Medical Operations RAG and Evaluation Dataset

Benjamin Lambright

May 6, 2025

Abstract :

I wrote an RAG system using a deepseek reasoning model for information extraction on medical operation data. I then evaluated my system on some toy evaluation datasets I annotated myself.

Index Terms: RAG, LangChain, medical operations

1 Introduction

After discussing ways in which RAG or LLMs could be effectively be used in medicine with some of my premed friends, we determined that having an LLM that was specifically an expert in preparing for medical operations would be useful. The RAG system I propose is capable of providing relevant information on an operation given information about a patient, their diagnosis, and a recommended procedure. It can summarize the operation, determine which operation is best for the patient, and what materials are likely necessary for the operation. There are many publicly available chatbots for answering medical questions for patients online, as well as tools for summarizing medical publications. Within this project, I wanted to specifically go into the niche of providing relevant information to help medical professionals prepare for an operation, given similar operations that the model is aware of.

2 Dataset

I used a publicly available dataset on Kaggle. While there is little information on the dataset itself, the data aligns with other datasets of operations and doctors' notes. Specifically, this dataset includes doctors' notes on operation reports, radiology reports (only text, no images), cardiology, and pulmonary reports. While the contents of the notes vary, they typically are sub-sectioned with capitalized words, and include information on the diagnoses, relevant patient information, and a description of the procedure executed by the doctor. These subsections are not standardized, and generally vary significantly from report to report. Sometimes, additional information is included in the subsections, such as materials or anesthesia use. Finally, as far as I could tell, the original purpose of this dataset was for a topic modeling project.

2.1 Preprocessing

My RAG model uses LangChain's document object, which means that the data needs to be organized into key-value pairs. To handle this, I had two preprocessing techniques, creating two datasets to assess the quality of the data. In one, I only split the data by their main categories: surgery, radiology, or cardiovascular/pulmonary. In the other dataset, I added metadata on each of the subcategories in the doctors' notes. In order to do this, I first split the data by capitalized words using a RegEx expression. Then, for each of these splits, I combined categories titled with compound words, such as "POSTOPERATIVE DIAGNOSIS." In this way, I was able to get key-value pairs of each of the subsections within the notes to see if this would help a RAG system retrieve necessary information. I called these two datasets my naive and verbose datasets, respectively. Once these two datasets had been built, I used HuggingFace's sentence-transformers/all-mpnet-base-v2 encoder model. Then, I stored the embeddings from this encoder model using LangChain's InMemoryVectorStore class. I hypothesized that because my dataset is relatively small, it is not necessary to use FAISS (which is arguably the industry standard for RAG right now), and this might actually run faster because it is on memory. Once I added all of my documents to each relevant dataset, I pickled the two datasets.

3 RAG Methodology

My actual RAG model is just just two class objects, a State object, and the Retriever object. Firstly, for this model, I use Together.ai's chat model ChatTogether, because it is built in the same way as OpenAI's chat model, but has options for using free models remotely. From there, I used DeepSeek's Distill Llama with 70B parameters, because I thought a reasoning model would be a valuable step for considering what information to extract from the dataset to answer prompts. Because I wanted the "production-ready" model to be relatively deterministic, I chose a temperature of 0.0. Finally, I pulled a chat prompt template from LangChain's hub.

The first class I built defines the state graph for the workflow of generating a response from the model. In my case, the state graph is always START –> "retrieve", so that the model can attempt to retrieve data before generating an output. In the second class, which is the main class for the RAG System, I initialize the LLM and the aforementioned state graph. Within the retriever function, I simply do a similarity search on the question over the vector store of my dataset, and return the answer within a "context" key. Finally, within the generate function, I join the context with the question in order to invoke an answer from the LLM, and, because I'm using a DeepSeek model, I ensure that only the answer and not the reasoning is included in the final output[1].

4 Evaluation

In order to evaluate the quality of my RAG system, I hand annotated 3 QA datasets, each with 10 questions and answers. The first data set is called materials, and it prompts the model with the name of the procedure. With this information, it asks the model to list all the materials used in the operation, or a yes or no question related to if and how a material was used. For the other two datasets: pro-

[1] Importantly, if the model fails to determine an answer, it outputs all of its thinking instead

cedure and operation, I used ChatGPT 4o to summarize the patient data and diagnosis. This is because I wanted to test if the model was able to reason which operation was most similar, without being able to directly identify the exact words in the prompt. I hoped that by prompting 4o, the cosine similarity would remaine similar [2]. With this summarized information, in the procedure dataset, I ask the model to state the name of the recommended operation for the patient and their diagnosis. For the operation dataset, I ask the model to describe the operation, including the any materials used [3].

