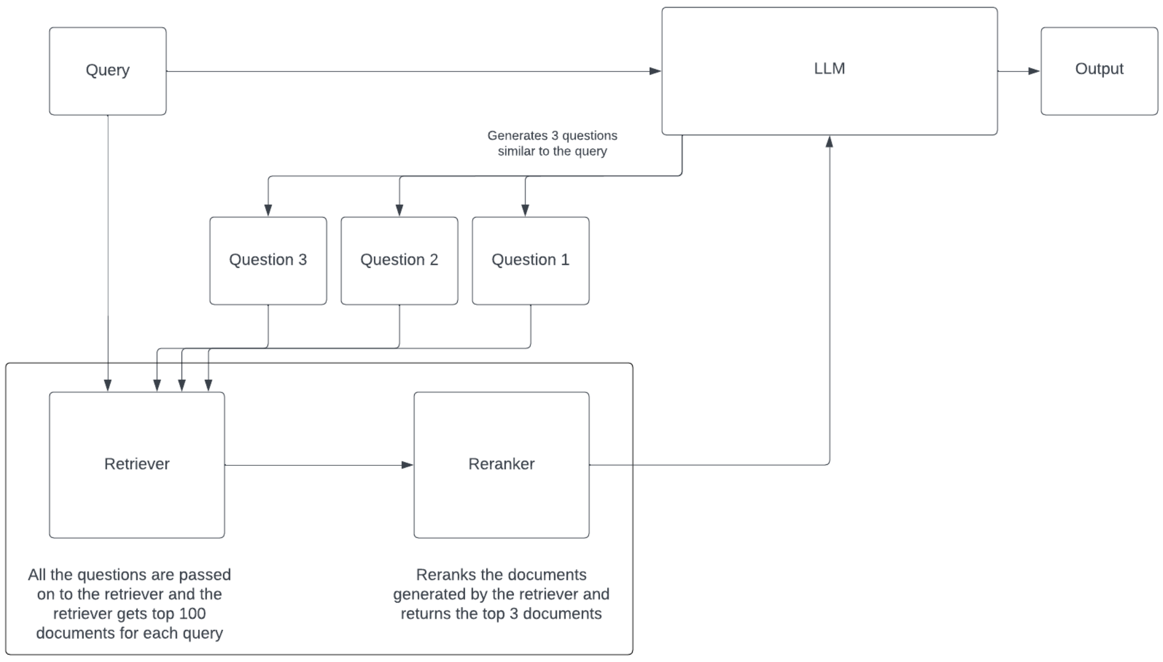

Query
LLM
Output
Generates 3 questions similar to the query
Question 3
Question 2
Question 1
Retriever
Reranker
All the questions are passed on to the retriever and the retriever gets top 100 documents for each query
Reranks the documents generated by the retriever and returns the top 3 documents

Figure 2: System architecture of our best system

everything else, and so decided to go with dense retrievers. We have used ChromaDB (git, a) as our vector database, to store our embeddings. The main reason for choosing ChromaDB was due to its ease of implementation. We have used a constant chunk size of 512 and an overlap of 100 between each chunk. The overlap was necessary as it helps in understanding the right context in certain examples. We initially fixed our chunk size to 5000, and we observed that the model was not able to retrieve the right documents. Especially in documents like the schedule of classes and faculty information, the retriever model was retrieving the wrong documents. We had initially increased our chunk size to 5000 in an effort to reduce the computation time during embedding. However, creating one vector for 5000 characters was not the right decision as the retriever was not able to give meaningful results even to simple questions. After some trial runs, we found that the chunk size of 512 works best. This chunk size is not too small, which would make retriever time shoot up and also retrieve meaningless small contexts. It's also not too big, which would lead to the issue of the retriever not finding meaningful documents after doing cosine similarity.

We then performed a quick qualitative analysis of our system and looked at the generated answers and their corresponding retrieved contexts. We then took a few questions where the system failed and looked at what documents it retrieved for those questions (the top 100 documents). We noticed that for many questions, the retriever was able to retrieve the correct document, but it wasn't in top-3 positions. This is why we added a reranker to the next system.

3.2 Retriever + reranker system

For reasons explained above, we performed a reranking of the documents using the model 'BAAI/bge-reranker-large.' This model is a cross-encoder reranking that encodes both queries and documents using neural model (Nogueira and Cho, 2019). We retrieved the top 100 chunks from the retriever and then passed them to the reranker model to rerank the documents, and then we chose the top 3 documents and passed them to the LLM (reader). This model gave better answers than the previous model.

