Enhancing Language Models with Retrieval-Augmented Generation for Accurate and Contextual Responses

MAURO MAZZEI
Istituto di Analisi dei Sistemi ed Informatica - National Research Council – LabGeoInf,
Via dei Taurini, 19,
00185 Rome,
ITALY

Abstract: - Large language models (LLMs), now also used in production environments, are susceptible to significant inaccuracies and errors, particularly when very specific topics in sectoral domains are involved. Incorrect responses produced by generative systems can cause many problems for non-expert users who, unfamiliar with the specific field of knowledge, are unable to assess the reliability of the responses generated. This problem further amplifies the errors of generative platforms. This paper explores how to mitigate such issues using Retrieval-Augmented Generation (RAG), a technique that enhances LLMs by integrating external information retrieval. RAG helps reduce hallucinations by grounding responses in relevant, retrieved content. The study examines the architecture and implementation of a RAG system and evaluates its effectiveness in improving response accuracy through simple experimental examples. It also investigates techniques and mathematical models to enhance the relevance of retrieved information and discusses the flow of structured and unstructured data into a vector database. This case study uses an open-source framework to demonstrate the design, implementation, and configuration of a RAG-based architecture using cloud infrastructure.

Key-Words: - RAG, Information Retrieval, Large Language Models (LLMs), Vector Embeddings, Cosine Similarity, Contextual Relevance, Multi-modal Embeddings, Retrieval Pipeline, Semantic Search, Long-Context Models.

Received: May 15, 2025. Revised: July 29, 2025. Accepted: September 16, 2025. Published: March 17, 2026.

1 Introduction

At this moment in history, large language models (LLMs) are used in all sectors where there is a need to quickly generate text and respond to questions of varying specificity. Properly trained models are capable of generating natural language text. Despite constant updates in the models LLMs have serious limitations especially when one wants to generate answers in very specific domains, in these cases models that have not been trained with respect to the specific domain generate wrong information with real hallucinations. This is a very significant problem especially for users who unknowingly do not verify their results. These problems greatly limit the use, reliability, and usefulness of LLMs in contexts where accuracy and timeliness of information are a very important issue.

To overcome these issues, the Retrieval-Augmented Generation (RAG) paradigm was developed. This type of architecture coupled with LLMs allows the integration of updated data from reliable knowledge bases. In this mode of integration, inaccuracies are reduced while improving the quality and reliability of responses.

To function, a RAG architecture needs two main components: the LLM generative model and information retrieval in a knowledge base that includes documents, databases, structured and unstructured data sources, [1], [2].

One of the most important aspects of RAG architectures is the ease of updating information, since updating does not require a new production of the LLM model but relies only on updating the knowledge base confined to the documentation that was required for information retrieval. These aspects are very important in contexts such as medicine finance and technology. We can envisage generating a RAG architecture linked to real-time news so that we always have the up-to-date and reliable answers, these are the most important aspects for this type of architecture.

A very important aspect is also related to cost reduction both in database management but also in energy aspects, since it is no longer necessary to update billion-parameter LLM models but only the part that one wants to integrate. Another relevant aspect is also the management and security of

information, which can be confined only to the RAG architecture part, [3], [4].

As Figure 1 shows, the operational flow of RAG architectures begins with the formulation of a query by the user. The query is processed by the Information Retrival system that goes on to weigh the contents of the knowledge base that has been integrated. All the information belongs to a vector database that uses the models in integration defined as Embedding Model, [5]. The vectors are contextualized are provided to the LLM model that uses both sui training data and the specific context that has been embedded to get an accurate answer.

