Open Domain Approach to answer Medical Questions through Wikipedia and Multi-hop Reasoning

Vinay Kumar
Department of Electrical Engineering, Media, and Computer Science
Ostbayerische Technische Hochschule Amberg-Weiden
Amberg, Germany
Technical Report: Natural Language Processing - June, 2025

Abstract

Medical question answering has always been an challenging because it requires both domain expertise and complex reasoning abilities. MedQA-USMLE is one of the unique and challenging biomedical question answering tasks as it's question is mostly consist of multi-hop reasoning and to solve this one should also have in-depth domain knowledge. Some start-of-the-art the new Gen Large Language models, such as 'DeepSeek-v3' and 'Gemini-2.5-Pro' that have been trained for reasoning tasks shows remarkable performance. Medium-sized language models and small language models (SLMs) still struggle with these kind of certain tasks. To experiment and try to bridge the gap, I have tried to introduce the fusion of Multi-hop reasoning with Medium-sized language models. Specifically I tried to categories the MedQA- USMLE questions into different types so facilitate information retrieval from Wikipedia related to that question, as each question require different strategy to get answer. To enable best performance of the Medium-sized language models I utilized pre-trained biomistral model for categorizing the question using N-shot strategy, and for text generation Phi-3-mini-4k-instruct model, as they both models were trained for reasoning and on different medical dataset including MedQA-USMLE. My findings demonstrate that targeted knowledge multi-hop strategies can significantly enhance Medium-sized language models medical reasoning capabilities, making specialized medical question answering more accurate and more reasonable.

Keywords- N-shot, Multi-hop reasoning, Wikipedia corpus, medical question answer

1 Introduction

Question Answering (QA) or nowadays Retrieval Augmented Generation (RAG) systems has now became the fundamental task in Natural Language Processing (NLP), which combine with information retrieval with a Large Language Model (LLMs) to generate more accurate and reliable answers. And to promote constant improvement in biomedical question answering (BQA) systems, the biomedical NLP community is being dedicated to introduce new terms and techniques to solve these kind od complex reasoning types systems (Tang and Yang, 2024).

The MedQA-USMLE task stand out as particular challenging (Jin et al., 2020) task. As this dataset has rapidly become an essential benchmark for large language models (LLMs) (Chen et al., 2025) because the professional medical board exams in the USA, it is designed to check the physicians skills and knowledge, especially the clinical decision-making ability. And the dataset contains each and every-part of the medical field and knowledge that can be used for the skill testing, as this also make dataset suitable for evaluating models potential in real medical scenarios.

The complexity of the dataset: Over 90% of its questions fall into criteria of multi-hop open domain QA, presenting difficulties for BQA systems. And to excel this type of system, extensive medical domain knowledge is required, while precise knowledge recall, chain-of-thought and strong multi-hop reasoning capabilities are the key.

Affirmatively, BQA systems are constantly making great progress on the MedQA-USMLE task as they get also evolved alongside some latest NLP paradigms. Initially (Jin et al., 2020) continued to use information retrieval (IR) - a traditional machine reading pipeline approach. Despite using BioBERT as one the pipleine, the system could achieve 34.1% accuracy. (Jin et al., 2020) illustrated this poor performance to the IR module's as they were incapable of conducting multi-hop reasoning in the evidence retrieving process. Afterwards, more advanced biomedical pre-trained models were introduced, when have the potential to surpass the baseline approaches BioMedLM (Bolton et al., 2024) which made impressive improvement with 40%. So these improvement arise from biomedical PLMs when capturing rich medical knowledge during pre-training, and fine-tuning enables transfer of this knowledge. Nevertheless these results remain unsatisfactory, indicating the lack of multi-hop reasoning capability in biomedical PLMs.

With NLP embracing the Pre-train+MultiHopPrompt+Predict patterns, LLMs demonstrated remarkable performance on the MEdQA-USMLE task. Codex (Chen et al., 2021) reached passing scores 60.2%, while Med-PaLM 2 (Singhal et al., 2023) achieved an remarkable 86.5%, nearly matching human medical experts. This success stems from two key factors: these massive models store enormous amounts of knowledge, and they can perform complex reasoning through chain-of-thought prompting. However, this reasoning ability only emerges in models with over 100 billion parameters.

The downside? These giant models come with serious drawbacks: 1) They're extremely expensive to build and run, putting them out of reach for most organizations. Updating them with new medical knowledge is also costly and slow. 2) They raise security concerns when handling sensitive medical data. This is where smaller language models (SLMs)- those with under 7B parameters and Medium-sized language models those with under 10B offer advantages. They're more affordable, easier to update with new information, and can be deployed locally for better security with sensitive medical data.

So now the question is - can we somehow give these smaller and medium sized models the same multi-step reasoning abilities as their larger counterparts, making them viable alternatives for medical exam tasks without the drawbacks?