We again performed a qualitative analysis and observed that this system overcame the previous problem of the correct context not being a part of top-3 contexts. We then experimented with changing the wording of our question and then checking if the retriever was able to find the correct documents. Surprisingly, the wording of the question played

Model	F1 score	Precision	Recall	Exact match
Closed book	0.1352	0.12	0.2894	8/176
Retriever only	0.2648	0.2432	0.4904	29/176
Reranker + Retriever	0.2760	0.2548	0.5309	28/176
Reranker + Multi-Query Retriever	0.4161	0.3898	0.6136	38/176

Table 1: Evaluation of the models based on F1 score, precision, recall and exact match.

a huge role when retrieving the context, so we decided to add a multi-query retriever to our next system.

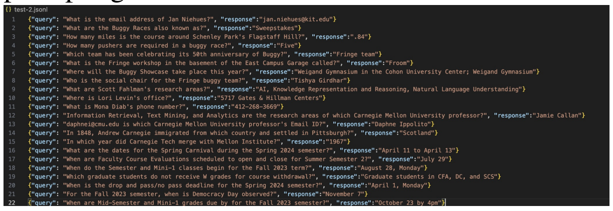
3.3 Retriever + reranker + multi-query system

For the reasons explained above, we added a multi-query retriever on top of the previous system. In this system, the question is first passed to the LLM along with a prompt instructing it to generate 3 different questions by paraphrasing the given question. We then retrieve 100 documents for all 4 questions (1 original question, and 3 generated questions), which gives us a total of 400 documents. The duplicate documents are removed from this, and the list of unique documents is passed to the reranker, which then gives the best/top 3 documents. This turned out to be our best system and the system architecture is shown in Figure 2.

3.4 Retriever + reranker + multi-query + few-shot prompting

To better format the answer generated by our RAG system, we also tried the few-shot prompting technique to facilitate the model's ability to generate structured responses by dynamically integrating relevant examples into the prompt. The prompt is configured with placeholders for both the context information and the dynamically loaded few-shot examples. Our prompt instructs the model to generate answers based on the provided context, without prior knowledge, and to format these answers similarly or the same as given few-shot QA examples. Central to this approach was the utilization of cosine similarity for the selection of few-shot examples from pre-prepared JSON files, which contain 50 annotated QA pairs examples (Figure 3), closely aligned with the semantic context of incoming queries. By embedding both queries and potential examples within the same high-dimensional space, cosine similarity allowed us to identify and dynamically integrate the model's pertinent examples into the model's prompt. Through this approach, we aim to generate better structure and formatted answers through our prompt engineering approach.

Surprisingly, few-shot prompting did not work very well. We think that this may be because we are confusing the reader model with the way that we have set up the examples in the prompt. The examples contain QA pairs but are missing the context part. Hence the reader does not know how it arrived at the answer in the example that we are passing. This may be the reason that our version of few-shot prompting failed.

