Journal of Modern Technology and Engineering
Vol.10, No.1, 2025, pp.42-56
https://doi.org/10.62476/jmte.10142

Jomard Publishing

RETRIEVAL-AUGMENTED GENERATION: ARCHITECTURE, TECHNIQUES, AND EVALUATIONS

Şevket Polan*

Ege University, Izmir, Türkiye

Abstract. Retrieval-Augmented Generation (RAG) is an innovative approach that enables large language models to produce more accurate and up-to-date texts by being augmented with external information sources. RAG models operate by combining information retrieval and text generation modules. The information retrieval module extracts the required data from external sources, while the text generation module uses this information to produce a response. The architecture of RAG, along with the roles of information retrieval techniques, embedded re-presentation methods, and vector databases, is explored in detail in the paper. Strategies for context enrichment with large language models, memory mechanisms, and information synthesis processes are discussed, along with the optimization of RAG systems. Techniques employed to enhance the accuracy and consistency of text generation, evaluation metrics, and the challenges encountered in developing RAG-based systems are also addressed. The paper comprehensively presents the prospects and potential areas of development for RAG in artificial intelligence and natural language processing.

Keywords: Retrieval-Augmented Generation, RAG, large language models, generative artificial intelligence.
Corresponding author: Şevket, Polan, Ege University, Izmir, Türkiye, e-mail: *sevket.polan@ege.edu.tr*
Received: 3 September 2024; Revised: 23 December 2024; Accepted: 12 January 2025; Published: 30 April 2025.

1 Introduction

RAG is an innovative approach that enables large language models to generate more accurate and up-to-date texts by integrating external information sources. Since contemporary language models are limited to their training data, they may provide outdated information over time or generate outputs known as "hallucinations," which are not grounded in reality (Gao et al., 2023). To address these issues, RAG combines the processes of information retrieval and text generation, allowing the language model to directly obtain relevant and reliable content from an external knowledge source and base its responses on this information. Particularly in areas such as knowledge-based question-answering systems, text summarization, and dialogue models, RAG helps mitigate the limitations of large language models, enhancing both accuracy and reliability.

RAG models consist of two primary components: the information retrieval module and the text generation module. The information retrieval module enables the model to extract relevant information from large knowledge bases, documents, or vector databases. During this process, embedding models are employed to match queries with the most semantically relevant content. Subsequently, the text generation module integrates the retrieved information with the large language model to generate an appropriate response. Compared to traditional pre-trained text generation models, RAG streamlines the information update process, offering a structure that is more easily updatable, scalable, and cost-efficient. This approach enhances the efficiency of large language models by reducing the need for frequent retraining (Yu, 2022).

This paper will examine the RAG architecture and its components in detail, investigating information retrieval techniques, embedding methods, and the role of vector databases. Context enrichment strategies with large language models, memory mechanisms, and information synthesis processes will be explained, discussing how RAG systems can be optimized. Techniques used to enhance the accuracy and consistency of text generation, such as fine-tuning and adaptation mechanisms, will be evaluated in the context of academic and industrial applications. Additionally, performance factors such as computational costs and hardware requirements of RAG models will be addressed, highlighting critical points to consider in the model development process. Evaluation metrics used to measure the effectiveness of RAG systems will be outlined, with a discussion on the applicability of metrics such as precision, recall, normalized cumulative gain (NDCG), RAGAS, ROUGE, BLEU, and BERTScore for RAG models. The use of these metrics to improve the accuracy and reliability of model outputs will be analyzed. Furthermore, the key challenges encountered in developing RAG-based systems and current research trends in this area will be assessed. A comprehensive discussion will be presented on the future role of the RAG approach in artificial intelligence and natural language processing, potential areas of development, and application scenarios.

2 RAG Architecture and Components

The RAG architecture consists of a two-stage structure, integrating a retrieval component with a text-generating language model. In the first stage, the information retrieval module finds text fragments related to the query from external information sources. In the second stage, the generation module uses the retrieved contextual information to produce the final response (Khattab et al., 2022). When we include data preprocessing for this architecture, the entire process typically consists of three steps:

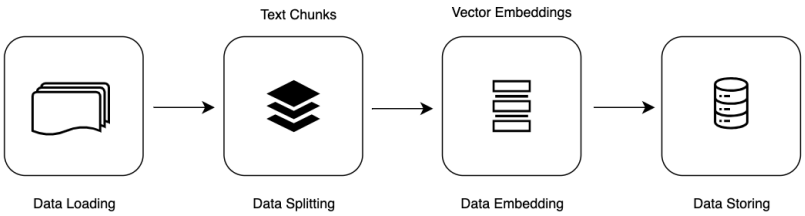

Figure 1: Data preprocessing

Indexing: Documents from external information sources are divided into small chunks (paragraphs or sentence blocks), with each chunk represented in an embedded vector space, and these vectors are stored in a vector database. This step is critical for enabling fast similarity searches in the future. For example, texts can be converted into vectors using a BERT-based model and stored in a vector database like FAISS. Such a vector indexing infrastructure enables meaningful similarity searches even with billions of data points.