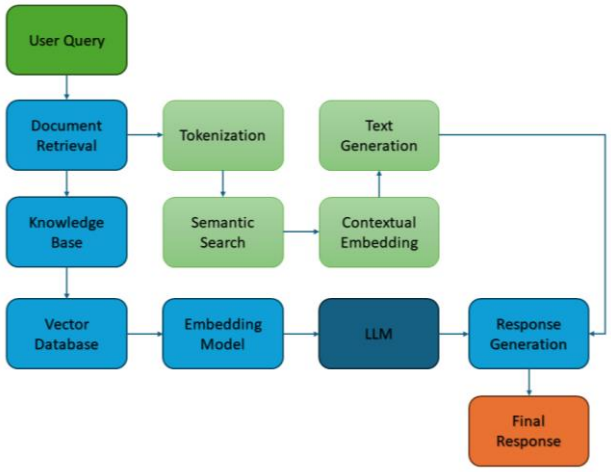

Fig. 1: Flowchart of main actions in RAG architectures

2 Related Work

The advent of LLM models and their weaknesses are to be considered fertile ground for researchers who have proposed various solutions to improve the reliability of these models, including the approach of RAG architectures, [6], [7].

A major limitation of LLM models is the dependence of training data; these systems make themselves very expensive in both management and energy aspects to produce more and more trained models. These are the motivations that have led researchers to find other more reliable solutions, [8], [9].

In many areas, especially where topics change rapidly such as finance, medicine, and technology, RAG architectures have brought substantial improvement, [10]. In particular, some studies such as the architecture of a convolutional neural network (CNN) with enhanced residual blocks (Res), and linear unit collectively called CNN-RAGNet architecture shows good results, [11].

Ethical and privacy issues are also a domain of work being addressed in RAG architectures, of evaluating ethical and privacy considerations for legal RAG by delving into usage benchmarks, [12].

Compared with traditional methods, with LLM, RAG architectures demonstrate a 20% increase in the correctness of answers, 15% increase in the relevance of answers, and 18% increase in the fidelity of generated content, with excellent performance in scenarios of numerous queries, [13].

Considering these results, we believe that more study should be done on these types of architectures to support numerous users who need to have detailed information in real time.

3 Methodology of Retrieval-Augmented Generation (RAG)

RAG architectures need well-established methodologies and well-organized pipelines so as to improve the accuracy and relevance of the responses produced. One of the key aspects to consider in this type of architecture are the components connected to each other. It is extremely important that the components are well structured so that the pipeline is seamlessly integrated and that there are no bottlenecks where information struggles to arrive.

3.1 Components of RAG

Generative Model (LLM):
Generative models are responsible for generating text based on requests made by users using different interactions, mostly prompt interaction. The learning phase of LLMs requires a massive amount of data to generate the responses.

Retrieval System:
The retrieval system searches for relevant information from external knowledge bases, databases, or documents.
It uses techniques such as semantic search and vector embeddings to find the most pertinent information.

Knowledge Base:
The knowledge base is a repository of structured and unstructured data that the retrieval system can access.
It includes documents, web pages, scientific articles, and other sources of information.

3.2 Process of RAG

The RAG methodology involves the following steps:

Query Processing:

Users ask the system a question via a prompt, and the system processes the content of the query to understand the context.
This process is called tokenisation, i.e. the processing of text into small units called "token".

Information Retrieval:
The retrieval system searches the knowledge base for relevant information using semantic search techniques.
It calculates the similarity between the query and the documents in the knowledge base using vector embeddings, [14], [15].

Contextual Embedding:
All stored information is converted into vector embeddings that represent the context of the query. Embeddings are useful to the system in providing additional knowledge compared to the generative model that was used.

Response Generation:
The generative model uses the contextual embedding along with its training data to generate a coherent and accurate response.
The response is formulated to be contextually relevant and factually correct.

3.3 Algorithms and Mathematical Equations

The basic principles for the operation of RAG architectures are related to the mathematical algorithms and models, below are some of the algorithms and equations that are the essential part for the operation of this type of architectures.

The first one we need to evaluate, and mention is definitely the cosine similarity, which is a measure used to estimate the similarity value between two vectors. This algorithm makes it possible to compare the vector of the query with the vectors of the documents previously inserted into the knowledge base that has been created. To express the value between two vectors A and B, this calculation is performed:

$$\text{cosine\_similarity}(A,B) = \frac{A \cdot B}{\| A \| \| B \|} \tag{1}$$

A and B are the vector representations of the query and the document. The similarity is computed using their scalar product: $A \cdot B$ that measures the degree of alignment between the vectors, while $\| A \|$ and $\| B \|$ denote their magnitudes.

