End-to-end NLP System Building: Retrieval Augmented Generation

Chuangji Li [*] Shizhuo Li[*] Alan Wang[*]
{chuangjl, shizhuol, minyangw}@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA, 15213

Abstract

For this project, we build an end-to-end retrieval augmented generation (RAG) system that is capable of answering questions related to Carnegie Mellon University (CMU) and its Language Technologies Institute (LTI). Our system involves a embedding model, a reader model, a retriver, and a re-ranking model. We experimented with multiple language models and selected a optimized solution for our embedding, reader, and re-ranking model. We tested our system with self-annotated question-answer pairs and proved our system's effectiveness.

1 Introduction

In recent years, the development of end-to-end retrieval augmented generation (RAG) systems has garnered significant attention in natural language processing (NLP) research. These systems aim to seamlessly integrate information retrieval and generation capabilities to effectively answer user queries. In this paper, we present our endeavor to build an RAG system tailored for addressing questions related to Carnegie Mellon University (CMU) and its Language Technologies Institute (LTI).

Our system is meticulously designed, comprising various components meticulously designed, including an embedding model, a reader model, a retriever, and a re-ranking model. Each component plays a crucial role in ensuring the system's effectiveness and efficiency in retrieving and generating accurate responses to user queries.

Our approach leverages state-of-the-art models such as the `gte-large` embedding and the `Mistral-7B-instruct-v0.2` reader, chosen based on their performance and suitability for our task. By meticulously curating a test set from data manually scraped from CMU and LTI websites, we evaluate our system's performance, achieving a precision of 0.7463 on our test set.

In this introduction, we provide an overview of the motivation behind our project, the architecture of our RAG system, the data sources utilized, and the evaluation methodology employed. Subsequent sections delve deeper into each aspect, providing insights into our design choices, experimental results, and analysis of the system's performance.

Through this work, we contribute to the advancement of NLP systems tailored for domain-specific question answering, with implications for educational institutions and beyond.

2 Data

2.1 Knowledge Source and Raw Data

The compilation process of knowledge source was strictly followed and included all recommendations in the project description under the section "Preparing raw data". Collected data includes: faculties of LTI and their research papers, CMU schedule of classes in the 2023-2024 calendar year, CMU academic calendar in 2023-2024, 2024-2025 calendar year, academics such as program details in LTI, events in CMU such as spring carnival, reunion week, commencement, and history of CMU and SCS, including CMU fact sheet, 25 Great things in SCS.

The data was retrieved from the knowledge sources by different tools and Python scripts. For PDF document knowledge sources, data is extracted into a plain text file by pypdf[1]. HTML pages are extracted in plain text by beautifulsoup4[2]. Research paper information from LTI faculties was extracted with Schematic Scholar API[3].

[*] All members have equal contribution, alphabetical order

[1] https://github.com/py-pdf/pypdf

[2] https://pypi.org/project/beautifulsoup4/

[3] https://www.semanticscholar.org/product/api

2.2 Data Processing

After gaining the raw data as text files, we separated the raw documents into two types: line-based document and file-based document. For line-based documents, including schedule of classes, academic calendars, LTI faculty list and research papers, are then parsed from plain text with keywords and features extracted, followed by conversion into sentences with connective words stressing the keywords and key information as processed data. The file based documents are re-formatted and directly prepared as processed data.

The motivation of having raw documents categorized into two types is that for line base documents, for example in schedule of classes, each course is independent from others but contains a lot of key features and metadata. The course identifiers are quite similar across multiple courses. If multiple courses appear in one chunk of text, the courses' identifiers, that are course numbers, will be hard to distinguish after embedding, causing difficulties in retrieving. Thus, we made the schedule of classes line based after processing, that is each line in the text contains only one class metadata, for convenience of splitting it into small documents. (Each line will be splited into a single documents). To make larger distinction between line documents and make easier for retriever to accurately retrieve correct course document, we emphasized course identifiers by repeating course numbers multiple times. Similar method are applied on other line based documents based on our categorization.

2.3 Annotation & Quality

For data annotation, we adopt the idea of an experimental ANOVA design[4] here and find that in order to achieve an effect size of 0.5 and a power of 0.95 under a 0.05 significance level, for differentiating the effectiveness of two models, we need at least 27 samples of question and answer pairs for each model. Therefore, we manually created 55 question and answer pairs, with each of us annotating a subset of the data according to the category of the related documents. Since during the data scraping part, we have 8 categories based on the documents used, we created around 7 questions per category and labeled them. For the train test split, we ulitize such labels so that there are approximately equal numbers of question and answer pairs for each category in the training and in the testing set.