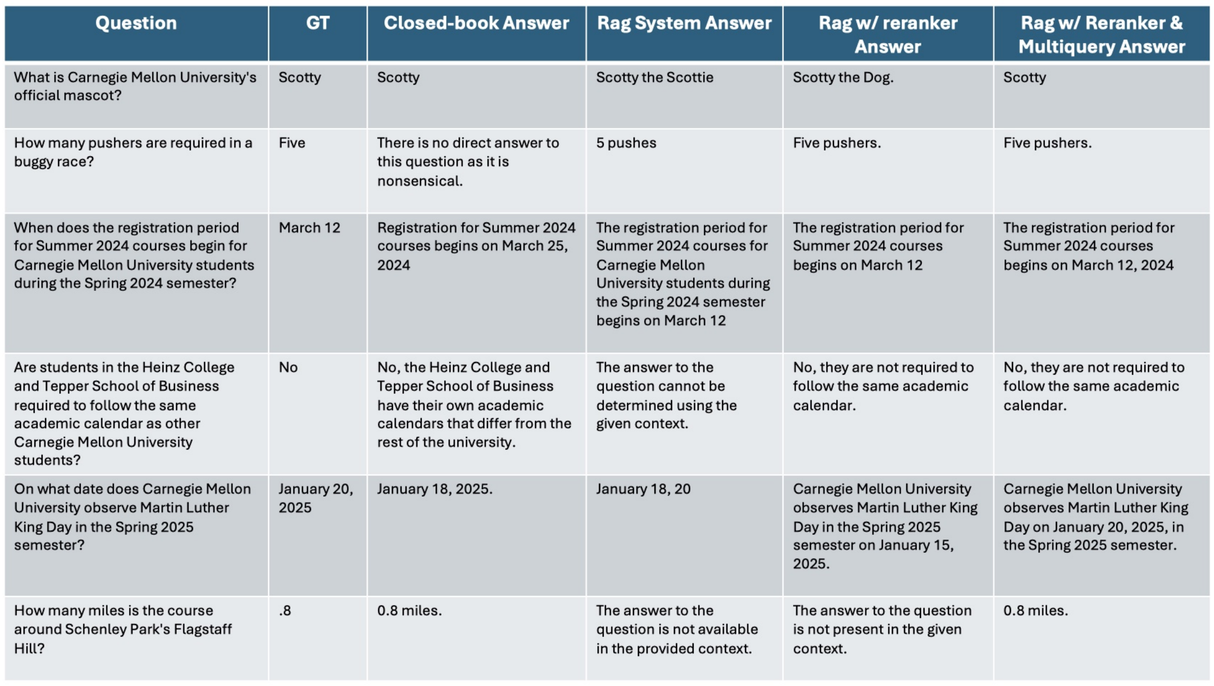


```
{"query": "What is the email address of Jan Niehues?", "response":"jan.niehues@kit.edu"}
{"query": "What are the Buggy Races also known as?", "response":"Sweepstakes"}
{"query": "How many miles is the course around Schenley Park's Flagstaff Hill?", "response":".84"}
{"query": "How many pushers are required in a buggy race?", "response":"Five"}
{"query": "Which team has been celebrating its 50th anniversary of Buggy?", "response":"Fringe team"}
{"query": "What is the Fringe workshop in the basement of the East Campus Garage called?", "response":"Froom"}
{"query": "Where will the Buggy Showcase take place this year?", "response":"Weigand Gymnasium in the Cohon University Center; Weigand Gymnasium"}
{"query": "Who is the social chair for the Fringe buggy team?", "response":"Tishya Girdhar"}
{"query": "What are Scott Fahlman's research areas?", "response":"AI, Knowledge Representation and Reasoning, Natural Language Understanding"}
{"query": "Where is Lori Levin's office?", "response":"5717 Gates & Hillman Centers"}
{"query": "What is Mona Diab's phone number?", "response":"412-268-3669"}
{"query": "Information Retrieval, Text Mining, and Analytics are the research areas of which Carnegie Mellon University professor?", "response":"Jamie Callan"}
{"query": "daphnei@cmu.edu is which Carnegie Mellon University professor's Email ID?", "response":"Daphne Ippolito"}
{"query": "In 1848, Andrew Carnegie immigrated from which country and settled in Pittsburgh?", "response":"Scotland"}
{"query": "In which year did Carnegie Tech merge with Mellon Institute?", "response":"1967"}
{"query": "What are the dates for the Spring Carnival during the Spring 2024 semester?", "response":"April 11 to April 13"}
{"query": "When are Faculty Course Evaluations scheduled to open and close for Summer Semester 2?", "response":"July 29"}
{"query": "When do the Semester and Mini-1 classes begin for the Fall 2023 term?", "response":"August 28, Monday"}
{"query": "Which graduate students do not receive W grades for course withdrawal?", "response":"Graduate students in CFA, DC, and SCS"}
{"query": "When is the drop and pass/no pass deadline for the Spring 2024 semester?", "response":"April 1, Monday"}
{"query": "For the Fall 2023 semester, when is Democracy Day observed?", "response":"November 7"}
{"query": "When are Mid-Semester and Mini-1 grades due by for the Fall 2023 semester?", "response":"October 23 by 4pm"}
```