To address the limitations of Medium-sized language models in medical question answering, I've developed a Wikipedia-based retrieval and reasoning based prompt for breaking down the complex question in simple question. This approach bridges the performance gap between resource-intensive large language models and more practical smaller models. My system works by first extracting relevant medical information from Wikipedia articles, chunking this content into manageable segments, and creating vector embeddings for efficient retrieval. When presented with a medical question, the system:

1. First, I use a specialized n-shot prompting technique to categorize medical questions into specific domains (USMLE, 2021) using BioMistral-7B
2. I extract relevant medical information from Wikipedia articles based on these categories and metamap_phrases (Jin et al., 2020) (keyword extracted from question), chunk this content, and create vector embeddings.
3. When presented with a question, the system retrieves the most relevant knowledge chunks using FAISS vector similarity search.
4. It then constructs a comprehensive context from these chunks.
5. Finally, it uses a chain-of-thought prompt template to guide the model through multi-step reasoning

The key innovation is the n-shot prompting strategy to categories the question into various category and using domain categorization to improve retrieval relevance, and implementing structured chain-of-thought prompts to enable medium-sized models to perform multi-hop thinking they couldn't achieve on their own. The experimental implementation demonstrates that even compact models like Phi-3-mini (Abdin et al., 2024) can effectively tackle complex medical reasoning tasks when supported by this retrieval framework and guided by carefully designed prompts.

All open source models are accessible on BioMistral[1] and Phi-3[2]

2 Related Work

Question answering (QA) has evolved significantly from text-dependent tasks requiring models to comprehend given texts to open-domain QA (OpenQA), where models must both find and understand relevant context. Early OpenQA approaches (Chen et al., 2017)

[1] https://huggingface.co/BioMistral

[2] https://huggingface.co/microsoft/Phi-3-mini-4k-instruct

used wikipedia text retrievers with reading comprehension models, while subsequent research introduced more sophisticated techniques to aggregate evidence or filter irrelevant texts . Most initial work focused on reasoning-based answers from retrieved documents (Chen et al., 2025), which could be solved with multi-hop reasoning . To promote advanced reasoning skills, researchers developed free-form multiple-choice OpenQA tasks using real-world exams questions. In the medical domain, several datasets emerged with varying characteristics, including LiveQA (Ben Abacha et al., 2017) and the most challenging dataset (Jin et al., 2020) each with different question sources and annotation methods, though few were formulated as OpenQA problems requiring complex reasoning.

To enhance multi-hop reasoning capabilities in language models, researchers have explored various pre-training approaches. Some methods leverage knowledge graphs (KGs) to provide structured background information, as demonstrated by DRAGON (Yasunaga et al., 2022a) and (Chen et al., 2025). Alternatively, researchers have constructed reasoning-rich datasets with specific supervision signals, such as phi-3 (Abdin et al., 2024), which transfers reasoning knowledge from bio experts and has been trained on various biomedical datasets. LinkBERT (Yasunaga et al., 2022b) represents a significant advancement by exploiting document link information, showing particular efficacy in multi-hop QA tasks. However, it still under-performs compared to larger language models, potentially because it wasn't exclusively developed for biomedical QA and only models document links, missing other effective connections for medical reasoning.

The most promising approach is (Chen et al., 2025) they used MHMKI in-corpus data retrieved form Wikipedia dumps, adapts LinkBERT specifically for biomedical question answering, with experimental results demonstrating its effectiveness across various tasks. Ablation studies revealed that single-hop medical knowledge significantly improved performance, while two-hop and three-hop knowledge sometimes had counterproductive effects, based on task requirements. For multi-hop biomedical QA tasks, MHMKI showed consistent gains across various models, with GPT-Neo benefiting most from integration and MedQA-USMLE showing the largest improvement. Notably,

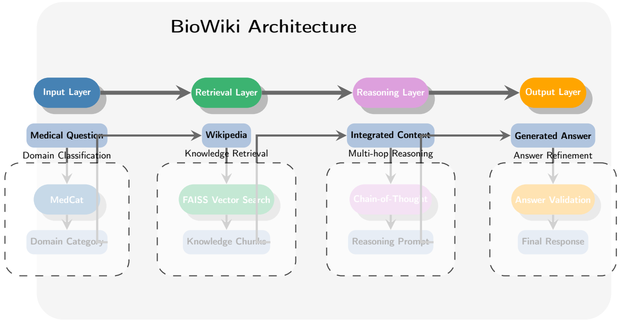

Figure 1: The BioWiki architecture for medical question answering combines domain-specific categorization, efficient knowledge retrieval, and chain-of-thought reasoning to enable language models to perform complex medical reasoning tasks.

BERT models enhanced with MHMKI demonstrated competitive or superior performance compared to much larger models, with BioLinkBERT+MHMKI reaching 45.59% accuracy on MedQA-USMLE, surpassing LLaMa-6.7B at 44.54%. These results were quite promising for as BERT models remain highly competitive for biomedical QA tasks.

3 My Work

In this section, I present the modules and methodology that facilitated the construction of this project. My approach adapts four strategy : 1) Categorization of questions; 2) Retrieval of information for each question; 3) Dividing into chunks and storing into vector database for future use; 4) Infusing the multi-hop prompts into my model. In the subsequent sections, I will elaborate on each step.

3.1 Categorization of question

As stated (Jin et al., 2020) categorized the MedQA-USMLE questions into type and type 2, where type 1 was direct short question which means that it doesn't require reasoning, it is just direct question and answer, while type 2 questions simulate real-word clinical problems by studying the patience case and necessitate for multi-hop reasoning. Although this categorization provides valuable insight, but it failed to give specific details about specific reasoning types of questions and the multi-hop knowledge involved in it.

And therefore, the authors (Chen et al., 2025) further categorized type 2 questions into more defined reasoning types. They discovered that when looking for evidence to answer these complex questions within their medical text database, there's typically a shortest path to the answer. Most of the time, this path stays within one disease-focused article - the patient's symptoms match information in the "Signs and symptoms" section, while the actual question asks for details from sections like "Cause," "Management," or "Mechanism." The authors created a more detailed breakdown of type 2 questions by analyzing 100 randomly selected questions. They found that most questions can be answered using at most three pieces of evidence from up to two medical articles, leading to three subcategories:

Type 2a questions ask for diagnosis directly (one reasoning step: symptoms → disease).

