UNITECH SELECTED PAPERS

International Scientific Conference
UNITECH`2024

EXPLORING RAG IN MEDICAL QUESTION ANSWERING: INTEGRATING LLMS AND VECTOR DATABASES

Matija Špeletić[1*], Stevica Cvetković[2], Milan Protić[3], Saša V. Nikolić[4]

[1,2,4] University of Niš, Faculty of Electronic Engineering, Niš, Serbia;
[3] Academy of Technical-Educational Vocational Studies, Niš, Serbia;

**Corresponding author: matija.speletic@elfak.ni.ac.rs*

Abstract

This paper presents the design, implementation, and evaluation of a Retrieval-Augmented Generation (RAG) system for medical question answering. The proposed system integrates a state-of-the-art large language model (LLM) with a vector database powered by Neo4j for fast and efficient information retrieval. To enhance the retrieval component, we employed Nomic's embedding model to generate high-quality vector representations of medical documents. The architecture leverages the synergy between retrieval and generation, enabling the LLM to generate context-aware responses based on relevant clinical trial data. A test dataset was created by extracting clinical trial reports from open-source documents and generating synthetic questions using a language model. Our experimental results demonstrate the potential of RAG-based systems in the medical domain, highlighting their ability to provide accurate and context-rich answers. This study presents both the strengths and limitations of the RAG approach for specialized domains such as healthcare.

Keywords: Retrieval-Augmented Generation (RAG), Large Language Model (LLM), vector database, clinical trial, information retrieval.

INTRODUCTION

In recent years, the release of ChatGPT by OpenAI has spurred rapid growth and popularity of generative models, particularly large language models (LLMs). These models are capable of understanding natural language inputs, identifying complex relationships within them, and generating coherent, context-aware responses. The underlying transformer architecture [1] was introduced in 2017, and today, models such as GPT-3.5 [2] and GPT-4 [3] power some of the most advanced systems.

One of the primary applications of LLMs is Question Answering (QA), where the model not only retrieves answers but demonstrates deep comprehension and the ability to link information meaningfully. However, LLMs face two key limitations: (1) they are restricted to the knowledge available at the time of training, and (2) they may lack the specific domain knowledge required for specialized applications, such as medical protocols or proprietary data.

A straightforward strategy for specialized QA would involve training a new LLM from scratch using domain-specific data. While effective, this process demands enormous computational resources, making it impractical and unscalable. Another approach is fine-tuning an existing model on a smaller, relevant dataset. Though less resource-intensive, fine-tuning still struggles with scalability and maintaining up-to-date information, requiring retraining whenever new knowledge needs to be integrated.

To address these challenges, Retrieval-Augmented Generation (RAG) has emerged as an effective alternative. RAG combines the power of LLMs with external retrieval mechanisms, such as vector databases. In this setup, relevant information is fetched in real-time from an external knowledge base and provided as input to the LLM, which then generates the response. This architecture eliminates the need for continuous retraining and ensures scalability and real-time access to updated information,

doi: www.doi.org/10.70456/.....

making it highly suitable for dynamic environments like medical QA systems.

This paper focuses on the use of RAG architectures for building advanced QA systems, with a particular emphasis on medical applications.

RELATED WORK

RAG aims to improve information fidelity in complex question-answering and knowledge-intensive tasks. Lewis et al. introduced the RAG architecture, demonstrating how dense passage retrieval (DPR) provides factual context to generative models [4]. RAG has since been applied across domains where current information is critical. Izacard and Grave [5] improved RAG with fusion-in-decoder (FiD) mechanisms, yielding better performance by integrating multiple retrieved documents. RAG's applications in specialized fields, including healthcare and legal domains, show promise for domain-specific, contextually relevant responses [6].