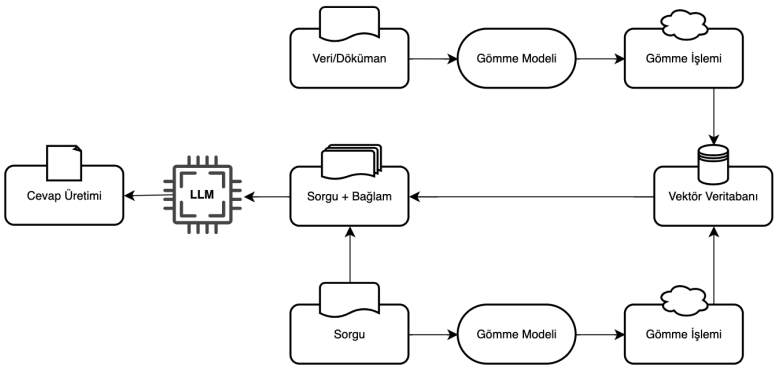

Figure 2: RAG pipeline

Retrieval: When a user query is received, the same embedding model is applied to the query to obtain the query vector. Then, the vector database is quickly searched for the vectors most similar to the query, and the top k most relevant text chunks (e.g., the first 5-10 most similar paragraphs) are retrieved. The similarity measure typically used in this step is cosine similarity, and the accuracy of the results largely depends on the quality of the embedding model used. For example, the similarity between a query vector and document vectors is computed, and the highest-scoring chunks are selected as context. The retrieved chunks are expected to be truly relevant to the query and contribute to the final response.

Algorithm 1: RAG Pipeline Pseudocode

Input: Query Q
Output: Response R
Step 1: Create Vector Database
Create a vector database from the dataset using proper embedding of the LLM model
$D \leftarrow$ Dataset
$D_{\text{enc}} \leftarrow \text{Encode}(D)$ ▷ Convert the dataset into vector representations using the LLM's embedding function
Step 2: Embed Query
Embed the Query for vector similarity search
$Q_{\text{enc}} \leftarrow \text{Embed}(Q)$ ▷ Convert the query into a vector representation
Step 3: Vector Search
Rank chunks (document parts) based on relevance to Q using vector similarity search on embedded data and retrieve most similar n chunks
$D_{\text{ranked}} \leftarrow \text{RankChunks}(Q_{\text{enc}}, D_{\text{enc}})$ ▷ Rank chunks based on the similarity between query and dataset vectors
$C \leftarrow \text{RetrieveChunks}(D_{\text{ranked}})$ ▷ Retrieve the top-ranked chunks
Step 4: Generate Response
Generate a response using LLM with the most similar chunks, system prompt, and user query
$R \leftarrow \text{GenerateResponse}(Q, C)$ ▷ Pass the query and relevant chunks to the LLM for response generation
Step 5: Output Response
Display the response along with relevant document parts for responding to the query
Output: R ▷ Return the generated response along with the relevant document sections

Generation: In the final stage, the original user query and the retrieved relevant text chunks are combined and fed into the language model, enabling the model to generate a response based

on this expanded context. The large language model (e.g., a GPT-based model) uses both the internally learned information from its parameters and the externally provided up-to-date information to generate a response that is appropriate and reasoned about the user's question. This allows the model to incorporate current or domain-specific knowledge that may not be present in its training data into its response.

This architecture fundamentally operates in a sequential "retrieve-and-read" manner and has become a standard approach today (Es et al., 2024). While a large language model such as ChatGPT cannot independently answer questions about recent events, the RAG approach bridges this gap by providing the model with external, up-to-date articles. This enables the model to generate more accurate responses based on current information (Gao et al., 2023). Compared to models that rely solely on parametric knowledge, this method not only enhances the accuracy of responses but also facilitates the traceability of model outputs. Specifically, RAG ensures source attribution by presenting the textual sources that support its responses, thereby making it possible to determine where the model obtained its information (Ram et al., 2023). In this regard, RAG adds transparency to generative models and is considered crucial for developing reliable real-world systems.

Early implementations of the RAG architecture primarily focused on retrieving information from a single large text collection. For instance, in original open-domain question-answering studies, the model retrieved relevant articles solely from a knowledge base such as Wikipedia before generating a response. However, retrieving information from a single source can be inherently limited due to gaps or biases within that source. Recent research has sought to enrich RAG systems by retrieving information from heterogeneous sources, including multiple databases, memory structures, and even knowledge graphs. By doing so, models can generate more comprehensive and reliable responses by leveraging different types of data (e.g., text, tables, graphs). A RAG system, for example, can simultaneously retrieve relevant passages from a document and a knowledge graph, integrating both sources to construct a more coherent and contextually accurate response.

2.1 Retrieval Techniques

The retrieval step in RAG processes is a critical component for the system's success. The primary objective of this step is to extract the most relevant information from a vast knowledge corpus with high accuracy in response to a user query. Currently, two fundamental retrieval approaches are utilized in RAG systems: (1) traditional keyword-based search methods (e.g., indexing algorithms such as BM25) and (2) embedding-based dense retrieval techniques.

Traditional search methods represent queries and documents based on term frequencies. For instance, the BM25 algorithm computes a relevance score by considering the frequency of query terms in the document and statistical measures such as TF-IDF. These methods are particularly efficient for queries that contain exact matching keywords; however, their performance declines in cases involving synonyms or paraphrasing.