Type 2b questions skip diagnosis and ask about treatment/causes/mechanisms (two steps: symptoms → disease → management/cause).

Type 2c questions require deeper knowledge from linked articles (three steps: symptoms → disease → specific aspect → linked article details).

Their analysis showed type 2b questions comprise 69.7% of cases, despite being overlooked in prior research. The authors focus on modeling multi-step knowledge connections within medical articles to handle these question types efficiently.

My Approach - firstly I tried to follow up with (Chen et al., 2025) and (Yasunaga et al., 2022a) what they used is knowledge graph technique to configure the reasoning types question, I followed up and in the end, I came to my type 1 question and all the questions in my dataset was categorized as type 1, so I tried alternative approach for to categorize in specific medical domain instead of categorizing into different reasoning types. I simply used the type of category domain defined by USMLE officials (USMLE, 2021) as it contains 18 categories of medical field and from those 18 categories, I picked up some pretty common 21 sub-medical categories and defined them.

After manually defining the categories, I employed BioMistral (Labrak et al., 2024), a model pre-trained on the MedQA dataset, for text generation. To adapt it for text classification, I utilized an n-shot learning approach, a family of zero-shot classification (STATWORX, 2022). In my case it is 21 shot text classification.

I utilized only 1000 of questions from the entire dataset for categorizing, I had pre-defined my categories and let my model predict which question belong to which category using the keyword terms given in the column name *metamap_phrases*. I also tried this approach with small language models but the result was not up-to the mark.

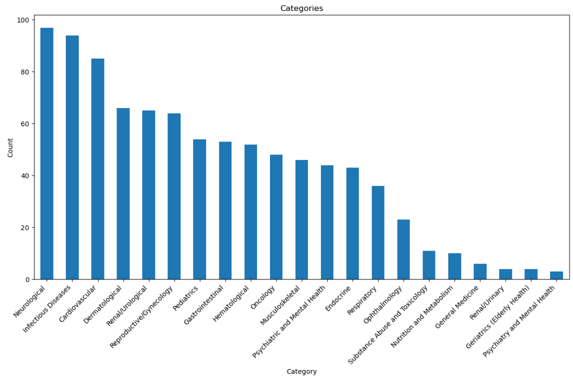

Figure 2: Categorized questions

As a result of doing this approach I also got some result of category which were not defined and that added to the new category. Fig.3

Question:
A 40-year-old zookeeper presents to the emergency department complaining of severe abdominal pain that radiates to her back, and nausea. The pain started 2 days ago and slowly increased until she could not tolerate it any longer. Past medical history is significant for hypertension and hypothyroidism. Additionally, she reports that she was recently stung by one of the zoo's smaller scorpions, but did not seek medical treatment. She takes aspirin, levothyroxine, oral contraceptive pills, and a multivitamin daily. Family history is noncontributory. Today, her blood pressure is 108/58 mm Hg, heart rate is 99/min, respiratory rate is 21/min, and temperature is 37.0°C (98.6°F). On physical exam, she is a well-developed, obese female that looks unwell. Her heart has a regular rate and rhythm. Radial pulses are weak but symmetric. Her lungs are clear to auscultation bilaterally. Her lateral left ankle is swollen, erythematous, and painful to palpate. An abdominal CT is consistent with acute pancreatitis.

Predicted Category: The most appropriate category for this medical question is **"Toxicology"**. The patient's symptoms are consistent with acute pancreatitis

Figure 3: Example of a question with its predicted category which was not defined.

3.2 In-domain Text and Web Chunking

I used Wikipedia as the knowledge source for building my text corpus. Specifically, I collected all Wikipedia documents that fit within my category and and question. As a result i scrapped and store the content obtained by for further processing My implementation uses a specialized crawler that extracts medical content from Wikipedia articles. I developed a system that retrieves Wikipedia pages based on medical phrases extracted from USMLE questions, processes the HTML content using BeautifulSoup to extract the main article text, and cleans the content by removing non-essential elements. I also used efficient techniques including concurrent requests using ThreadPoolExecutor (Moussa k. E., 2024) for faster processing, caching to avoid redundant downloads, and careful error handling to ensure robust extraction and being sticking to 5 pages per question.

After extraction, I had to got maximum Wikipedia content were pretty messy when you scrape it. I have got all sorts of weird formatting, citation numbers scattered everywhere, and inconsistent spacing that makes it hard to work with. So I cleaned it, some contents were already processed, some were still in raw list format and normalizes everything. Stripping out these [1], [2] citation markers, escape characters that don't belong, and fixes up the spacing so everything looks consistent.

Now the next thing that we can't just throw entire cleaned Wikipedia articles at your embedding model and expect good results. I mean, technically we could try to convert a whole article into one giant vector, but a single vector will have too much of information that would end up taking a lot of computational power. Plus, when we compress an entire article ,which might cover dozens of different topics into a single vector, we lose so much of the meaningful context that makes the information useful in the first place. It's like trying to summarize a whole book in one sentence - we can do it, but we're going to miss all the important details.(Ruben Winastwan, 2024a)

So for doing doing breaking down each Wikipedia content int certain chunks so that it should convey some information. After some going through different kinds of chunking strategy, I settled on chunking by words rather than characters. Why? Because it just makes more sense from a language perspective. Each chunk ends up being around 300 words rather than 512, which I found reasonable enough that had good context which is to be meaningful, but not so much that important details get lost. This created a clean, organized knowledge base that preserves the important medical information while making it easier to search through and retrieve relevant content when needed.