In the medical domain, RAG has emerged as a promising method for delivering accurate, context-sensitive answers to complex clinical and biomedical queries. By integrating retrieval mechanisms with large language models, RAG can incorporate up-to-date information from reliable sources, crucial in the rapidly evolving field of medicine. Early work by Lee et al. [7] introduced dense retrieval techniques optimized for biomedical literature, enhancing the accuracy of QA systems. Hu et al. [8] further refined RAG by incorporating structured medical ontologies, which improved the relevance and precision of generated answers. These advancements highlight RAG's potential to enhance medical AI, enabling models to provide factually accurate responses grounded in current clinical evidence.

METHOD

RAG architecture enhances the quality of answers generated by LLMs by integrating domain-specific information, ensuring accurate responses. This architecture offers several advantages:

- Access to up-to-date, reliable information by integrating external knowledge bases.
- Transparency and verifiability by providing references for the data used in responses.
- Reduced hallucinations by minimizing the reliance on latent information within the model's parameters.
- Ease of updates, as adding or modifying domain knowledge does not require model retraining.

Architecture

RAG consists of two primary phases: indexing and retrieval with response generation (**Fig. 1**).

Indexing occurs offline, before system use. It involves uploading relevant documents into a vector database by splitting them into smaller segments (chunks) to fit within the LLM's input limits. Each chunk is transformed into an embedding vector (a numerical representation of semantic content) using models such as BERT [9] or other transformer-based encoders. These vectors are stored alongside the text in the vector database to support semantic search (finding the most relevant chunks based on embedding vector similarity).

Chunking documents is essential for building a usable knowledge base. Documents, which can span thousands of pages, are divided into smaller, overlapping sections. Overlaps help maintain semantic coherence between sections, preventing key information from being split across chunks. For structured formats (e.g., HTML, DOCX), chunking is straightforward, but unstructured formats like PDFs may require more complex processing. Saving metadata (e.g., document name, page number) with each chunk improves traceability and simplifies database updates.

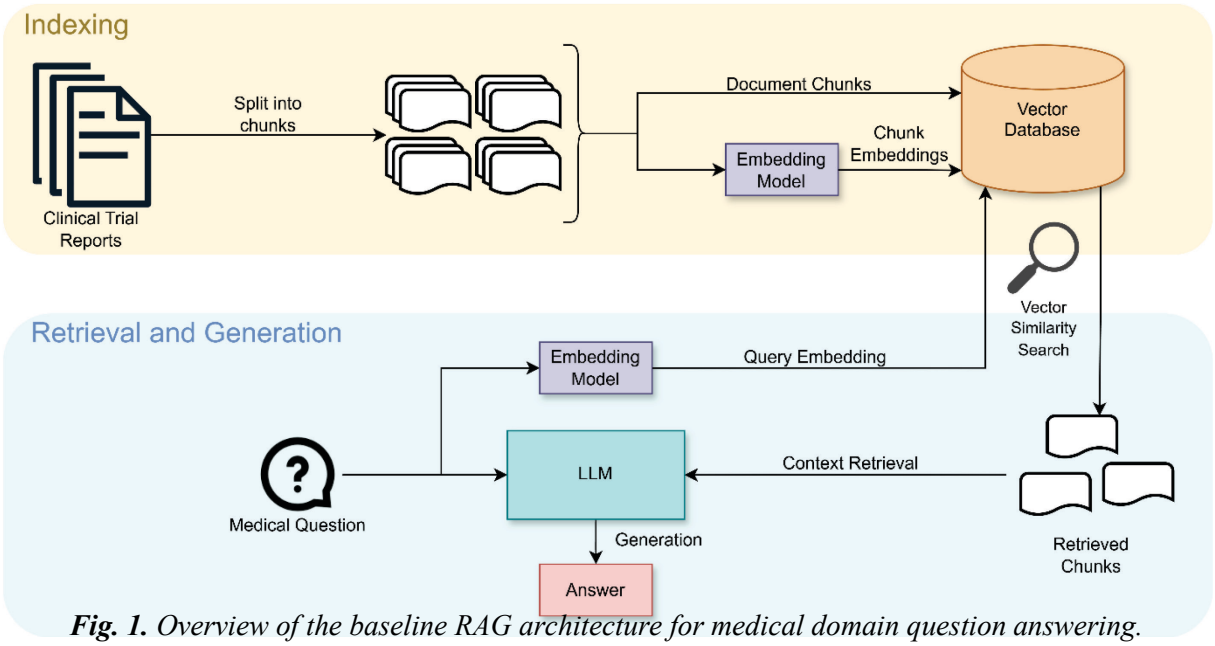

Fig. 1. *Overview of the baseline RAG architecture for medical domain question answering.*