Figure 3: Example of QA Examples stored in JSON file

3.5 Choosing the reader model

When choosing the reader model, we had just one condition. The condition was to choose a strong enough model such that if the retriever was able to retrieve the correct documents, then the LLM must be able to get to the correct answer. As there were no restrictions placed on inference time and cost, we decided to go with a strong LLM viz., Llama2-70B-chat (Touvron et al., 2023). We also used Mistral-7B-instruct-v0.2 (Jiang et al., 2023), because Mistral has one of the best instruction-tuning on QA datasets. We observed that LLAMA2 was very verbose in comparison with Mistral. However, the answers generated by LLAMA2 were better than those of Mistral. Mistral was much better at following instructions provided in the prompt, such as "Do not be verbose." As explained before, the emphasis is put on getting the answer right (increased performance), and hence we went ahead with using Llama2-70B-chat as our reader. We used together API (tog) to run this LLM. This was extremely good, as they have an optimized implementation, and the inference time, even for such



Question	GT	Closed-book Answer	Rag System Answer	Rag w/ reranker Answer	Rag w/ Reranker & Multiquery Answer
What is Carnegie Mellon University's official mascot?	Scotty	Scotty	Scotty the Scottie	Scotty the Dog.	Scotty
How many pushers are required in a buggy race?	Five	There is no direct answer to this question as it is nonsensical.	5 pushes	Five pushers.	Five pushers.
When does the registration period for Summer 2024 courses begin for Carnegie Mellon University students during the Spring 2024 semester?	March 12	Registration for Summer 2024 courses begins on March 25, 2024	The registration period for Summer 2024 courses for Carnegie Mellon University students during the Spring 2024 semester begins on March 12	The registration period for Summer 2024 courses begins on March 12	The registration period for Summer 2024 courses begins on March 12, 2024
Are students in the Heinz College and Tepper School of Business required to follow the same academic calendar as other Carnegie Mellon University students?	No	No, the Heinz College and Tepper School of Business have their own academic calendars that differ from the rest of the university.	The answer to the question cannot be determined using the given context.	No, they are not required to follow the same academic calendar.	No, they are not required to follow the same academic calendar.
On what date does Carnegie Mellon University observe Martin Luther King Day in the Spring 2025 semester?	January 20, 2025	January 18, 2025.	January 18, 20	Carnegie Mellon University observes Martin Luther King Day in the Spring 2025 semester on January 15, 2025.	Carnegie Mellon University observes Martin Luther King Day on January 20, 2025, in the Spring 2025 semester.
How many miles is the course around Schenley Park's Flagstaff Hill?	.8	0.8 miles.	The answer to the question is not available in the provided context.	The answer to the question is not present in the given context.	0.8 miles.

Figure 4: Generated Answer Comparison between Different Models

a heavy model, was in milliseconds. It is recommended to see the exact wording of the prompt we used for the LLM in our GitHub.

3.6 Choosing k in top-k documents to pass

We chose to pass only the top 3 documents to the model rather than the top 20 or top 25 documents. The reason for this is, according to (Liu et al., 2024), the performance of the reader is often highest when relevant information occurs at the beginning or end of the input context, and significantly degrades when models must access relevant information in the middle of long contexts, even for long-context models. Hence, we decided to pass 3 documents, each having 512 characters.

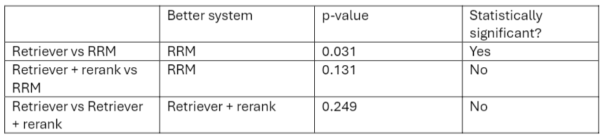
3.7 Closed-book use of model

Here, we did not retrieve any context and directly passed the question to the reader model. We observed that this system was able to answer general questions like "When did Andrew Carnegie Die?" but was performing very badly in questions specific to Carnegie Mellon University and LTI. Especially in questions that are temporal-based, such as "Who is teaching 11711 in Spring 2024?". This is because the LLMs do not have access to recent data, and their knowledge cutoff is at least a couple of years back.

4 Results

We calculated the exact match, F1 score, precision, and recall. Our implementation of these metrics is exactly the same as done in (Rajpurkar et al., 2016). The results for the 3 systems and the closed-book model version are shown in Table 1. We see that as expected, every addition we made to the previous system resulted in an increase in these metrics.

Significance testing is also performed. This script (git, c) is modified, and we use the F1 score as the metric. Bootstrapping was done 10,000 times, and the sample ratio used was 0.5. All three models were tested with each other. We fixed the significance value (α) as 0.05. The results of the significance test are shown in Figure 5.