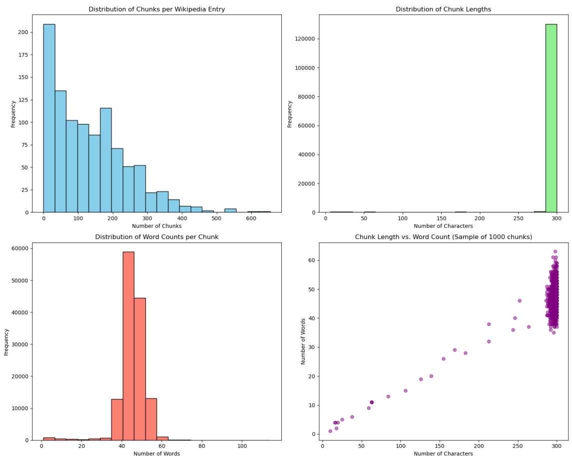

Figure 4: Chunk analysis

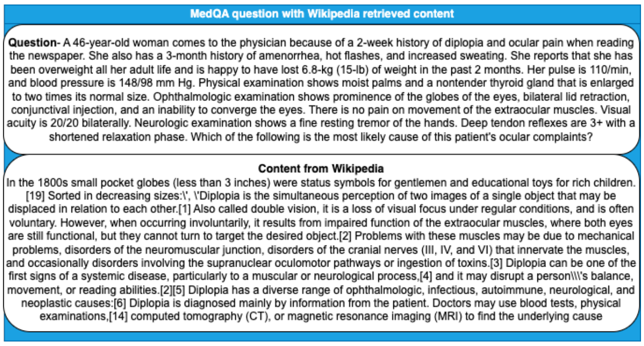

Figure 5: One chunk of the example of the content retrieved for the particular question

3.3 Web Chunking to Vector Embeddings and storing

In the previous section 3.2, where i converted the cleaned content into chunks now the next task is to transform each chunk into a vector embedding. This approach allows us to extract the most relevant chunks as contexts for our language model in QA. Usually there are two types of vector embeddings that are applied to real world problems (Ruben Winastwan, 2024b):

- The first type I use is called **"dense embeddings."** These come from advanced

AI models like those from OpenAI or tools like Sentence Transformers. I like to think of dense embeddings as a compact summary - they pack the meaning of text into a relatively small list of numbers, and most of those numbers actually contain useful information (they're not zero). It's like creating a detailed but concise profile of what the text means.

- The second type is **"sparse embeddings,"** which I get from older, more traditional methods like TF-IDF or BM25. These work more like a giant checklist - they create a much longer list of numbers, but most of those numbers are zero because they represent words that don't appear in that particular text. Only a few positions have actual values, which is why we call them "sparse" - they're mostly empty space with just a few meaningful numbers scattered throughout.

What I used is a simple embeddings model from sentence transformer (Huggingface, 2024) "all-MiniLM-L6-v2"—to turn each chunk into a set of 384 numbers. These numbers are like a code that captures what the text means. So, if two chunks are similar in meaning, their number sets will be close to each other. This lets me find related content even if the exact words don't match.

So now when I have those huge embeddings or these vectors where to save them? Well If the embeddings are not so long if less than 200K then we can store it in pytorch defined library for storing embeddings (PyTorch, 2024) but for my case it was pretty huge so I utilized FAISS (Douze et al., 2024) known for fast similarity search which helps to find a way to quickly those ones that were similar to each other. I also used techniques called IVF and Product Quantization described in this repo (Facebook AI Research, 2024). These help it handle large amounts of data by organizing the codes into clusters and shrinking them down, so it doesn't take up too much memory and still works super fast.

At last, the system saves three key pieces: the compressed vector index for fast searching, the original text chunks so I can read them, and the mapping that connects everything back to the source questions and answers.

3.4 Retrieval Integration

The retrieval system bridges the gap vector embeddings and language model, means defining the model that will generate the response to my query. As discussed 3.3 I have stored all the embeddings now it is time for next task which is implementing the retrieval part. My implementation begins by loading the previously constructed FAISS index and its associated metadata—the text chunks and their mappings to original questions. This allowed me for efficient storage and retrieval, with the vector index handling similarity computations while the corpus provides the textual content and contextual information needed for answer generation.

One of the biggest challenges I encountered was memory management, particularly when trying to deploy language models. Through several readings and errors, I find out several optimization techniques that proved essential. The most significant finding in my memory optimization came from implementing 4-bit quantization for the language model. Using the BitsAndBytesConfig (G, 2024) from the Transformers library, I configured normalized float 4 quantization with double quantization enabled. Since most of time the University server was over loaded and it always throw CUDA out-of-memory and this error lead me to amazing finding of using language models not only on GPU instead of CPU.

When we search for information, we want to make sure our system understands the question in the same way it understood the data it's searching through. So, I use the same embedding model—to process both the question and the data. The search operation retrieves both the indices of similar vectors and their corresponding distances, which are then used to rank the results. These indices are subsequently mapped back to their original text chunks and associated metadata, creating a detailed result set that encompasses both the relevant content and its source information.

So the next, retrieved chunks into a structured context string, which is then provided to the language model. To prevent exceeding the context window capacity. The method preserves as much relevant information as possible while ensuring the system remains stable across different query types.