On the other hand, dense retrieval methods represent each sentence or paragraph as a vector (embedding) in a semantic space and perform searches based on the similarity between the query vector and document vectors. This approach excels at identifying semantically relevant content even in the absence of exact word matches. For example, in a RAG system, a BERT-based embedding model can be employed to generate vector representations for queries and all sentences in the database. The system then retrieves the most relevant texts by identifying vectors with the highest cosine similarity to the query vector. Consequently, the model can capture content that conveys the same meaning despite being expressed in different words.

Vector databases play a crucial role in enabling searches using embedding representations. These databases are specialized systems optimized for efficiently storing high-dimensional vectors and performing similarity searches among them. In recent years, several vector database management systems (VDBMS) have been developed, including FAISS, Milvus, Weaviate, and

Pinecone. These systems are capable of conducting nearest-neighbor searches among billions of vectors within milliseconds. For instance, a RAG application leveraging Pinecone or Weaviate can store the vector representations of an entire encyclopedia and rapidly return the closest neighbor vectors to a given query (Wang et al., 2025). Such infrastructures have significantly enhanced the scalability of RAG systems, making real-time searches across massive knowledge repositories feasible.

An essential consideration in information retrieval techniques is the ranking and filtering of results. To prioritize the most relevant documents among the retrieved results, re-ranking algorithms are frequently employed. For example, in an initial retrieval step, a dense search method may return 100 paragraphs, which are then re-evaluated using a BERT-based cross-encoder model to select the top 5 paragraphs. This multi-stage retrieval strategy improves retrieval accuracy, ensuring that the language model is provided with genuinely useful information. Empirical studies have demonstrated that such hierarchical retrieval approaches enhance RAG performance (Chen et al., 2024; Miao et al., 2024; Wu et al., 2024).

Nonetheless, the retrieval component faces several challenges. Balancing precision and recall is often difficult; retrieving an excessive number of texts may introduce irrelevant information (low precision), while retrieving too few may omit crucial details (low recall). In practice, the retrieval phase may occasionally return passages that are only partially relevant or entirely unrelated to the query. This issue can lead to the language model being supplied with unnecessary or incorrect information. For instance, if a RAG system fails to retrieve a sentence containing critical information (recall error), the generated response will be incomplete. Conversely, if an irrelevant sentence is included as context (low precision), the model may produce extraneous details or erroneous outputs. To mitigate these challenges, researchers have explored optimization techniques such as query generalization/narrowing, query expansion, and result set aggregation (Gao et al., 2023). For example, rewriting the query before retrieval to achieve better search results or filtering retrieved results post-retrieval to retain only critical sentences are among the strategies employed. Information retrieval techniques in RAG systems integrate both classical search approaches and modern deep learning-based dense retrieval methods. The intelligent utilization of embedding models and vector databases is a fundamental determinant of RAG's effectiveness. A RAG system capable of retrieving the right information at the right moment effectively extends the boundaries of language models, equipping them with up-to-date and comprehensive knowledge.

2.2 Embedding Models and Vector Databases

In RAG approaches, embedding models are fundamental components that generate numerical vector representations for both user queries and texts stored in the database. These models enable the semantic content of texts to be expressed as high-dimensional vectors, allowing semantic similarity to be calculated as geometric proximity. For instance, a sentence can be represented as a point in a 768-dimensional vector space using a model such as BERT or RoBERTa. As a result, the vector of a question asking about city populations would be positioned close in space to the vector of a sentence containing the information "The population of Ankara is 5.6 million."

Embedding models can be considered the "brain" of a RAG system, capturing semantic meaning. The stronger these models are, the more accurate the retrieval results will be. In recent years, sentence-level embedding models such as Sentence-BERT and embedding versions of large language models like OpenAI's ada-002 have become widespread. These models, trained on millions of sentences, map semantically similar sentences to closely related vectors. For example, even though the question "When was the COVID vaccine developed?" and the sentence "The COVID-19 vaccine was developed in 2020" do not share identical words, they contain a similar topic and thus generate similar vectors.

Once embedding representations are obtained, vector databases come into play for storing and searching them. Unlike traditional relational databases, vector databases index data within

a geometric space. These systems are optimized to solve the nearest neighbor search problem at a massive scale. Finding the closest 100 vectors among millions during a search can be executed extremely quickly through specialized algorithms (such as LSH and HNSW) and data structures.

Modern vector databases form the backbone of RAG applications. Open-source VDBMS systems like Weaviate and Qdrant, or cloud-based services like Pinecone, provide developers with scalable embedding search capabilities. These systems act as the external memory of large language models, instantly retrieving the necessary information. As Wang et al. (2025) point out, vector databases have become critical infrastructure for advanced LLM applications like RAG. For instance, searching a database containing 2 trillion tokens from hundreds of millions of articles (as seen in DeepMind's RETRO study) was only made possible through efficient vector indexing and compression techniques. Borgeaud et al. (2022) demonstrated the power of embedding representations and vector databases in their Retrieval-Enhanced Transformer (RETRO) model, achieving GPT-3-level performance with a model 25 times smaller by searching within a dataset of 2 trillion tokens.