Embedding models convert both text and user queries into vectors, capturing their semantic meaning. Unlike traditional word-matching techniques (e.g., TF-IDF), embeddings can handle synonyms and different phrasings effectively. Modern embedding models, such as BERT, use attention mechanisms to weigh the importance of words in context. This research employs the Nomic [10] model, which follows the BERT framework, to generate embeddings for both chunks and user queries.

During the retrieval and response generation phase, which happens online, the system processes user queries in real-time. When a query is received, the retrieval component calculates its embedding vector and searches the vector database for the top K most relevant text chunks. These retrieved chunks, along with the user query, are fed to the LLM, which generates a final response.

To enable rapid and efficient lookups, even when working with large knowledge bases, RAG systems usually employ vector databases. These databases are used to find the most similar chunks of data based on metrics such as cosine similarity or Euclidean distance between embedding vectors. In this paper, we utilize the Neo4j database, which supports vector search indexes [11] and uses the HNSW (Hierarchical Navigable Small World) [12] algorithm for fast, approximate KNN (K-nearest neighbors) searches.

This methodology ensures that RAG-based QA systems are scalable, accurate, and easy to maintain, making them ideal for applications requiring real-time access to specialized information, such as in medical or legal domains.

Implementation

The organization of the system's classes is represented by the UML class diagram shown in **Fig. 2**. The *VectorStore* class serves as an abstract class that provides an interface for working with vector databases. The purpose of this abstraction is to allow for easy integration of various vector databases (besides Neo4j) to support future extensions. The *Neo4jVectorStore* class inherits from this abstract class, implementing the core vector storage functionality with the Neo4j database.

The *Embedding* and *LLM* classes offer the respective interfaces for working with embedding models and large language models. The retrieval component of the RAG architecture, which searches for the