The next process is of generating answers, where my system takes the retrieved context, the question, its category, and any available options, and applies an advanced prompting strategy to produce the output. Prompt templates are stored in external files, allowing for easy experimentation with different prompting methods. The system supports both standard prompting and chain-of-thought reasoning, with adjustable generation parameters to manage the balance between consistent, predictable outputs and more creative, exploratory responses.

4 Evaluation

To evaluate the performance of my model, I processes each question in the dataset (that is 1000 question in total) through the Retriever system and compares the generated answers against the correct options.

I conducted experiments with two distinct prompting strategies to assess their impact on model performance **Simple Prompt:** A straightforward approach that presents the question and options directly to the model, requesting an answer choice.Fig.6 **Multi-hop Reasoning Prompt:** A more sophisticated approach that guides the model through a structured reasoning process.Fig.7

Doctor Assistant - Simple Prompt

You are a doctor assistant that answers multiple-choice questions based on the provided context.
Context: {context}
Question: {question} **Category:** {category} **Options:** {options}
Give me the correct answer based on question and category and choose from the option which is correct.

Figure 6: Simple prompt

The multi-hop reasoning prompt proved significantly more effective, and the answer produced were quite reasonable but in the Fig.8 and my results it's quite opposite the accuracy should improved and the reason which I have found out will discuss in conclusion section.

The overall accuracy is given by:

$$\text{Overall accuracy} = \frac{\text{correct answers}}{\text{total questions}}$$

Medical Expert Assistant - Prompt

You are a medical doctor expert assistant tasked with solving multiple-choice questions by applying clinical reasoning and evidence-based analysis.
Context: {context}
Question: {question} **Category:** {category} **Options:** {options}
Follow this structured approach:
1. Analyze the question to identify key medical concepts, conditions, or mechanisms being tested
2. Extract and synthesize relevant evidence from the provided context
3. Apply clinical reasoning to connect the evidence to the question 4. Evaluate each option systematically: - For each option, give good reason from the context that supports or contradicts it - Assign a confidence score (0-10) with specific justification for each option - Explain the medical reasoning behind why an option is correct or incorrect 5. Conclude with your final answer, clearly stating the letter (A, B, C, or D) of the correct option
Remember that you are doctor assistance and you know the correct answer but you need to think clearly step-by-step reasoning throughout your analysis.

Figure 7: multi-hop prompt

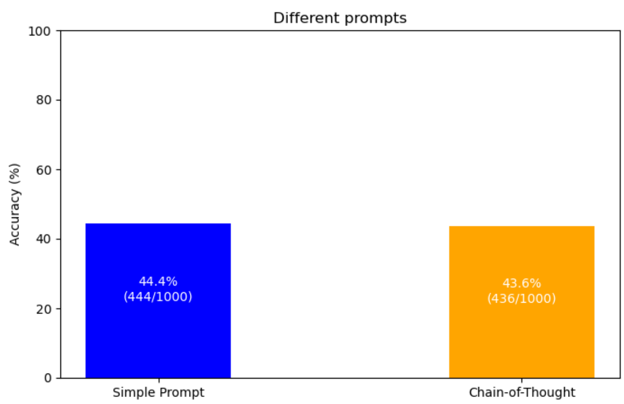

Figure 8: Comparison

5 Results and Discussions

In this section, we report, analyze, and discuss the performance of my model which demonstrated promising results on the MedQA-USMLE dataset, achieving an accuracy of 44.4% with simple prompt and 43.8% with multi-hop prompting when compared to the baseline small original model's where accuracy was 34.1% (Jin et al., 2020) and accuracy of 50% (Chen et al., 2025). This represents a substantial approx 10 percentage improvement, point on the baseline approach validating my hypothesis that retrieval-augmented generation can significantly enhance the performance of language models on complex medical reasoning tasks. Table1

While my approach doesn't match the performance of large-scale models like Med-PaLM 2 (86.5%) or GPT-4 (90.2%) (Chen et al., 2025), it offers a compelling trade-off between accuracy and computational efficiency. And also seen quite improvement on when applied with better prompt(N., 2024) and (Roy, 2024) 20.10% and 32.1%. These were the exact task that they implement but my model received quite good accuracy because of the techniques I used, However, what makes my CHARM architecture particularly compelling is that despite using a much smaller base model (Phi-3-mini with 4B parameters), I achieved competitive accuracies through strategic architectural innovations. The combination of domain-aware categorization using BioMistral-7B, optimized FAISS vector retrieval with quantization, and structured chain-of-thought prompting enabled my smaller model to deliver strong performance.

5.1 Model Limitation

As discussed in the above section, my model somehow performed well, but it have lot of limitations that need to seen, and that i have noticed while making it and evaluating it. These issues not only affected the performance of my system but also got to learn more about the functioning

- **Issue with Question categorizing:** One of the first major problems I noticed was with how my system categorized medical questions into different domains. Even though I used BioMistral-7B to improve the relevance of information retrieval, I found through manual checking that the system frequently got the classification wrong. This was particularly problematic when dealing with questions that covered multiple medical specialties or involved rare conditions. What made this worse was that these classification mistakes had a direct effect throughout the entire system. When the domain was wrong, the system would retrieve irrelevant Wikipedia content, which then led to incorrect final answers. I realized that treating domain classification as a simple, one-shot decision was a fundamental flaw in my approach. Looking back, I think a better solution would be to go with the approach of reasoning to categorize the question based on the knowledge graphs and the strategy discussed over here (Chen et al., 2025).