During the annotation, we consider a variety of questions: first, we focus on the question type - namely, how, what, why, where, when, who questions. We then consider symmetrical question answer pairs to ensure that the model does not only learn in one direction. For instance, both "Who is teaching 10711 in Spring 2024?" and "What class is Prof. Neubig teaching in Spring 2024?" are included.

Lastly, we take the sample questions created by other members, and without seeing the annoated answer, we go through the documents, write down a new answer and then compare it with the previous one. Since all the answers could be easily located in the documents, without considering the syntax of the English language, we achieved an almost perfect match in this process, with the Cohen's kappa statistic of 0.9.

We also considered using Large Language Models (ChatGPT) to generate question pairs. It can generate a large quantity of question and answer pairs which is more capable than a human annotator. However, it inherits some shortcomings: first, our current approach does not involve any finetune of the model weights of either the embedding model or the reader model, having a large number of training pairs do not help with our model performance. Some answers to those questions are long that our metrics of evaluation (f1, precision, recall) scores weren't reflecting true accuracy of answer.

3 Architecture

In this section, we discuss the models we deployed in the RAG system. The justification of choosing over these models are discussed, and their statistics.

3.1 Embedding Models

The potential embedding models are bge-large-en-v1.5(Xiao et al., 2023)[5], gte-large(Li et al., 2023) [6], and UAE-Large-V1(Li and Li, 2023)[7]. We choose these models based on the following criterion:

Simplicity: Complicated models are less favored; we prefer models with less layers and parameters, hence smaller size.

[4] https://www.rdocumentation.org/packages/pwr/versions/1.3-0/topics/pwr.anova.test

[5] https://huggingface.co/BAAI/bge-large-en-v1.5

[6] https://huggingface.co/thenlper/gte-large

[7] https://huggingface.co/WhereIsAI/UAE-Large-V1

Model	# Params	Architecture	Dimension	Max Seq Length
bge-large-en-v1.5	335M	BERT	1024	512
gte-large	335M	BERT	1024	512
UAE-Large-V1	335M	BERT	1024	512

Table 1: Statistics of Embedding Models of Selection

Model	# Params	Architecture	Max Context Length
Mistral-7B-instruct-v0.2	7.24B	Mistral	4096
Llama-2-7b-chat-hf	6.74B	Llama2	4096

Table 2: Statistics of Reader Models of Selection

Architecture: To better compare the embedding models while limiting the effects of architectures, all the models are BERT-based.

They all have the same number of embedding dimension and maximum sequence length. We also avoid complicated fine-tuned variations of them and use those offered by the original developers

Performance: These models all used to be the top of the MTEB Leaderboard. Plus, they all have relatively high performance in the retrieval tasks. Their scores are 54.29, 52.22 and 54.66(out of 100), respectively.

Table 1 summarized the statistics of these models. For more information, please refer to MTEB LeaderBoard.

3.2 Reader Models

The potential reader models are Mistral-7B-instruct-v0.2(Jiang et al., 2023)[8] and Llama-2-7b-chat-hf(Touvron et al., 2023)[9]. We choose these models based on the following criterion:

Simplicity: With the same reason above, complex models are less favored. We limit our choice to models with less or equal to 7B parameters. We also avoid complicated fine-tuned variations of them and use those offered by the original developers

Performance: These models have been proven to have great potentials in the field of text-generation by numerous corporations.

Table 2 summarized the statistics of these models. For more information, please refer to MTEB LeaderBoard.

3.3 Retriever

We use FAISS as database and retriever. Faiss is a library for efficient similarity search and clustering of dense vectors. The documents were first embedded with embedding models. Documents are split using recursive text splitter based on embedding models. The processed documents then were used by FAISS to build a dense database. Retriever was built based on cosine similarity.

3.4 Re-ranking Model

After the retriever retrieves the top-k most relevant documents, a cross-encoder based re-ranking model was used to re-rank these documents. It jointly encode both queries and documents using neural model. The model precludes approximate nearest neighbor lookup, so can only be used on small number of candidates. The re-ranking model that we use is Colbertv2.0(Santhanam et al., 2022)[10]. It is a BERT-based model that are small yet robust.