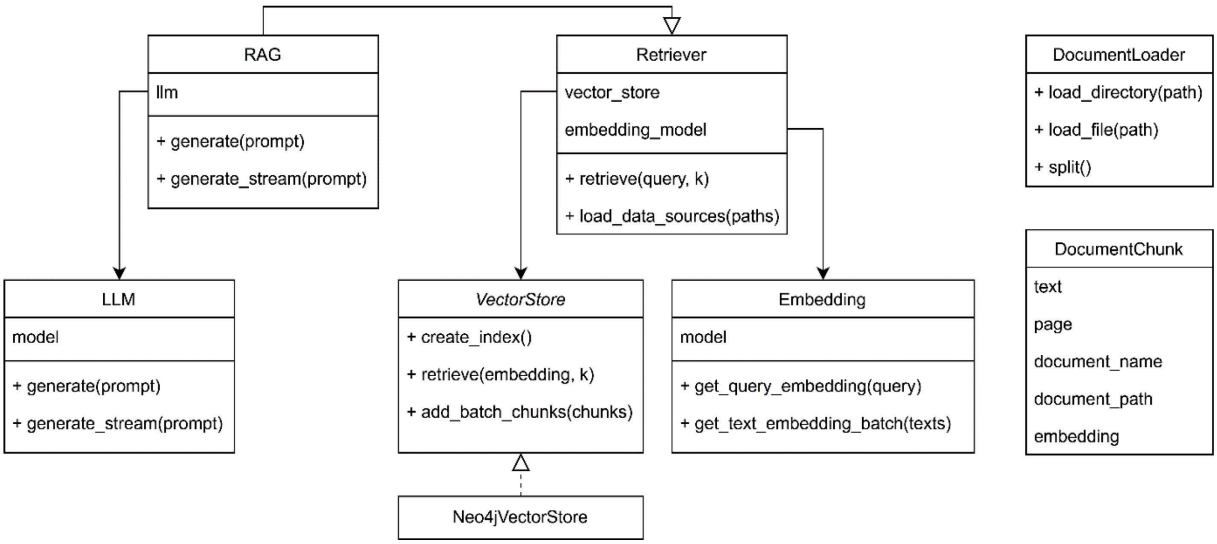

Fig. 2. *Class Diagram of the Implemented QA System Based on RAG Architecture*

most relevant text chunks based on a user query, is implemented within the *Retriever* class. This class is further extended by the *RAG* class, which provides the final interface for interacting with the whole system (both the *LLM* and *RAG* classes expose identical interfaces).

Apart from the main system components, the diagram also includes utility classes such as:

- *DocumentLoader*: Responsible for loading documents and splitting them into manageable chunks.
- *DocumentChunk*: Models individual chunks of text that flow through the system.

This modular design ensures the flexibility and maintainability of the system, allowing for future enhancements, such as integrating new vector databases or trying out different LLMs and embedding models.

EVALUATION

The final step in developing the QA system involves testing and evaluating the implemented solution to verify how accurately it responds to queries over a specific dataset. It is essential to ensure that the knowledge base documents are not part of the LLM's pre-trained corpus. Testing with general knowledge questions or literary topics would likely produce biased results, as such information is often already known to the LLM. To get reliable results from the RAG architecture, the dataset used for testing should contain recent or highly specific information—for example, clinical trial reports or company documentation published after the LLM's training period.

The system will be tested using the GPT-3.5 LLM, using text chunks with a length of 1024 and an overlap of 128 characters.

The test dataset consists of NIH clinical trial protocols published after November 11, 2023, which were downloaded from [13]. The system will be evaluated using 8 documents and 22 synthetic questions related to the content (denoted as Q1-Q22). These questions aim to test the RAG system's ability to provide detailed answers requiring reasoning and multiple information sources, unlike traditional QA systems that typically offer short answers (e.g., "yes/no" or simple definitions).

To provide a quantitative evaluation, the RAGAS (Retrieval Augmented Generation Assessment) [14] method will be used. RAGAS provides several metrics to assess the system’s performance:

- *Answer Relevancy* (AR): Measures how relevant the generated answer is to the query by calculating the average cosine similarity between the original query and artificially generated questions based on the system’s answer.

- *Faithfulness* (F): Assesses whether the generated answer is factually correct based on the retrieved context. An answer is faithful if all statements can be inferred from the retrieved context.
- *Context Recall* (CR): Measures how well the retrieved context aligns with the ground truth answer, based on the overlap of related sentences.
- *Answer Semantic Similarity* (AS): Evaluates the cosine similarity between the generated answer and the correct answer to determine semantic closeness.

Question	AR	F	CR	AS
Q1	0.943	1	1	0.988
Q2	0.95	1	1	0.968
Q3	1	1	0.943	0.944
Q4	0.947	1	0.95	0.874
Q5	1	1	1	0.937
Q6	1	0.756	0.947	0.963
Q7	1	1	0.879	0.955
Q8	0.5	0.95	0.888	0.967
Q9	1	0.926	1	0.982
Q10	1	0.818	1	0.916
Q11	1	1	1	0.982
Q12	0.5	0.954	1	0.956
Q13	1	0.981	0.909	0.851
Q14	0.75	0.964	1	0.979
Q15	1	0.946	0.833	0.883
Q16	0.972	0.833	1	0.913
Q17	0.966	1	1	0.987
Q18	0.943	1	0.8	0.948
Q19	0.967	1	1	0.9
Q20	0.961	1	1	0.938
Q21	0.947	1	0.2	0.974
Q22	0.959	1	1	0.985
AVG	0.923	0.96	0.925	0.945

Table 1*. Results of the RAGAS evaluation on the test dataset*

The quantitative results based on the RAGAS metrics show that the system performs well across all evaluation criteria, with 1 being the best score and 0 the worst.