- **Unwanted data retrieved from Wikipedia:** Another major limitation I discovered was in how the system retrieved relevant medical knowledge. I relied on metamap_phrases for this process, but when I manually examined what content was being retrieved, I was often surprised by how irrelevant it was. In some cases, the retrieved information was not just unhelpful but actively misleading. For example where a question about urine problem somehow retrieved information about a carpenter's work injury, and another about diabetes management returned content about solar temperatures. These kinds of errors made it clear that my approach was too simplistic - it was essentially doing keyword matching without really understanding the medical concepts involved. The root of this problem, I believe, is that the current system lacks proper medical entity recognition. It treats all extracted phrases as equally important, which means that when there are non-essential terms in a question, they can dilute the relevance of what gets retrieved. If I were to continue this work, I would definitely implement specialized medical named entity recognition and find ways to weight core medical concepts more heavily than peripheral terms. The keyword extracted from question, was also not perfect it also need to be modified for

Model	MedQA Accuracy (%)				
	Simple Prompt	Multi-hop MKI	Hop Type Used		
Phi-3-mini	44.4	43.8	One-hop ✓	Two-hop ✗	Three-hop ✗

Table 1: Accuracy of Phi-3-mini on MedQA using a simple prompt and multi-hop prompt.

the further use.

- **Choosing wrong embedding models:** I choose for this because it was very pretty common one and everyone used it and I couldn't find any relevant one for the bio embeddings. As discussed this model, which lacks specialized training on medical terminology and concepts. This creates a fundamental mismatch between the semantic space of your embeddings and the specialized language of medicine. Medical terms with similar meanings but different lexical forms (e.g., "myocardial infarction" and "heart attack") may not be properly clustered in the embedding space, while terms that appear lexically similar but have distinct medical meanings may be incorrectly considered related

- **Chunking with Overlapping:** (Ganesh, 2024) Another significant limitation I discovered was with how I handled text chunking for the retrieved medical content. I used a simple approach where I split text into chunks of approximately 300 words each, without really thinking about whether this made sense from a medical perspective. Medical information has a natural structure - diseases are typically described along with their symptoms, which are then followed by treatment options. But my fixed word-count approach would often break up these logical connections, sometimes splitting a disease description from its symptoms, or separating treatment information from the condition it was meant to address. I could have tried with Chunking with Overlapping that gives a promising results for these kind of scenarios . Medical texts have this hierarchical structure where everything is interconnected, and by breaking that structure arbitrarily, I was forcing the smaller model to try to piece together relationships across different chunks - something it wasn't very good at, even with the chain-of-thought prompting I implemented.

- **Limitations in the Reasoning Process:** Even though the chain-of-thought prompting approach worked better than simple prompting, I still found significant limitations in how the system handled complex medical reasoning. As I passed the same prompt but the way of thinking of each question was very different ,this was clearly problematic because different types of medical questions need different reasoning approaches. Questions about what causes a disease need causal reasoning, questions about diagnosis need differential reasoning, and questions about treatment need yet another approach. But my system treated them all the same way. I also noticed that the model would sometimes create reasoning chains that sounded doable but were actually medically incorrect. Without any way to fact-check or verify information during the reasoning process, these errors would carry through to the final answer. This was particularly concerning because in medicine, getting the facts right is absolutely critical.

- **Answer Generation Problems:** Perhaps the most concerning issue I encountered was the model's tendency to generate answers that weren't even among the provided options. This happened frequently with simpler prompts - the model would confidently select "Option E" or "Option F" when the question only had options A through D, or sometimes it would create entirely new answer choices. This behavior really highlighted a fundamental problem with the model's ability to follow instructions and stay within the constraints of the task. Even when I explicitly told it to choose from the provided options, it would often ignore this instruction, which suggests some serious limitations in how well smaller models can adhere to given constraints. What made this even more problematic was that when using simple prompts, the model would often just guess randomly and that lead to my simple model to reach my model best accuracy upto 44.4% while other prompt struggled to get accuracy but it's answer given was quite relevant and understandable.

6 Future Improvements

- **Better Medical Keywords:** As discussed the biggest drawback for my model is this part, so for future i would like to make new keywords extracted columns that would contains only medical extracted keywords.

- **Different Prompts for different types of question:** as this dataset contains different reasoning type of types and what I was doing passing the same prompt for each and every question. Like a question about diagnosing a condition needs differential reasoning, while a question about treatment options needs therapeutic decision-making guidance.

- **Comparing the model without any Wikipedia knowledge:** So in my approach I just used different prompt comparison, so for better checking of my model, I would have compared my retrieval without any Wikipedia knowledge and then seeing the accuracy how many questions were correct, that way I could find put if my retrieval is working or not.

- **Multi-stage Retrieval:** Implementing an iterative retrieval process where initial reasoning identifies knowledge gaps that trigger additional targeted retrieval could address the limitations of single-pass retrieval.

7 Conclusion

In this work, I presented BioWiki, a novel architecture that enables medium-sized language models to perform complex medical reasoning through Wikipedia-based knowledge retrieval and structured prompting. By combining domain categorization, efficient vector search, and chain-of-thought reasoning prompt, my model achieves relevant performance on MedQA questions.

The results demonstrate that medium-sized language models can effectively tackle challenging medical reasoning tasks when provided with relevant external knowledge and appropriate reasoning guidance. This approach offers a practical alternative to massive language models in resource-constrained environments, making advanced medical question answering more accessible.