3.5 The RAG Pipeline

Finally, the RAG pipeline combines all the aforementioned components together. When query comes as input of RAG pipeline, the retriever will retrieve the most relevant documents to the query. The re-ranking model re-ranks these documents and output a small subset of them. This subset is passed to the reader model as context to generate response.

4 Experiment

We have proposed 3 promising embedding and 2 reader models in the previous section, we now choose the best combination of embedding models and reader models to build our RAG system.

[8] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2

[9] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[10] https://huggingface.co/colbert-ir/colbertv2.0

Embedder	Reader	Precision	Recall	F1-Score
bge-large-en-v1.5	Mistral-7B-instruct-v0.2	0.6903	0.7725	0.6914
gte-large	Mistral-7B-instruct-v0.2	**0.7491**	**0.8284**	**0.7463**
UAE-Large-V1	Mistral-7B-instruct-v0.2	0.7123	0.7864	0.7029
bge-large-en-v1.5	meta-llama/Llama-2-7b-chat-hf	0.2195	0.5841	0.2525
gte-large	meta-llama/Llama-2-7b-chat-hf	0.2545	0.6165	0.2852
UAE-Large-V1	meta-llama/Llama-2-7b-chat-hf	0.2263	0.6233	0.2675

Table 3: Performance Evaluation for Different Embedding and Reader Models

We use R to denote reference answer and G to denote generated answer.

$$\text{precision} = \frac{|R \cup G|}{|G|} \tag{1}$$

$$\text{recall} = \frac{|R \cup G|}{|R|} \tag{2}$$

$$\text{F1} = \frac{2 \times (\text{precision} \times \text{recall})}{\text{precision} + \text{recall}} \tag{3}$$

For generation task, these metrics are computed as follows:

5 Result

The result of the experiment is shown in Table 3. We computed the precision, recall, and F1-score of the 6 RAG models. The combination of gte-large + Mistral-7B gives the best results across all metrics. The precision is 0.7491, the recall is 0.8284, and the F1 is 0.7463. The RAG system is therefore built with such combination.

5.1 Significance Test

To better evaluate the performance of our RAG system. We run a significance test between two RAG system. Since the difference among embedding models are to small, we choose to vary our reader model. We therefore choose Llama2 as the base reader model, and choose the best the combination based on Llama2, which is Llama2 + gte-large.

The result of the significance tests shows that our model indeed has better performance. In terms of precision and F1-score, our RAG system triumphs in all of 100 simulations, and only loses 2 times to base model in terms of recall. This means that our system is superior in terms of precision with p-value of 0.0; our system is superior in terms of recall with p-value of 0.02; and our system is superior in terms of F1 score with p-value of 0.0.

6 Analysis

To better analyze the performance of the model, we analyze of the generated output. We also compare the performance of closed-book model and our RAG.

6.1 Qualitative Analysis

To learn more about the model's performance, we will look at the accuracy of the response to each type of questions. We discovered that the model behaves almost equally good across all types of question. We enhanced the performance of line-based questions by splitting each line into an independent documents. The most significant factor that determines the accuracy of the RAG system is the retriever rather than the reader.

Some examples of QA pair are presented in Appendix. We can see that, when given correct documents, the model has no problem answer the question correctly. We have implemented few-shot learning so that the model has the similar output as the reference answer, though sometimes the model tends to generate full sentences. If the context provided is no sufficient, the model explains such fact, as shown in the first example in the Appendix.

6.2 Closed-Book Model vs RAG

To show that the RAG pipeline is indeed significant, we also compare it with its closed-book version (i.e. reader model alone). We use the same metrics as the previous procedures. The results are demonstrated in Table 4. The differences in the performances is large enough for us to say that the RAG system is significantly better than closed-book model.

Model	Precision	Recall	F1-Score
Closed-book	0.0090	0.1846	0.0162
RAG	**0.7491**	**0.8284**	**0.7463**

Table 4: Performance of Closed-book versus RAG

7 Conclusion

The project builds an end-to-end retrieval augmented generation (RAG) system, which utilizes several state-of-the-art models and techniques to address domain-specific questions related to Carnegie Mellon University (CMU) and its Language Technologies Institute (LTI). The combination of the gte-large embedding model with the Mistral-7B-instruct-v0.2 reader model emerged as the most effective model, achieving a F1-score of 0.7463.