The interaction between embedding models and vector databases is one of the key success factors of RAG. When designing a RAG system, careful consideration must be given to selecting the appropriate embedding model and determining how vectors will be stored. Additionally, techniques such as clustering, dimensionality reduction, or quantization can be applied to optimize the vector space. For example, 768-dimensional vectors can be reduced to 256 dimensions to save memory and increase speed, or suitable quantization methods can be used to minimize memory usage while maintaining search accuracy.

Embedding representations serve as tools that "teach" external knowledge to language models in a meaningful way, while vector databases act as intelligent memory banks that store and retrieve this knowledge efficiently. In RAG processes, the compatibility between these two elements determines both the speed and accuracy performance of the system.

2.3 Large Language Models and Context Augmentation

In RAG systems, the LLM is the component that processes a context enriched with external information and generates the final textual output. At this stage, the model combines both the internal knowledge embedded in its learned parameters and the external information provided by the retrieval module to answer the user's query. Context enrichment involves not only incorporating the user's query but also relevant supplementary information into the LLM's input. This approach enables the model to be augmented with information it may not independently possess or that is not contained within its learned parameters. For instance, if a model like ChatGPT was trained up until 2024, and a question is posed about an event occurring in 2025, the model would be unable to provide an accurate response. However, through the RAG approach, if news articles regarding the event are injected into the model's context, the model can respond based on current information. As Gao et al. (2023) stated, this methodology allows LLMs to access up-to-date knowledge, thereby addressing the limitation of relying solely on training data.

During the context enrichment process, the external information presented to the model must be in a format comprehensible to the model and provided in the appropriate quantity. Typically, the retrieved documents are incorporated directly as text fragments into the prompt. For example:

```
Query: [your query here]
Context: [the provided paragraph]
Response: [the generated response here]
```

When such an introduction is prepared, the language model can use this context to generate a response. Shi et al. (2024) demonstrated the power of this approach in their work, REPLUG.

REPLUG significantly improved performance by simply adding external documents to the language model (such as GPT-3) without making any architectural changes. They optimized the retrieval module using the outputs produced by the model, which resulted in a 6.3% improvement in GPT-3's language modeling performance. This result highlights the effectiveness of augmenting LLMs with external context.

Another benefit of context enrichment is that it reduces the hallucination rate in language model responses. When the model relies on trustworthy text in front of it rather than attempting to recall information from its parameters, the accuracy of the response increases. Mallen et al. (2023) observed in their experiments that large language models struggle with unpopular (rare) information, but when supported with external information, they overcome this difficulty. Especially in areas like lesser-known historical events or specific technical details, where the likelihood of such topics being contained within the model's parameters is low, the RAG approach enables the model to "remember" from the external world. This enhances the reliability of the response.

However, feeding the model with excessive or unnecessary information is not ideal. Due to context length limitations, LLMs can suffer performance degradation when processing too much text. A study revealed that language models struggle to use information in the middle of very long inputs. Liu et al. (2023), in their research titled "Lost in the Middle," demonstrated that models are more sensitive to information at the beginning and end of long contexts and may neglect the information in the middle. Therefore, it is crucial for the context provided in RAG systems to be selective and concise. The retrieved fragments should include only the critical sentences necessary to answer the question and be free from unnecessary details. Some approaches opt to first select relevant sentences from the context and present only those, allowing the model to work with focused attention rather than distributed attention.

Context enrichment is also used in multi-step reasoning problems. For example, in an indirect question like "First, X happened, then Y occurred; what is the cause of Y?" the model needs to combine information from multiple steps. In such cases, RAG allows the model to update the context step by step, enabling it to conclude multiple retrieval-generation cycles. Khattab et al. (2022) introduced the Demonstrate-Search-Predict (DSP) method, which solves complex problems by interacting with the LLM with the search module in a sequential program. For instance, the model first performs a "demonstrate" step to break down the question into subparts, then searches for each part, and finally generates an answer by combining the results. This approach allows the model to answer difficult questions more reliably, which it might not be able to solve in a single pass.Enriching large language models' contexts with external information forms the foundation of RAG. This enables LLMs to access both up-to-date and specialized information. As the accuracy and reliability of the model's output increase, the source of the output can also be made available when needed. The effective use of context is one of the most important success criteria for RAG systems, and when applied correctly, it can significantly enhance the performance of the language model.

2.4 Large Language Models and Context Augmentation

The final stage in RAG processes is the text generation phase, where a LLM produces a final response based on the retrieved context. At this stage, the model processes the user's query alongside the provided supplementary information to generate a coherent and accurate response. The quality of the generated output is contingent upon both the model's intrinsic capabilities and the relevance of the retrieved context. In the context of RAG, various optimization techniques have been developed to enhance the effectiveness of this stage.

The primary objective is to ensure that the model's response maintains high levels of accuracy and consistency. Effective utilization of the retrieved context plays a critical role in achieving this goal. Ideally, the model should articulate the information from the provided sources in its own words without diverging from the original content. However, there are instances where the model

may misinterpret details or integrate unrelated information. For example, if the model is asked, "What is the capital of France?" while being provided with an irrelevant historical account of Paris, it may generate a response discussing the city's history rather than directly answering the query. To mitigate such issues, researchers implement various control and balancing mechanisms to refine the model's outputs.

One prominent optimization approach involves jointly training the retrieval and generation modules. Traditionally, search models and language models are optimized separately, yet studies such as the original work on RAG by Lewis et al. (2020) have explored methodologies that train these two components together toward a shared objective. However, given the computational constraints associated with large-scale language models, this approach is not always practical. Consequently, alternative strategies have been investigated to enhance the synergy between retrievers and generators.