Despite promising results, significant limitations remain in areas like knowledge retrieval precision, reasoning flexibility, and handling of specialized medical concepts. These limitations highlight the challenges of medical reasoning and point to important directions for future work.

The BioWiki architecture represents a step toward more efficient and accessible AI systems for medical education and decision support. By continuing to refine this approach, I can develop systems that combine the reasoning capabilities of large language models with the practical advantages of smaller, more efficient implementations.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. https://www.microsoft.com/en-us/research/publication/phi-3-technical-report.

Asma Ben Abacha, Eugene Agichtein, Yuval Pinter, and Dina Demner-Fushman. 2017. Overview of the medical question answering task at trec 2017 liveqa. In *TREC 2017*.

Elliot Bolton, Abhinav Venigalla, Michihiro Yasunaga, David Hall, Betty Xiong, Tony Lee, Roxana Daneshjou, Jonathan Frankle, Percy Liang, Michael Carbin, and Christopher D. Manning. 2024. Biomedlm: A 2.7b parameter language model trained on biomedical text. *Unpublished Manuscript*. Preprint.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Jing Chen, Zhihua Wei, Wen Shen, and Rui Shang. 2025. Infusing multi-hop medical knowledge into smaller language models for biomedical question answering.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Facebook AI Research. 2024. Faiss indexes. https://github.com/facebookresearch/faiss/wiki/Faiss-indexes.

Anitha G. 2024. Bitsandbytesconfig: Simplifying quantization for efficient large language models. https://medium.com/@anitha6g/.

Jagadeesan Ganesh. 2024. Understanding chunking algorithms and overlapping techniques in natural language processing. https://medium.com/@jagadeesan.ganesh/.

Huggingface. 2024. sentence transformer sentence similarity. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? a large-scale open domain question answering dataset from medical exams.

Yanis Labrak, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. 2024. Biomistral: A collection of open-source pretrained large language models for medical domains.

Moussa k. E. 2024. Building a rag system with wikipedia. https://moussakam.github.io/demo/2024/09/26/arabic-rag.html.

Snehanshu N. 2024. Medprompt: Prompting small language models for medical qa. https://github.com/sn2727/medprompt-small-llms/blob/master/medprompt.ipynb.

PyTorch. 2024. torch.nn.embedding — pytorch documentation. https://docs.pytorch.org/docs/stable/generated/torch.nn.Embedding.html.

Ranadeep Roy. 2024. Llm with rag using medqa, pubmed, and wikipedia: Llama3_rag_context_summary_faiss notebook. https://github.com/royranadeep/.

Ruben Winastwan. 2024a. A beginner's guide to website chunking and embedding for your rag applications. https://zilliz.com/learn/.

Ruben Winastwan. 2024b. A beginner's guide to website chunking and embedding for your rag applications. https://zilliz.com/learn/.

Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, Mike Schaekermann, Amy Wang, Mohamed Amin, Sami Lachgar, Philip Mansfield, Sushant Prakash, Bradley Green, Ewa Dominowska, Blaise Aguera y Arcas, Nenad Tomasev, Yun Liu, Renee Wong, Christopher Semturs, S. Sara Mahdavi, Joelle Barral, Dale Webster, Greg S. Corrado, Yossi Matias, Shekoofeh Azizi, Alan Karthikesalingam, and Vivek Natarajan. 2023. Towards expert-level medical question answering with large language models.

STATWORX. 2022. Zero-shot text classification. https://www.statworx.com/en/content-hub/blog/zero-shot-text-classification.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries.

USMLE. 2021. Usmle catgeory outline. https://www.usmle.org/sites/default/files/2021-08/USMLE_Content_Outline.pdf. Accessed: 2025-05-20.

Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D. Manning, Percy Liang, and Jure Leskovec. 2022a. Deep bidirectional language-knowledge graph pretraining. In *Neural Information Processing Systems (NeurIPS)*.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2022b. Linkbert: Pretraining language models with document links. In *Association for Computational Linguistics (ACL)*.

Result with multi-hop reasoning prompt

Question: A 23-year-old pregnant woman at 22 weeks gestation presents with burning upon urination. She states it started 1 day ago and has been worsening despite drinking more water and taking cranberry extract. She otherwise feels well and is followed by a doctor for her pregnancy. Her temperature is 97.7°F (36.5°C), blood pressure is 122/77 mmHg, pulse is 80/min, respirations are 19/min, and oxygen saturation is 98
Category: Renal/Urinary
Options: A: Ampicillin, B: Ceftriaxone, C: Doxycycline, D: Nitrofurantoin
Retrieved Context: Here is some relevant information:
[1] due to interference with fetal testosterone metabolism, and as a precaution, pregnant women should not handle crushed or broken tablets. A 2018 Cochrane review of studies on men over 60 with moderate to severe lower urinary tract symptoms analyzed the impacts of phosphodiesterase inhibitors (PDE) in
[2] high-quality medical care, though babies born at this time will likely experience serious health complications such as heart and respiratory problems and long-term intellectual and developmental disabilities. Prenatal care improves pregnancy outcomes. Nutrition during pregnancy is important to
[3] ensure healthy growth of the fetus. Prenatal care also include avoiding recreational drugs (including tobacco and alcohol), taking regular exercise, having blood tests, and regular physical examinations. Complications of pregnancy may include disorders of high blood pressure, gestational diabetes,
[4] daily living or pose a health-threat to the mother or baby. Complications during pregnancy can cause other more severe symptoms, such as those associated with anemia. Early signs and symptoms of pregnancy may include: Other signs and symptoms that some people may experience at different stages of

[5] pregnant before. It is common for some women not to feel the fetus move until much later. During the second trimester, when the body size changes, maternity clothes may be worn. During pregnancy, a woman undergoes many normal physiological changes, including behavioral, cardiovascular, hematologic,

---

Generated Answer: Step 1: Analyzing the Question The patient, a pregnant woman in her second trimester, presents with burning upon urination that has been worsening. She is generally healthy, maintains a regular prenatal check-up, and does not show signs of costovertebral angle tenderness. The question tests the understanding of the most appropriate initial treatment for a urinary tract infection (UTI) in pregnant women.