Example:
Query Vector: $A = [1,0,1]$
Document Vector: $B = [0.5,0.5,1]$
Calculating the cosine similarity:

$$\text{cosine\_similarity}(A,B) = \frac{(1 \cdot 0.5) + (0 \cdot 0.5) + (1 \cdot 1)}{\sqrt{1^2 + 0^2 + 1^2} \cdot \sqrt{0.5^2 + 0.5^2 + 1^2}} = \frac{0.5 + 0 + 1}{\sqrt{2} \cdot \sqrt{1.5}} = \frac{1.5}{\sqrt{3}} \approx 0.866 \tag{2}$$

Vector Embeddings
Embeddings transform words or documents into vectors within a continuous space, allowing them to be represented numerically. In this way, they capture the semantic relationships between terms, helping the model to interpret context.
Word2Vec Equation:

$$\text{embedding}(w) = \frac{1}{| C(w) |} \sum_{c \in C(w)} v_c \tag{3}$$

Here, w is the target word, $C(w)$ is the context window around the word, and v_c is the vector representation of the context word.

Example:
The target word is the one we want to represent, while the contextual words are those that surround it and define its meaning. For example, if the target word is "person", the contextual words could be "man" and "woman". The embedding of "person" is calculated by averaging the vectors of the words in its context.

Attention Mechanism
The attention mechanism serves to give more weight to the parts of the input that are most important for generating the response. In this way, the model focuses on relevant information and ignores superfluous details.
The attention mechanism is defined as follows:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

Q is the query matrix, K is the key matrix, V is the value matrix, and d_k is the dimension of the key vectors.

Example:
Query Matrix: $Q = [q_1, q_2, q_3]$
Key Matrix: $K = [k_1, k_2, k_3]$

Value Matrix: $V = [v_1, v_2, v_3]$
Calculating the attention weights:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (5)$$

Confidence Thresholds
Confidence thresholds are used to determine how relevant a retrieved document is. These thresholds allow you to filter out less useful information and keep only the most relevant information to generate responses. The thresholds are based on metrics such as cosine similarity.

Example:
High threshold (0.9): only documents with a similarity greater than 0.9 are considered highly relevant.
Medium threshold (0.7): documents with a similarity between 0.7 and 0.9 are moderately relevant.
Low threshold (0.5): documents with a similarity lower than 0.7 are not very relevant.

3.4 Practical Application

Let's imagine a case in which a user asks for information on the latest advances in energy storage systems. The RAG system takes charge of the query, searches the knowledge base for the most relevant content and generates a response.

Query: "What are the latest advances in the field of energy storage systems?"

Retrieval: the system identifies recent articles on the subject.

Similarity calculation: it compares the query vector with those of the documents using cosine similarity.

Confidence thresholds: only documents with a score above 0.9 are considered highly relevant.

Contextual embedding: the selected texts are converted into vectors (embedding).

Response generation: the model uses these embeddings to formulate an accurate response.

Thanks to this combination of algorithms and metrics, RAG offers a robust solution for producing accurate and contextually relevant responses. This approach has the potential to revolutionise many applications, making the system more reliable and effective in handling complex queries and providing quality information.

4 Demo and Results

For the implementation of the RAG architecture we chose an Iaas solution, inside we have the docker for using the CheshireCat.ai framework, all developed in a python environment. The docker inside has its containers with their own file system that talk to each other but not to the outside. The volumes allow us to connect the dockers with the outside via plugins to allow them to talk to the outside through an internal port to our container with an external port.

The Retrieval-Augmented Generation (RAG) framework leverages both declarative and episodic memory to construct the prompt that is ultimately passed to the Large Language Model (LLM). This step is essential for producing responses that are not only accurate but also contextually appropriate (see Figure 2).