	Better system	p-value	Statistically significant?
Retriever vs RRM	RRM	0.031	Yes
Retriever + rerank vs RRM	RRM	0.131	No
Retriever vs Retriever + rerank	Retriever + rerank	0.249	No

Figure 5: Significance testing results

From Figure 5, we observe that the results of the retriever-only system and Retriever + reranker + multi-query retriever (RRM) are statistically significant because the p-value is less than the α value. This proves that our test set contains enough samples to differentiate between these systems and that the increase in performance is not by chance. But, we can see from Figure 5 that for the next two tests, even though the system with the higher average F1 score wins, the results are not statistically significant. This shows that maybe our test set is not big enough to capture this increase in performance.

Outputs are generated for the test set provided to us. We have generated the outputs on all three systems we have developed, and we believe "system_output_1.txt" will give the best performance as that output is generated with our best system.

5 Analysis

In our quantitative analysis, the four systems — Closed Book Model, RAG System, RAG with Reranker, and RAG with Reranker & Multiquery — displayed distinct strengths and weaknesses when addressing a range of questions. The Closed Book Model adeptly provided answers to well-known facts, such as Carnegie Mellon University's official mascot, reflecting its proficiency in recalling general knowledge ingrained during training, as illustrated by the first question in Figure 4. It can swiftly generate responses but exhibits limitations with questions necessitating current and specialized knowledge, resulting in less accurate or incorrect answers for recent or detailed inquiries, such as the second, third, and fifth questions in Figure 4.

The basic RAG System, integrating a document retriever with a QA component, improved answer accuracy by leveraging domain-specific documents, yet its effectiveness is contingent on the quality of the retrieved documents. However, its performance heavily depends on the quality of the retrieved documents and it might struggle with questions that require synthesizing information from multiple sources, exemplified by the fourth question in Figure 4. The RAG with Reranker builds on this by implementing a re-ranking step and sharpening document selection to enhance the relevance and accuracy of responses. It effectively addressed the fourth question in Figure 4, where the basic RAG system faltered. Despite reranking, it may still produce less accurate responses if the reranker's criteria do not align perfectly with the question's intent.

The introduction of the Multiquery module, in conjunction with re-ranking, employs diverse representations of queries to retrieve and then re-rank a broader array of documents. This advanced configuration demonstrated its superior ability to synthesize comprehensive and accurate answers, as shown in its correct responses to the fifth and sixth questions in Figure 4, signifying its robustness in dealing with complex and nuanced information.

6 Conclusion

In summary, the project successfully developed a Retrieval Augmented Generation (RAG) system tailored for answering questions specifically about Carnegie Mellon University (CMU) and the Language Technologies Institute (LTI). By integrating a reranker, multi-query retriever, and leveraging the LLAMA2-70B-chat model for reading comprehension, our system has shown substantial improvement in generating accurate responses. Through extensive data preparation, annotation, and innovative approaches such as converting tabular data into a natural language format, we have enhanced the system's ability to interpret and process complex information. The quantitative analysis underscores the efficacy of our approach, with our best model configuration achieving a significant increase in performance metrics, demonstrating a 1.57 times improvement in the F1 score compared to a retriever-only model. This project not only highlights the potential of RAG systems in specialized domains but also sets a foundation for future research to explore and refine these techniques further.

References

a. GitHub - chroma-core/chroma: the AI-native open-source embedding database — github.com. https://github.com/chroma-core/chroma. [Accessed 13-03-2024].

b. GitHub - pymupdf/PyMuPDF: PyMuPDF is a high performance Python library for data extraction, analysis, conversion & manipulation of PDF (and other) documents. — github.com. https://github.com/pymupdf/PyMuPDF. [Accessed 14-03-2024].

Semantic Scholar | AI-Powered Research Tool — semanticscholar.org. https://www.semanticscholar.org/. [Accessed 14-03-2024].

TOGETHER — api.together.xyz. https://api.together.xyz/. [Accessed 14-03-2024].

c. util-scripts/paired-bootstrap.py at master · neubig/util-scripts — github.com. https:

//github.com/neubig/util-scripts/blob/master/paired-bootstrap.py. [Accessed 14-03-2024].

Negar Arabzadeh, Xinyi Yan, and Charles LA Clarke. 2021. Predicting efficiency/effectiveness trade-offs for dense vs. sparse retrieval strategy selection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2862–2866.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Leonard Richardson. 2007. Beautiful soup documentation. *April*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.