For instance, the REPLUG method enhances the retriever model using reverse feedback from the language model's outputs. Here, the LLM identifies the documents that best support its responses, and the retriever model learns from these preferences to improve future retrievals. This method significantly enhances the quality of the contextual information fed into the generation stage, thereby improving overall model performance.

Similarly, the Augmentation-Adapted Retriever (AAR) approach aims to provide a retriever model that generalizes across different language models. Yu et al. (2024) trained a retriever based on the preferences of a well-established language model and applied it to larger, immutable models (i.e., frozen models). This strategy allows the retriever to be fine-tuned on a smaller model while subsequently optimizing the zero-shot performance of a larger model. Notably, this approach optimizes the retriever without necessitating any modifications or fine-tuning of the language model itself, demonstrating an efficient means of enhancing text generation quality.

Another technique used to improve text generation is iterative refinement, wherein the model iteratively updates its responses based on feedback. If an initial response fails to meet quality expectations, the model can reformulate the question or modify its generated answer before re-engaging in the RAG cycle. This method is particularly effective for open-ended and complex queries. Within the literature, self-ask (where the model identifies gaps in its response and queries itself) and self-checking (where the model verifies its own answer) have been explored (Khattab et al., 2022). For example, under the DSP framework, a model may initially draft a partial response, identify missing elements, retrieve additional information, and refine its final answer accordingly. This iterative refinement process is particularly beneficial for multi-hop reasoning tasks, ensuring that essential information is incorporated into the response.

To enhance factual accuracy in generated text, post-verification mechanisms can also be integrated into the generation phase. A notable example is SelfCheckGPT (Manakul et al., 2023), which employs the language model itself to validate its own responses. In a similar vein, FactScore, developed by Min et al. (2023), evaluates long-form responses by segmenting them into atomic propositions and assessing whether each claim is supported by credible sources. This approach has revealed that even state-of-the-art language models, such as ChatGPT, exhibit factually accurate sentence rates of approximately 58% in biographical texts. These findings underscore the need for continued advancements in verification methodologies to further reduce the risk of factual inaccuracies.

Ultimately, optimizations in the text generation phase significantly influence the overall performance of RAG systems. Enhancing the alignment between retrieval and generation modules, determining when and how much external information is required (e.g., direct answers for simple queries versus multi-step retrieval for complex ones) (Jeong et al., 2024), and integrating verification mechanisms remain focal points of ongoing research. The overarching goal of these optimizations is to minimize errors while producing highly informative and factually reliable responses. Consequently, RAG systems continue to evolve through advancements in both retrieval and generation components to achieve this objective.

3 Evaluation Metrics and Criteria

Evaluating the performance of RAG systems is inherently more complex compared to traditional language model assessments. This complexity arises from the fact that the effectiveness of RAG systems depends on both the retrieval and generation components (Es et al., 2024). Consequently, evaluation processes incorporate metrics that assess these two components separately and in combination.

For the retrieval phase, standard metrics commonly used in the field of information retrieval (IR) are applied. Precision measures the proportion of retrieved results that are relevant, while recall quantifies the proportion of relevant results that have been successfully retrieved. For instance, if a RAG system retrieves 10 documents and 8 of them contain useful information related to the query, the precision is 80%. If there were 5 relevant documents in total and the system retrieved 4 of them, the recall would be calculated as 80%. Other metrics such as Top-n Precision, Mean Average Precision (MAP), and Normalized Discounted Cumulative Gain (NDCG) are also employed to evaluate the ranked retrieval performance. In particular, NDCG measures how well the retrieved documents are ordered by relevance, indicating the system's ability to prioritize the most pertinent documents at the top. Achieving high relevance and coverage in the retrieval module is crucial for ensuring that the model is provided with accurate and comprehensive contextual information.

The evaluation of the generation phase, on the other hand, relies on traditional text generation metrics. If the RAG system is tested on a dataset with reference answers, the appropriateness of the generated responses can be numerically measured using metrics such as ROUGE and BLEU. For example, ROUGE metrics assess the overlap between the generated response and the reference response based on n-grams, while BLEU, widely used in machine translation, evaluates precision based on n-gram matches between the generated text and the reference. A high BLEU or ROUGE score suggests that the model's output closely aligns with the reference expressions. However, these metrics do not inherently capture semantic meaning, as they rely on exact word overlaps. Consequently, they may fail to properly assess responses that are paraphrased or expressed differently but hold the same meaning.

To address this limitation, BERTScore was developed as an alternative evaluation metric that considers semantic similarity. BERTScore compares the embeddings of words in the generated text with those in the reference text, allowing for a more nuanced assessment of meaning. For instance, the sentences "The city was very crowded" and "The population density in the city was high" may receive a high similarity score with BERTScore, whereas traditional word-overlap-based metrics would yield a lower score. Given this advantage, BERTScore is frequently utilized in evaluating RAG-generated outputs to ensure a more meaning-focused assessment.