Step 2: Extracting and Synthesizing Relevant Evidence The patient's symptoms are consistent with a urinary tract infection (UTI). As a pregnant woman, she should avoid certain medications (e.g., doxycycline) due to potential harm to the fetus. Additionally, good prenatal care can improve pregnancy outcomes, indicating that evidence-based treatment is crucial.

Step 3: Applying Clinical Reasoning For pregnant women with a UTI, the first-line treatment is usually a safe and effective antibiotic that targets the bacteria causing the infection. Nitrofurantoin is often recommended for uncomplicated cystitis in pregnant women, as it has a low risk of causing side effects for both the mother and fetus. However, it should be used with caution as it's contraindicated in late pregnancy due to the risk of developing hemolytic anemia.

Step 4: Evaluating Each Option Systematically A. Ampicillin: While it's a common antibiotic used to treat UTIs, its safety during pregnancy is less well established than nitrofurantoin.
B. Ceftriaxone: It's a third-generation cephalosporin and can be used to treat UTIs, but is usually reserved for more complicated cases or when other treatments fail or are contraindicated.
C. Doxycycline: This antibiotic is contraindicated in pregnancy as it may cause permanent discoloration of teeth and inhibit bone growth in the fetus.
D. Nitrofurantoin: This antibiotic is considered safe and effective for the treatment of uncomplicated UTIs during pregnancy. However, it should be used with caution in late pregnancy (after 36 weeks).
Confidence Scores: A. Ampicillin: 6/10 - It's an option but less preferable due to less clear evidence for safety in pregnancy. B. Ceftriaxone: 5/10 - Safe in pregnancy, but usually not first-line for uncomplicated UTIs. C. Doxycycline: 0/10 - Highly contraindicated in pregnancy. D. Nitrofurantoin: 9/10 - First-line treatment for uncomplicated UTIs in pregnancy, safe when used with caution in later pregnancy.

Step 5: Conclusion The best option for this patient is D. Nitrofurantoin, as it is the preferred first-line treatment for uncomplicated UTIs in pregnant women. However, the patient's care provider should be aware of its use in late pregnancy.

Result with standard reasoning prompt

Question: A 23-year-old pregnant woman at 22 weeks gestation presents with burning upon urination. She states it started 1 day ago and has been worsening despite drinking more water and taking cranberry extract. She otherwise feels well and is followed by a doctor for her pregnancy. Her temperature is 97.7°F (36.5°C), blood pressure is 122/77 mmHg, pulse is 80/min, respirations are 19/min, and oxygen saturation is 98
Category: Renal/Urinary
Options: A: Ampicillin, B: Ceftriaxone, C: Doxycycline, D: Nitrofurantoin

---

Generated Answer: 1. Key medical concepts/conditions: - Pregnant woman with a urinary complaint (burning upon urination) - No systemic symptoms (well feeling, normal vital signs) - No costovertebral angle tenderness

2. Evidence: - Presenting symptom: Dysuria without infection (no fever, no costovertebral angle tenderness) - Current treatment: Increased water intake, cranberry extract

3. Clinical reasoning: - The most likely cause of burning urination in a pregnant woman, absent systemic symptoms, is a lower urinary tract infection (UTI), likely asymptomatic bacteriuria or a bladder infection (cystitis).

4. Evaluation of each option: - A. Ampicillin: This would be appropriate for a symptomatic UTI. However, the context suggests no systemic symptoms (normal vital signs), which might imply an asymptomatic bacteriuria. Antibiotics in pregnancy should be used judiciously to avoid resistance.
Confidence: 4/10 - Not the first line of treatment without evidence of systemic infection. - B. Ceftriaxone: Given the lack of systemic symptoms, this broad-spectrum antibiotic might be excessive. However, as the patient is pregnant, a cautious approach is warranted.
Confidence: 3/10 - Too broad-spectrum for uncomplicated UTI in pregnancy. - C. Doxycycline: This antibiotic is contraindicated in pregnancy as it can affect fetal bone and tooth development.
Confidence: 0/10 - Contraindicated in pregnancy. - D. Nitrofurantoin: This is often used as first-line therapy for uncomplicated UTIs in pregnancy, especially before 36 weeks gestation.
Confidence: 9/10 - Highly recommended for uncomplicated UTIs in pregnancy before the third trimester.

5. Conclusion: The best treatment in this scenario would be Nitrofurantoin (Option D), which is the preferred first-line antibiotic for uncomplicated UTIs in pregnant women before 36 weeks gestation. It has good efficacy against common urinary pathogens and a favorable safety profile in pregnancy when used appropriately.

Final Answer: D

Figure 9: 1st figure is an example of multi-hop reasoning step and 2nd figure is an example of simple prompt