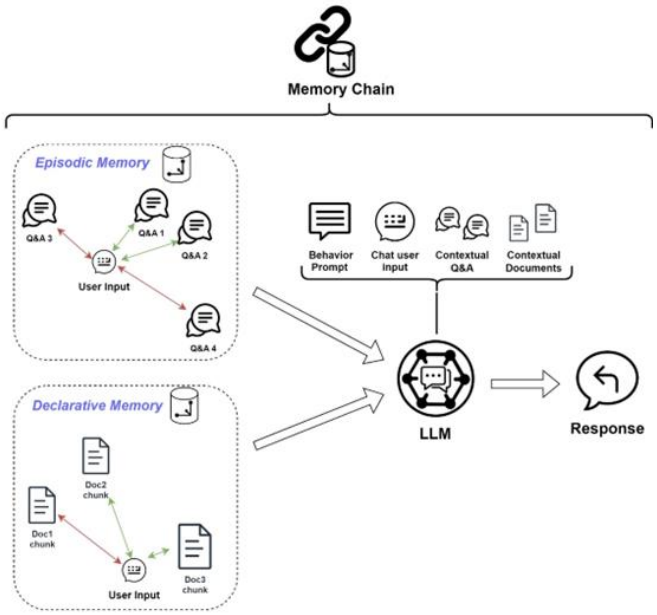

Fig. 2: Declarative and episodic memory process in RAG architetures

The process functions through the integration of the following components:

Declarative Memory
This type of memory is the classic source of knowledge stored and made available by LLM models. This information is supplemented with input from the user who provided reliable and up-to-date content, in this mode the output of LLM models can be improved.

Episodic Memory
Episodic memory takes information from the various interactions and enables exchanges in continuity allowing the system to adapt responses. This component is critical to the smooth functioning of architectures of this type.

Prompt Construction
The generated response is then synthesized with respect to declarative and episodic memory, the combination of this prompt helps to ensure that the LLM model is relevant and has all the necessary information to produce a consistent response with respect to the requests made.

Prompt Structure
User Query: The query posed by the user.
Declarative Memory: Static and structured information retrieved from the retrieval system.
Episodic Memory: Information about past interactions with the user.

An example of questions and answers through the use of LLM chatGPT 4 models, first without adding documentation and later using RAG model embedding through uploading two specific papers:

User: *Deep learning and artificial intelligence methods to smart edge devices and stereo cameras to measure the distance between objects with an Edge AI device stereo cameras, a Luxonis OAK-D, an Intel Myriad-X neural chip and an improved version of the open source Luxonis DepthAI library?*

Bot: *Ah, a curious query! While I can't point to a specific paper, I suggest searching academic databases like IEEE Xplore or arXiv for terms like "deep learning stereo cameras Edge AI" and "Luxonis OAK-D." You'll likely find relevant studies discussing those themes. Happy hunting!*

User: *Deep learning and artificial intelligence methods to smart edge devices and stereo cameras to measure the distance between objects with an Edge AI device stereo cameras, a Luxonis OAK-D, an Intel Myriad-X neural chip and an improved version of the open source Luxonis DepthAI library?*

Bot: *Indeed, there is a paper titled "Applications of deep learning and artificial intelligence methods to smart edge devices and stereo cameras." It discusses the use of a Luxonis OAK-D with an Intel Myriad-X neural chip and the DepthAI library for measuring distances between objects. A delightful read for your interests!*

User: *Method, application and experimental for analysis and simulation of hydraulic and hydrogeological hazards through 3D WebGIS digital platforms?*

Bot: *Ah, the quest for knowledge! I don't have a specific paper at hand, but I recommend exploring journals focused on hydraulic engineering or hydrogeology. Look for keywords like "3D WebGIS," "hydraulic hazards," and "hydrogeological simulation." You'll likely uncover valuable resources!*
User: *Method, application and experimental for analysis and simulation of hydraulic and hydrogeological hazards through 3D WebGIS digital platforms?*