In some cases, reference answers may not be available for evaluating RAG systems, particularly in open-ended questions or broad information-seeking tasks. To address such scenarios, frameworks like RAGAS (Retrieval-Augmented Generation Assessment) have been introduced. Proposed in 2023, RAGAS offers a suite of metrics specifically designed to evaluate RAG systems without requiring reference answers. This framework assesses multiple dimensions, including the relevance of the retrieved context, the accuracy of the model's utilization of this context, and the coherence and fluency of the generated response. RAGAS employs sub-metrics to determine whether the retrieved passages are genuinely relevant to the query and whether the generated response remains faithful to the provided context. By doing so, it enables a structured evaluation of different aspects of RAG performance without necessitating human-labeled reference responses.

As emphasized by Yu et al. (2024) in their comprehensive evaluation study, assessing RAG systems is a multi-faceted challenge. To evaluate a RAG-generated response, several key questions should be considered:

(1) Relevance: Has the model retrieved the correct and relevant information?

(2) Accuracy: Does the response contain the necessary information, and is it factually correct?

(3) Faithfulness: Does the response remain true to the provided context and factual knowledge?

For instance, if a model generates a response that includes details absent from the retrieved context, this would indicate low faithfulness, meaning the system is prone to hallucination. Even if the response appears superficially relevant, it may be penalized in evaluation if it lacks verifiable support. To systematically assess factual consistency, FactScore has been introduced as a metric that verifies whether each factual statement in the generated response is backed by a reliable source.

Some evaluation frameworks incorporate human feedback or model-based critics into the assessment process. For example, LLMs can serve as evaluators, where a strong language model is provided with the query, response, and original context and is asked: "Is this response accurate, complete, or biased?". Studies employing GPT-4 as an evaluator have reported that this approach captures subtleties missed by traditional automated metrics. While such methods may not be entirely error-free, they offer a more comprehensive assessment compared to conventional evaluation techniques.

Evaluating RAG systems requires a multi-metric approach that combines both retrieval and generation performance assessments. Success is not merely about achieving a high ROUGE score; rather, it entails producing responses that are grounded in reliable sources, relevant to the query, and factually accurate. Accordingly, rigorous evaluation protocols are being developed to holistically measure both retrieval and generation effectiveness. In the future, establishing a standardized RAG evaluation framework will significantly enhance the ability to compare different systems and methodologies.

4 Challenges Encountered

RAG systems are an innovative approach that combines information retrieval and language generation components to enable language models to produce more accurate and consistent responses. These systems allow the model to generate more reliable results by leveraging external data sources. However, the integration of these two modules introduces various technical and practical challenges. This section outlines the main challenges faced by RAG systems, along with brief explanations of each.

Table 1. Challenges of Retrieval-Augmented Generation Systems

Category	Challenge	Description
Retrieval Module	Retrieval of irrelevant content	The system may retrieve contextually unrelated results due to lexical similarity, leading to incorrect or off-topic answers.
	Missing essential information	If the key sentence needed to answer a question isn't retrieved, the model can't produce a correct response (low recall).
	Scalability and latency issues	Searching through large datasets increases memory usage and slows down retrieval, which is critical for real-time applications.
	Dynamic data indexing	Continuously updating databases require reindexing or advanced strategies to efficiently retrieve new information.
Generation Module	Hallucination problem	When the retrieved context is incomplete or incorrect, the model may fabricate plausible but false information.
	Lack of faithfulness to context	The model may misinterpret or confuse contextual details, especially in long or complex documents.
	Difficulty in integrating conflicting information	Conflicting details from multiple sources (e.g., differing birth years) may result in inconsistent or fragmented answers.
	Loss of information in long contexts	Models often ignore or poorly process middle sections in long inputs, even if critical information is present there ("Lost in the Middle" issue).
Alignment & Integration	Style mismatch between IR and NLG components	The language style of retrieved texts may not align with the model's training, reducing understanding and generation quality.
	Format mismatch	Retrieved information may not be in a format easily usable by the language model (e.g., extracting a number for a "how many?" question).
	Computational cost	RAG systems require extra computation compared to standalone LLMs—retrieval, generation, and sometimes iterative steps increase latency and resource usage.
	Need for adaptive strategies	Techniques like Adaptive-RAG avoid retrieval for simple questions, reducing costs by activating retrieval only when necessary.

While RAG systems have great potential, overcoming the challenges in the retrieval and generation processes is crucial for their effective operation. Issues such as fast access to accurate information, proper utilization of that information, preventing hallucinations, and ensuring system efficiency remain significant problems to be solved. These challenges represent an active area of research, and addressing them will enhance the reliability of RAG systems.

5 Future Trends

The rapid advancements in the field of RAG indicate that this approach will become even more sophisticated and widespread in the future. Some key trends that researchers are expected to focus on in the coming years include:

Modular and Adaptive RAG: Future RAG architectures are expected to become more modular and adaptable. The "Modular RAG" paradigm described by Gao et al. (2023) trans-

forms the RAG process from a linear pipeline into a flexible structure composed of modules that can branch and merge as needed. This allows different retrieval strategies or language model modules to be used depending on the query type. For instance, if a query requires visual data, an image retrieval module can be activated (multimodal RAG), or if a query involves multi-step reasoning, a logic chain module can be added. This flexibility aims to enhance RAG systems beyond a single rigid design, evolving them into a framework enriched with various subcomponents.