Upon inspecting the quantitative metrics, it is evident that the GPT-3.5-powered system produces concise and relevant answers with high scores across all RAGAS metrics. The system demonstrates strong performance in generating relevant and factually correct answers. The results confirm that it can effectively extract key information from the retrieved context and generate accurate, concise answers aligned with the user's query.

CONCLUSION

This work presents a baseline implementation of the RAG architecture within the medical domain, providing a scalable, up-to-date, and verifiable solution for building QA systems. The architecture leverages the power of LLMs combined with retrieval components, enabling the generation of answers based on the latest relevant information, even for complex medical questions.

In addition to implementing the system, this project also provides an overview of the system's performance on a relevant medial dataset. To evaluate the RAG architecture, the RAGAS evaluation framework was employed—a relatively recent method offering several metrics to assess both individual components and the overall architecture. Using the implemented system and the RAGAS framework, the performance was tested with GPT-3.5, showing that the system achieved excellent results in all key metrics.

The strong performance of the GPT-3.5-powered system highlights that high-quality QA systems can be achieved without the need for complex or resource-intensive infrastructure, expanding the possibilities for deploying these systems across various platforms.

The strong performance of the GPT-3.5-powered system highlights that high-quality QA systems can be achieved without the need for complex or resource-intensive infrastructure, expanding the possibilities for deploying these systems across various platforms. While QA systems have long been a topic in NLP, the integration of modern approaches such as LLMs and RAG architectures offers significant improvements to their functionality. In the medical domain, this advancement is particularly promising, as it enables more accurate and efficient retrieval of medical information, assisting healthcare professionals in clinical decision-making.

Funding: This work was supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia, grant number 451-03-65/2024-03/200102.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[2] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language Models are Few-Shot Learners." 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[3] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, *et al.*, "GPT-4 Technical Report." 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., Curran Associates, Inc., 2020, pp. 9459–9474. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf

[5] G. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, P. Merlo, J. Tiedemann, and R. Tsarfaty, Eds., Online: Association for Computational Linguistics, Apr. 2021, pp. 874–880. doi: 10.18653/v1/2021.eacl-main.74.

[6] V. Karpukhin, B. Oğuz, S. Min, L. Wu, S. Edunov, D. Chen, and W. Yih, "Dense Passage Retrieval for Open-Domain Question Answering." Apr. 2020. doi: 10.48550/arXiv.2004.04906.

[7] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, Sep. 2019, doi: 10.1093/bioinformatics/btz682.

[8] M. Hu, L. Zong, H. Wang, J. Zhou, J. Li, Y. Gao, K.-F. Wong, Y. Li, and I. King, "SeRTS: Self-Rewarding Tree Search for Biomedical Retrieval-Augmented Generation." 2024. [Online]. Available: https://arxiv.org/abs/2406.11258

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." 2019.

[10] Z. Nussbaum, J. X. Morris, B. Duderstadt, and A. Mulyar, "Nomic Embed: Training a Reproducible Long Context Text Embedder." 2024.

[11] "Vector indexes." [Online]. Available: https://neo4j.com/docs/cypher-manual/current/indexes/semantic-indexes/vector-indexes/

[12] Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 4, pp. 824–836, 2020, doi: 10.1109/TPAMI.2018.2889473.

[13] "Docugami Knowledge Graph Retrieval Augmented Generation (KG-RAG) Datasets: NIH Clinical Trial Protocols." Accessed: May 01, 2024. [Online]. Available: https://github.com/docugami/KG-RAG-datasets/tree/main/nih-clinical-trial-protocols

[14] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated Evaluation of Retrieval Augmented Generation." 2023.