8 Knowledge Source

1. Carnegie Mellon University-Schedule of Classes
https://enr-apps.as.cmu.edu/open/SOC/SOCServlet/completeSchedule
2. Carnegie Mellon University-Academic Calendar 2023-2024 School Year
https://www.cmu.edu/hub/calendar/docs/2324-academic-calendar-list-view.pdf
3. Carneige Mellon University-Academic Calendar 2024-2025 School Year
https://www.cmu.edu/hub/calendar/docs/2425-academic-calendar-list-view.pdf
4. List of Faculties at Carnegie Mellon University Language Technology Institute
https://lti.cs.cmu.edu/people/faculty/index.html
5. Research Paper by faculties at Carnegie Mellon University Language Technology Institute retrieved by
https://www.semanticscholar.org/product/api
6. Academics at Carnegie Mellon University Language Technology Institute
https://lti.cs.cmu.edu/academics/index.html
7. Carnegie Mellon University Language Technology Institute Program Handbooks
https://lti.cs.cmu.edu/academics/phd-programs/files/handbook_phd_2023-2024.pdf
https://lti.cs.cmu.edu/academics/masters-programs/files/mlt-student-handbook-2023-2024.pdf
https://lti.cs.cmu.edu/academics/masters-programs/files/miis-handbook_2023-2024.pdf
https://lti.cs.cmu.edu/academics/masters-programs/files/mcds-student-handbook-2023_2024.pdf
https://lti.cs.cmu.edu/academics/masters-programs/files/handbook-msaii-2022-2023.pdf
8. Carnegie Mellon University Spring Carnival and Reunion Week
https://web.cvent.com/event/ab7f7aba-4e7c-4637-a1fc-dd1f608702c4/websitePage:645d57e4-75eb-4769-b2c0-f201a0bfc6ce?locale=en
9. Carnegie Mellon University Commencement
https://web.cvent.com/event/ab7f7aba-4e7c-4637-a1fc-dd1f608702c4/websitePage:645d57e4-75eb-4769-b2c0-f201a0bfc6ce?locale=en
10. Carnegie Mellon University School of Computer Science 25 Great Things
https://www.cs.cmu.edu/scs25/25things
11. Carnegie Mellon University Fact Sheet and History
https://www.cmu.edu/about/cmu_fact_sheet_02.pdf
https://www.cmu.edu/about/history.html
12. Carnegie Mellon University Buggy and History
https://www.cmu.edu/news/stories/archives/2019/april/spring-carnival-buggy.html
13. Carnegie Mellon University Athletics
https://athletics.cmu.edu/athletics/tartanfacts
https://athletics.cmu.edu/athletics/mascot/about
https://athletics.cmu.edu/athletics/kiltieband/index

References

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Xianming Li and Jing Li. 2023. Angle-optimized text embeddings. *arXiv preprint arXiv:2309.12871*.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. Colbertv2: Effective and efficient retrieval via lightweight late interaction.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding.

Appendix

Question	Generated	Reference
When is does the class of 90978 meet in Spring 2024	The document does not provide information about the meeting dates for course 90978 in Spring 2024.	Days not yet decided
When is last day of classes in Spring 2024	April 26, Friday.	April 26, Friday
What is Daniel Fried's research area interested?	Natural Language Processing, Information Retrieval, Conversational AI, Intelligent Agents, Dialogue, Discourse and Pragmatics, Multimodal AI.	Natural Language Processing, Information Retrieval, Conversational AI, Intelligent Agents, Dialogue, Discourse and Pragmatics, Multimodal AI.
Who is the instructor of 45907 being held in Fall 2023	Markovitz	Markovitz
In the paper ChatGPT MT: Competitive for High (but Not Low-) Resource Languages, what is the primary factor determining ChatGPT's relative ability to translate a language?	The primary factor determining ChatGPT's relative ability to translate a language is its resource level.	A language' s resource level.
Whom should applicants contact about MIIS program for specific application/admissions questions	Contact Brianna Eriksen, Academic Program Manager - MIIS, at bfreema2@andrew.cmu.edu or 412-268-4277.	Brianna Eriksen
What did Andrew Carnegie believe when he was rich?	Andrew Carnegie believed that "to die rich is to die disgraced."	To die rich is to die disgraced.