Bot: *Certainly! A common method involves using hydrological models integrated with 3D WebGIS platforms to simulate water flow and assess hazards. Applications like ArcGIS or QGIS can be utilized for visualization and analysis. Experimental setups often include real-time data integration to enhance accuracy. Curious to explore further?*

User: *Method, application and experimental for analysis and simulation of hydraulic and hydrogeological hazards through 3D WebGIS digital platforms?*

Bot: *Ah, yes! There's a paper titled "A System for Analysis and Simulating Hydraulic and Hydrogeological Risks Through WebGIS 3D Digital Platforms" by Mauro Mazzei and Davide Quaroni. It discusses methods, applications, and experimental setups for such analyses. A splendid choice for your inquiry!*

User: *Method, application and experimental for analysis and simulation of hydraulic and hydrogeological hazards through 3D WebGIS digital platforms?*

Bot: *Indeed! A widely used method involves employing hydrological modeling combined with 3D visualization tools in WebGIS platforms. Applications often include flood risk assessment, groundwater management, and stormwater simulation.*

Experimental setups can incorporate real-time data to improve predictive capabilities. Would you like to delve deeper into any specific aspect?

In Figure 3 graphs the quantitative evaluation (accuracy, factual consistency, human relevance

score, response latency, use of tokens per response, user satisfaction) comparing the performance of a basic Large Language Model (LLM) with a version configured from specific structured and unstructured Retrieval-Augmented Generation (RAG) data.

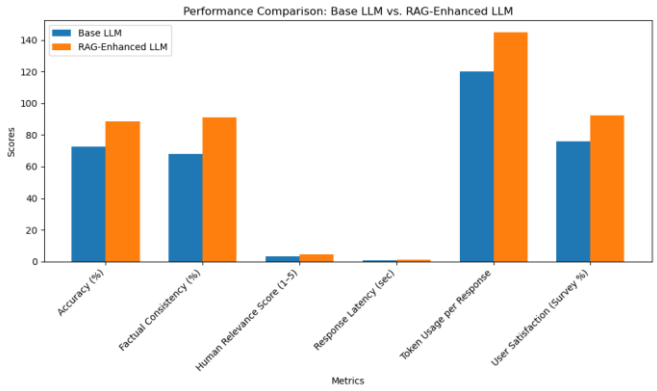

Fig. 3: Quantitative evaluation of the comparison between LLM and RAG

5 Conclusion

The combination of large language models (LLMs) with retrieval-augmented generation (RAG) provides a breakthrough in the field of artificial intelligence.

This type of architecture significantly improves responses in the required context. The most significant improvement is particularly evident in areas where in-depth knowledge is required, and the fact that RAG architectures are designed to autonomously include documents and detailed information in specific areas strengthens these artificial intelligence systems. Multimodal RAG architectures, which integrate heterogeneous data sources such as textual, visual, auditory and possibly sensory modalities, need to be explored and tested. This type of related data can improve the depth and adaptability of content generated in complex and real-world scenarios.

Declaration of Generative AI and AI-assisted Technologies in the Writing Process

The author wrote, reviewed and edited the content as needed and verifies that none utilized artificial intelligence (AI) tools were used. The author takes full responsibility for the content of the publication.

References:

[1] Majjed Al-Qatf, Rafiqul Haque, Saeed Hamood Alsamhi, Samuele Buosi, Muhammad Asif Razzaq, Mohan Timilsina, Ammar Hawbani, Edward Curry: RAG4DS: Retrieval-Augmented Generation for Data Spaces - A Unified Lifecycle, Challenges, and Opportunities. *IEEE Access* 13: 39510-39522 (2025).

[2] Mahd Hindi, Linda Mohammed, Ommama Maaz, Abdulmalik Alwarafy: Enhancing the Precision and Interpretability of Retrieval-Augmented Generation (RAG) in Legal Technology: A Survey. *IEEE Access* 13: 46171-46189 (2025).