Additionally, the concept of Adaptive RAG will gain importance. Intelligent systems capable of dynamically selecting strategies based on query complexity may emerge. For example, simple questions could receive direct answers, medium-complexity questions could undergo a single-step RAG process, and highly complex questions could trigger multi-step RAG. This approach would enhance efficiency by preventing unnecessary computations while applying full processing power when needed to solve difficult problems.

Advanced Retrieval Techniques and Embedding Models: On the retrieval side of RAG, research will continue on improved embedding models and search techniques. Particularly, non-open-domain and multilingual retrieval will gain importance. Current systems predominantly focus on English and general knowledge. In the future, domain-specific RAG applications—such as in medicine and law—will require embedding models that better capture the terminology and linguistic nuances of these fields.

Moreover, semantic inference methods may replace traditional semantic indexing. Instead of mere keyword matching, systems could be designed to understand the intent behind a query and retrieve relevant background information using logical inference.

Efforts will also continue to improve the reliability and scalability of vector databases, which are fundamental to RAG infrastructures. The study "Towards Reliable VDBMS" (Wang et al., 2025) has already raised concerns regarding the robustness and fault tolerance of vector databases. As these databases mature, we can expect more reliable and error-free retrieval layers. Additionally, real-time indexing capabilities may be developed, enabling the integration of the latest information—such as breaking news—into the vector space within minutes. This would help keep RAG systems constantly updated.

Diversification of Information Sources (Multi-Modality): Future RAG systems will leverage not just text but multiple data types to generate responses. If the answer to a question is embedded in an image, the system will be able to retrieve and interpret it using a visual processing module. Approaches like Knowledge-Graph Augmented RAG (Matsumoto et al., 2024) aim to incorporate structured data into the RAG loop. This will enable models to extract information not only from plain text but also from knowledge graphs and databases.

For example, when asked about a country's economic data, a RAG system could connect to a statistical database, fetch relevant figures, and generate a structured response. This represents a higher level of integration than traditional document retrieval approaches.

Similarly, tool-augmented RAG will become more prevalent. A language model could dynamically call external tools such as a calculator or a calendar to enhance its responses (LLM + Tools approach). When combined with RAG, this capability would give the model both memory (text retrieval) and computation power. For example, if asked, "How much did Turkey's population increase in 2023?", a RAG system could retrieve the population data for 2022 and 2023 and then compute the difference using an external tool before delivering the final answer. These combinatorial abilities will form the foundation of future intelligent information assistants.

Evaluation and Reliability: The evaluation of RAG systems will also become standardized and more comprehensive in the future. Although frameworks like RAGAS have recently emerged, universally accepted benchmarks and evaluation protocols for comparing different RAG systems will likely be established. Unified scoring methods that assess both retrieval quality and response accuracy may be developed.

A significant step in this direction could be the introduction of a single composite metric

that summarizes a RAG system's overall performance—similar to how the BLEU score became widely used in machine translation. Additionally, reliability remains a critical issue. Future systems will be designed to detect when they are uncertain and either alert the user or refrain from providing an answer. A RAG model that can say, "Sorry, I don't have information on that," instead of generating incorrect responses will score higher in terms of trustworthiness.

Human Feedback-Driven Development: Reinforcement Learning with Human Feedback (RLHF) played a major role in training models like ChatGPT. Similarly, RAG systems may enter a phase where they are fine-tuned based on human feedback. Human evaluators could assess whether retrieved contexts are relevant and whether responses are satisfactory, helping to improve the system iteratively.

This approach will be particularly valuable in corporate applications, such as customer support bots. By collecting real-world user interactions, RAG systems can continuously learn and enhance their performance. Early models like BlenderBot 3 (Shuster et al., 2022) attempted to update themselves based on live user feedback. In the future, this approach will become more refined, allowing RAG systems to improve even after deployment through online learning.

Expansion of Application Domains: Until now, RAG has primarily focused on general knowledge and question-answering tasks. However, in the future, domain-specific and industrial applications will expand.

For instance, legal AI assistants utilizing RAG could analyze court rulings from a legal database and provide relevant precedents to lawyers. In the medical field, RAG systems could retrieve information from past patient records and the latest medical literature to answer doctors' queries. In fact, a specialized RAG model was developed in 2024 for cancer clinical trials, with key performance results documented. Such domain-specific systems will represent customized adaptations of general RAG technology.

Retrieval-Augmented Generation has opened new frontiers in AI by combining the vast capabilities of large language models with real-world knowledge retrieval. In the coming years, we will see more flexible, reliable, and intelligent RAG systems.

These advancements will lead to systems that can integrate heterogeneous information sources, utilize external tools when needed, and provide users with trustworthy responses. As this evolution continues, academic research on RAG will further accelerate innovation in areas such as architectural improvements, optimization techniques, and evaluation methodologies.By enabling seamless retrieval and intelligent synthesis of information, the RAG paradigm is poised to play a central role in the future of AI-powered information systems.

6 Conclusion

RAG approach is an innovative framework that enables large language models to generate more accurate, up-to-date, and traceable text by integrating external knowledge. This paper provides a detailed analysis of the architectural structure and fundamental components of RAG processes, including the working principles of the retrieval module, the use of embedding models and vector databases, the enrichment of the language model with context, and the optimization of the text generation phase. Since evaluating the performance of RAG systems requires a hybrid assessment beyond traditional metrics, both retrieval and generation dimensions are discussed. In light of current research, the challenges faced by RAG systems (such as hallucination, incorrect retrieval, and inconsistency issues) and the proposed solutions are presented. Finally, future directions in the field of RAG are evaluated, emphasizing the emergence of more modular, multimodal, adaptable, and reliable RAG systems.

