Retrieval-Augmented Generation Integrated with Cloud-Based Document Archive

Adam Valík*

Abstract
Generative AI models struggle to answer domain-specific questions because they lack access to proprietary external data. This thesis presents a system that extends language models with an efficient retrieval mechanism using a vector database integrated with cloud-based archive. Documents are processed and retrieved using a combination of semantic and full-text search and reranking for context-aware generation. The resulting Retrieval-Augmented Generation (RAG) pipeline is designed to respond with accurate, relevant and trustworthy answers by grounding them in real organizational knowledge.

*xvalik05@stud.fit.vut.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Large language models (LLMs) remain limited by static training data and lack access to proprietary or up-to-date information. Training a custom LLM is prohibitively expensive, and sharing sensitive documents with public cloud models poses risks of data leakage. Retrieval-Augmented Generation (RAG) addresses these challenges by allowing LLMs to incorporate private external knowledge securely.

This work focuses on enabling efficient question answering over text-based documents stored in the cloud. The system processes diverse document formats, performs hybrid retrieval, and provides accurate, context-grounded responses. The approach integrates document processing, vector database synchronized with the cloud archive and retrieval mechanisms designed to maximize precision and minimize operational costs.

2. Vector Databases

Vector databases store data as high-dimensional vectors, enabling semantic search over unstructured information such as text, images and audio [1]. Unlike relational databases, which rely on exact matches across structured tables, vector databases retrieve data based on meaning and similarity. Modern applications like recommendation systems, chatbots and Retrieval-Augmented Generation (RAG) systems depend on vector databases for fast and accurate semantic retrieval.

3. Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a technique that enhances the capabilities of language models by retrieving relevant external information and incorporating it into the generation process. RAG systems dynamically fetch document fragments to ground the model's responses in up-to-date or proprietary knowledge.

The concept of RAG was introduced by Lewis et al. [2], who demonstrated that integrating retrieval into text generation significantly improves factual accuracy and reduces hallucination.

4. System Implementation

The proposed system follows a modular Retrieval-Augmented Generation (RAG) architecture. The implementation focuses on optimizing the retriever module to build a precise context for answering user queries, while minimizing costs and ensuring information security by sending only the most necessary data to the generator.

The system consists of three primary processes: cloud archive synchronization, document processing, and retrieval.

4.1 Cloud archive synchronization

Cloud archive synchronization allows automatic monitoring of a specified folder in Google Drive. Changes such as additions, modifications, or deletions of documents are detected through a webhook integration and synchronized with the vector database. New or modified documents are processed on the fly, ensuring that the knowledge base reflects the current state of the cloud archive.

4.2 Document processing

In the document processing phase, various file formats including TXT, DOC, DOCX, PDF, JPG, PNG and HEIC are ingested. Optical character recognition (OCR) is applied to extract text from non-textual files. The extracted text is then cleaned, partitioned into sections using a chunking method that preserves context while respecting the input token limit of the embedding model and enriched with metadata such as access rights and the position of the text passage within the file. Each chunk is subsequently embedded into a dense vector using the sentence transformer model `all-mpnet-base-v2` and stored in a Weaviate vector database optimized with HNSW indexing.

4.3 Retrieval

The retrieval phase begins when a user submits a query. The system first reformulates the query to improve retrieval quality, using a language model to simplify or disambiguate the input without changing its meaning. The query is then vectorized using the same embedding model as in document processing and passed to hybrid retrieval, which combines keyword-based BM25 search with vector-based Approximate Nearest Neighbor Search (ANNS). Retrieved candidates are reranked using a cross-encoder model to better match semantic relevance. The top-ranked chunks, labeled as relevant to the query and potentially holding the answer, serve as context for a prompt. The prompt is designed using prompt engineering principles to ensure that the answer is derived strictly from the provided context and that a fallback message is generated if relevant information is missing, preventing hallucinations. The prompt is then passed to an OpenAI language model, which generates a factual, source-cited response. The system also uses streaming responses to deliver generated answers to the user in real-time.

5. Evaluation

The system was experimentally evaluated on a subset of Wikipedia dataset to verify its overall effectiveness in document retrieval and answer generation as well as to fine-tune some system parameters such as hybrid search balancing and reranker filter threshold.

The full RAG pipeline was evaluated using the RAGAs framework. This end-to-end assessment employed an LLM-as-a-judge approach, automatically scoring the system across multiple dimensions, including context precision, context recall, response relevancy, faithfulness, and factual correctness.

The evaluation results showed that the retrieval module performs well in finding relevant information for direct user queries. The language model consistently generates responses that are relevant to the query and grounded in the retrieved context, effectively reducing the risk of hallucinations. Metrics related to retrieval and answer quality achieved values between 0.84 and 0.89.

The F1 score remained around 0.5, indicating possible limitations in factual correctness when compared to reference *golden answers*. This discrepancy may result from variations in wording, as the model can generate factually correct but differently phrased or more enriched responses. Moreover, since the answer format is not strictly defined, correct responses may differ in structure or detail from the reference answers. Thus, the F1 score should be interpreted cautiously, particularly for open-ended questions.

6. Conclusions

This work presents a modular Retrieval-Augmented Generation system designed for efficient question answering integrated with cloud-based document archive. The system successfully integrates document processing, hybrid retrieval, reranking, and prompt-based answer generation.

There is significant room for further development, including adding multimodal capabilities and multilingual support. The main future direction is to transition from a local prototype to a production-ready system that can be offered to real users, with options for fine-tuning based on specific organizational needs.

Acknowledgements

I would like to sincerely thank Ing. Filip Hadač and Mgr. Jiří Poclar, Ph.D., from Gauss Algorithmic for introducing me to this topic, providing the project assignment, and offering their valuable expertise and consultations throughout the development of this work.

References

[1] Yikun Han, Chunjiang Liu, and Pengfei Wang. A comprehensive survey on vector database: Storage and retrieval technique, challenge, 2023.

[2] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.