[3] Binita Saha, Utsha Saha, Muhammad Zubair Malik: QuIM-RAG: Advancing Retrieval-Augmented Generation With Inverted Question Matching for Enhanced QA Performance. *IEEE Access* 12: 185401-185410 (2024).

[4] Zheng Wang, Shu Xian Teo, Jieer Ouyang, Yongjun Xu, Wei Shi: M-RAG: Reinforcing Large Language Model Performance through Retrieval-Augmented Generation with Multiple Partitions. *ACL* (1) 2024: 1966-1978.

[5] Shenglai Zeng, Jiankun Zhang, Pengfei He, Yiding Liu, Yue Xing, Han Xu, Jie Ren, Yi Chang, Shuaiqiang Wang, Dawei Yin, Jiliang Tang: The Good and The Bad: Exploring Privacy Issues in Retrieval-Augmented Generation (RAG). *ACL* (Findings) 2024: 4505-4524.

[6] Leonardo Pasquarelli, Charles Koutcheme, Arto Hellas: Comparing the Utility, Preference, and Performance of Course Material Search Functionality and Retrieval-Augmented Generation Large Language Model (RAG-LLM) AI Chatbots in Information-Seeking Tasks. CoRR abs/2410.13326 (2024).

[7] Shamane Siriwardhana, Rivindu Weerasekera, Tharindu Kaluarachchi, Elliott Wen, Rajib Rana, Suranga Nanayakkara: Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering. *Trans. Assoc. Comput. Linguistics* 11: 1-17 (2023).

[8] Kangyu Ji, Weizhe Lin, Yuqi Sun, Lin-Song Cui, Javad Shamsi, Yu-Hsien Chiang, Jiawei Chen, Elizabeth M. Tennyson, Linjie Dai, Qingbiao Li, Kyle Frohna, Miguel Anaya, Neil C. Greenham, Samuel D. Stranks: Self-supervised deep learning for tracking degradation of perovskite light-emitting diodes with multispectral imaging. *Nat. Mac. Intell.* 5(11): 1225-1235 (2023).

[9] Weizhe Lin, Rexhina Blloshmi, Bill Byrne, Adrià de Gispert, Gonzalo Iglesias: LI-RAGE: Late Interaction Retrieval Augmented Generation with Explicit Signals for Open-

Domain Table Question Answering. *ACL* (2) 2023: 1557-1566.

[10] Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, Suranga Nanayakkara: Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering. CoRR abs/2210.02627 (2022).

[11] Shivani Singh, Sudhan Majhi, Udit Satija, CNN-RAGNet Architecture for CFO Estimation in RIS-Assisted MIMO-OFDM Systems. *IEEE Commun. Lett.* 29(5): 1092-1096 (2025).

[12] Mahd Hindi, Linda Mohammed, Ommama Maaz, Abdulmalik Alwarafy, Enhancing the Precision and Interpretability of Retrieval-Augmented Generation (RAG) in Legal Technology: A Survey. *IEEE Access* 13: 46171-46189 (2025).

[13] Rong Hu, Sen Liu, Panpan Qi, Jingyi Liu, Fengyuan Li, ICCA-RAG: Intelligent Customs Clearance Assistant Using Retrieval-Augmented Generation (RAG). *IEEE Access* 13: 39711-39726 (2025).

[14] Joan Figuerola Hurtado: Harnessing Retrieval-Augmented Generation (RAG) for Uncovering Knowledge Gaps. CoRR abs/2312.07796 (2023) 2022.

[15] Alexandru Coca, Bo-Hsiang Tseng, Weizhe Lin, Bill Byrne: More Robust Schema-Guided Dialogue State Tracking via Tree-Based Paraphrase Ranking. *EACL* (Findings) 2023: 1413-1424.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

Mauro Mazzei implemented the information retrieval techniques, prepared a dataset and evaluated the results. The author wrote and revised the manuscript.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

This research was financed with funds from the LabGeoInf laboratory of the CNR.

Conflict of Interest

The author declares that he has no conflict of interest relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International , CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0
https://creativecommons.org/licenses/by/4.0/deed.en_US