As a discipline at the intersection of natural language processing and information retrieval, RAG has rapidly advanced in recent years. Academic studies conducted over the past few years have demonstrated RAG's potential from various perspectives, repeatedly proving that the RAG architecture is an effective solution for developing knowledge-based AI systems. Particularly in

knowledge-intensive tasks, RAG methods have achieved strong results, enhancing the language capabilities of traditional language model approaches (Khattab et al., 2022). However, for RAG applications to fully mature, there are still unresolved issues and areas requiring optimization. Ongoing research continues to introduce new methods to make RAG systems more efficient.

In conclusion, a technical literature review of RAG processes clearly highlights both the strengths and limitations of this approach. This architecture, which combines information retrieval and text generation, symbolizes AI systems' ability to "speak with knowledge." In the future, it is highly likely that RAG will become a standard component across various industries and applications. Findings from academic studies confirm that, when applied correctly, RAG enhances large language models by making them more reliable, up-to-date, and functional. Accordingly, we can anticipate groundbreaking developments in both experimental studies and real-world applications of RAG-based systems in the coming years.

References

Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., ... & Sifre, L. (2022). Improving language models by retrieving from trillions of tokens. *Proceedings of the 39th International Conference on Machine Learning (ICML)*, *162*, 2206–2240.

Chen, J., Lin, H., Han, X., & Sun, L. (2024, March). Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 38, No. 16, pp. 17754–17762).

Es, S., James, J., Anke, L. E., & Schockaert, S. (2024, March). RAGAS: Automated evaluation of retrieval-augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations* (pp. 150–158).

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2.

Hung, T.K., Kuperman, G.J., Sherman, E.J., Ho, A.L., Weng, C., Pfister, D.G., & Mao, J.J. (2024). Performance of retrieval-augmented large language models to recommend head and neck cancer clinical trials. *Journal of Medical Internet Research*, *26*, e60695. `https://doi.org/10.2196/60695`

Jeong, S., Baek, J., Cho, S., Hwang, S.J., & Park, J.C. (2024). Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity.

Khattab, O., Santhanam, K., Li, X. L., Hall, D., Liang, P., Potts, C., & Zaharia, M. (2022). Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv:2212.14024*.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., & Liang, P. (2023). Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Liu, Y., Gan, A., Zhang, K., Tong, S., Liu, Q., & Liu, Z. (2024, August). Evaluation of retrieval-augmented generation: A survey. In *CCF Conference on Big Data* (pp. 102–120). Singapore: Springer Nature Singapore.

Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., & Hajishirzi, H. (2023). When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of ACL 2023 (Long Papers)* (pp. 9802–9822).

Matsumoto, N., Moran, J., Choi, H., Hernandez, M. E., Venkatesan, M., Wang, P., & Moore, J. H. (2024). KRAGEN: A knowledge graph-enhanced RAG framework for biomedical problem solving using large language models. *Bioinformatics (Oxford, England)*, *40*(6), btae353. https://doi.org/10.1093/bioinformatics/btae353

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W., Koh, P.W., Iyyer, M., Zettlemoyer, L., & Hajishirzi, H. (2023). FActScore: Fine-grained atomic evaluation of factual precision in long-form text generation.

Miao, J., Thongprayoon, C., Suppadungsuk, S., Garcia Valencia, O.A., & Cheungpasitporn, W. (2024). Integrating retrieval-augmented generation with large language models in nephrology: Advancing practical applications. *Medicina*, *60*(3), 445.

Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., & Shoham, Y. (2023). In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, *11*, 1316–1331.

Shi, W., Min, S., Yasunaga, M., Seo, M., James, R., Lewis, M., Zettlemoyer, L., & Yih, W. (2024). REPLUG: Retrieval-augmented black-box language models. In *Proceedings of NAACL 2024 (Long Papers)* (pp. 8371–8384).

Shuster, K., Xu, J., Komeili, M., Ju, D., Smith, E.M., Roller, S., ... Weston, J. (2022). Blenderbot 3: A deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*.

Wang, S., Zhao, Y., Xie, Y., Liu, Z., Hou, X., Zou, Q., & Wang, H. (2025). Towards reliable vector database management systems: A software testing roadmap for 2030. *arXiv preprint arXiv:2502.20812*.

Wu, S., Xiong, Y., Cui, Y., Wu, H., Chen, C., Yuan, Y., ... Xue, C.J. (2024). Retrieval-augmented generation for natural language processing: A survey. *arXiv preprint arXiv:2407.13193*.

Yu, H., Gan, A., Zhang, K., Tong, S., Liu, Q., & Liu, Z. (2024, August). Evaluation of retrieval-augmented generation: A survey. In *CCF Conference on Big Data* (pp. 102–120). Singapore: Springer Nature Singapore.

Yu, W. (2022). Retrieval-augmented generation across heterogeneous knowledge. In *Proceedings of the NAACL 2022 Student Research Workshop* (pp. 52–58).