In order to evaluate the inferences made by the model on these QA datasets, I used BLEU score and ROUGE. However, recognizing that these evaluation metrics are not great for evaluating LLM outputs on QA datasets, I tracked the row in my original Kaggle dataset in which each question was made. Then, when evaluating the model, I checked to see if the model only extracted information from that row. For each question in the dataset, the question either scored a 1 if it provided a complete answer to the question and only mentioned information from the correct doctor's note. Otherwise, the answer scored a 0. While this binary metric does not perfectly evaluate the answers, it ensures that there are at least no false positives in the answer, which I think is the most important thing when evaluating a medical retrieval model.

Naive Dataset	BLEU	ROUGE	Human Eval
Materials	13.0	46.6	90.0
Procedure	3.0	11.1	40.0
Operation	4.4	29.0	70.0
Verbose Dataset	BLEU	ROUGE	Human Eval
Materials	12.1	40.0	90.0
Procedure	5.7	19.0	60.0
Operation	1.7	28.8	70.0

Table 1
Benchmarking my model on 3 evaluation datasets - testing the models ability to retrieve the materials for an operation (Materials), testing the ability of the model to suggest a procedure (Procedure), and testing the ability of the model to describe an operation (Operation). Each of these evaluation datasets were tested on my two retrieval datasets: naive and verbose (See 2.1).

4.1 Results

From the results above, I found that the verbose dataset was not that necessary, and in some cases actually counter productive. According to the BLEU and ROUGE scores, the naive dataset performed better for the materials and operation dataset, and this was clear to the human eye with the operation dataset. The summary of the operation was clearly more complete, and provided better information of the materials used in the operation. I think this is because the additional metadata in the verbose dataset actually caused the model to not always review all sections of the medical notes, preventing the answers from the verbose dataset from being as complete. That being said, it seems like including the additional metadata was useful for the procedure dataset. This is likely because almost every doctor's note had a subsection titled "PROCEDURE" that only included the name of the procedure. This made it much easier for the model to extract the correct name of the procedure, which is indicative with the human eval score being 60%, instead of 40%. Because of this, it seems like the verbose dataset was useful when attempting to extract information that can be easily found in a subsection of a doctor's note, but when it is necessary to review the entire doctor's note, including the metadata can cause the model to hyperfocus on certain sections.

Upon review of the errors in the dataset, I found that answers for the materials dataset were always correct for the human eye, except for one true/false question that the model consistently said it did not know the answer to. This question was more complex, and was asking if it was necessary to use a certain tool for an operation, which seems to be out of scope of this model in cases where the model does not know the tool. The answer was no, because the tool was not used, but the model couldn't say that because it had never heard of the tool before. In terms of the other two datasets, I found that the errors here were because the model gave false recommendations. In one case, the model suggested giving a young, healthy woman a C-section, when she instead had a natural birth. In another case, the model suggested giving an obese woman a bariactric surgery, when that was not necessary nor in the interest of the patient. This might suggest that the model can be too quick to select a more dramatic, extreme operation given the data. It might also suggest some kind of bias. In another case, the model suggested the right operation, but gave the operation a different name from the gold text that means the same thing. For all of these examples, I think the model can sometimes have trouble reasoning what operation to suggest when there are multiple notes that fit the description within the prompt. Just like a real doctor, it probably needs more information to more reliably determine the procedure.

5 Conclusions and Future Work

The obvious next step for this work is more rigorous benchmarking on possible tasks for the model. I think the materials task is particularly interesting, as it was suggested to me by a medical professional as a possible task for LLMs that is less likely to cause a significant accident if the model is incorrect. That said, there were few notes in this dataset that explicitly described which tools and materials were used in the operation, so it would require either more extensive annotation or scraping to generate more questions prompting the model to generate a complete list of materials to use for an operation. However, I think that is a very interesting task, and I think consulting additional medical professionals would be valuable to determine additional next steps.

Ultimately, from my own annotation, I found that this model was effective at determining which materials should be used on an operation that can be found in it's vector database, as well as summarizing these operations. However, I would not suggest using this RAG system to determine if a user needs a surgery, or exactly how this surgery should go, because it lacks the data to consider unknown operations and patients and could potentially give dangerously false information to even a patient it does know about. Instead, I would suggest using this program to help medical professionals see how certain surgeries have been done before,

[2] I recognize that both me and ChatGPT are not medical professionals, so while I tried to ensure that summarizing the patient data didn't change any of the factual information about the patient, I can't ensure that this doesn't change any of the medical data

[3] These prompts can be found within these evaluation datasets in my model code

given information on the patient and their diagnosis.