International Journal on Engineering Artificial Intelligence Management, Decision Support, and Policies
Vol. 03, Iss. 01, S. No. 02, pp. 09--21, March 2026
ISSN (Online): 3048-8788

Received: 02 Feb 2026, Accepted: 22 Feb 2026, Published: 08 April 2026
Digital Object Identifier: https://doi.org/10.63503/j.ijaimd.2026.233

Review Article

Retrieval-Augmented Generation (RAG) for Large Language Models: A Comprehensive Survey

Tanay Chowdhury

Data Science Lead, Gen AI Center of Innovation, Amazon Web Services,

Seattle, USA

tanayz@outlook.com

*Corresponding author: Tanay Chowdhury, tanayz@outlook.com

ABSTRACT

The Retrieval-Augmented Generation (RAG) paradigm has been proposed as a potent concept that would increase the factual accuracy, reliability, and adaptability of Large Language Models (LLMs) by incorporating external information retrieval and text generation. In comparison to an independent LLM, which leverages solely parametric knowledge, RAG dynamically accesses non-parametric knowledge sources during inference that are up-to-date and are founded on the evidence that was accessed to generate the response. This paper summarizes the foundations of RAG, its architecture, major components (retriever and generator) and an indexing-retrieval-generation methodology. It critically examines retrieval strategies in which sparse, dense and hybrid retrieval are examined with their efficiency trade-offs, semantic insight, interpretability and application to domain-specific tasks including healthcare. The paper further compares RAG with fine-tuning and brings out the differences in updating knowledge, customization and hallucination reduction. Finally, the augmentation strategies, the higher-level techniques iterative, recursive and adaptive retrieval are discussed to solve the complex, multi-step reasoning tasks. In summary, the paper reveals that RAG is a scalable and powerful AI-based solution that is knowledge-intensive.

Keywords: *Retrieval-Augmented Generation (RAG), Large Language Models (LLMs), Knowledge-Augmented NLP, Hallucination Reduction, Adaptive Retrieval.*

1. Introduction

LLMs have radically revolutionized NLP, with impressive text-generating and text-reading capabilities. Nevertheless, the insights that are contained in these models are necessarily limited to the data in which they were trained [1]. This drawback leads to a knowledge cut-off, that is, LLMs cannot read information that was published post-training, or can read proprietary information or domain-specific nuances that are essential to enterprise applications.

As an example, a general-purpose LLM may not know about any new internal policy changes or new market changes that happened since the time of training. One of the prevailing problems following this stagnant reservoir of knowledge is the phenomenon of hallucinations [2]. LLMs are capable of producing answers that appear plausible, but that are factual or misleading. Such inaccuracies are due to gaps or biases in their training data or by the model itself creating information where there is uncertainty.

RAG is a new paradigm that can be used to address these constraints, as it integrates external information retrieval techniques with generative language models. RAG systems do not rely solely on parametric knowledge as model weights, but rather dynamically read the relevant documents in external knowledge stores, which could be a database, a vector store or a web corpus, and condition the generation process based on the evidence that has been retrieved [3]. This hybrid design is quite efficient in the aspect of

factual accuracy, breadth of knowledge, interpretability and flexibility, and reduction of the model retraining frequency.

The nature of industrial labor is radically changing as the factory environment is becoming more digitalized, with employees interacting with smart systems. Large Language Models (LLMs) do much to improve the capacity to comprehend complex technical information through the use of natural language conversation [4]. The LLM-based chat assistants are user interfaces between human beings and technical infrastructures that assist operators with view to troubleshooting, optimization of processes and knowledge transfer.

In addition to providing dynamic access to external information, RAG systems are a major advancement in NLP and AI that makes use of LLMs' capabilities. This new technique is what makes it possible to generate replies that are more accurate, relevant and up-to-date across many applications [5], [6]. RAG research is important because it can potentially radically change AI applications. These systems increase the accuracy and relevance of text, enhance the ability of AI to work with complex, knowledge-intensive tasks, and provide opportunities to constantly update knowledge without re-training the models.

The following is the outline of the current paper. The introduction to the concepts of retrieval-augmented generation is found in Section 2. Section 3 explores Retrieval Techniques in Rag. The emphasis of Section 4 is on Augmentation Process in Rag for LLM. The literature related to the study is reviewed in Section 5. Lastly, Section 6 outlines future research directions, with applications in pressure piping systems and structural stability analysis highlighted in particular.

2. Fundamentals of Retrieval-Augmented Generation

RAG is a new and advanced AI paradigm, which relies on generative models together with Information Retrieval (IR) to improve a reliability and accuracy of responses (Lewis et al., 2020). The traditional IR systems can be used to locate the appropriate material but cannot generate original material. Conversely, the LLMs are fluid writers, though they make use of already learned information. RAG closes this gap by first retrieving the necessary documents through other databases before using them to guide text synthesis. Three primary stages of RAG process are as follows: indexing, which is the process of storing documents in a database; retrieval, which is the process of retrieving the relevant materials based on user queries; and generation, which is the process of synthesizing replies by using the input prompt, utilizing the user question and the relevant documents.

2.1 Architecture of Retrieval-Augmented Generation

An interesting invention is the architectural idea referred to as RAG. This model is more accurate and context-rich because it combines the benefits of generation-based and retrieval-based methods to generate AI responses. Fig. 1 identifies the elements and procedure of RAG architectural model.

Workflow of the RAG Architecture Model is as follows**:**

Question Input: The customer makes a query to the system. This initiates the process whereby the query is passed on to the framework.

Semantic Search: The framework queries the vector database using semantic search methods. Based on the submitted query, this search returns pertinent contextual information.

Contextual Data Utilization: The obtained information is then utilized to generate a prompt. This question is especially designed to help the LLM generate an answer that is both relevant and insightful.

Response Generation by LLM: A response is generated by the LLM once it processes the prompt [7]. The LLM is able to provide excellent results since it has been trained extensively on large datasets.

Post-Processing: The resulting response is processed after to ensure that it is clear, coherent, and relevant. This stage can involve the refinement of the wording, correction of errors, and the overall quality of the answer.

Response Delivery: The customer gets the final polished solution which gives them the information they needed in a way that is easy to understand and efficient.

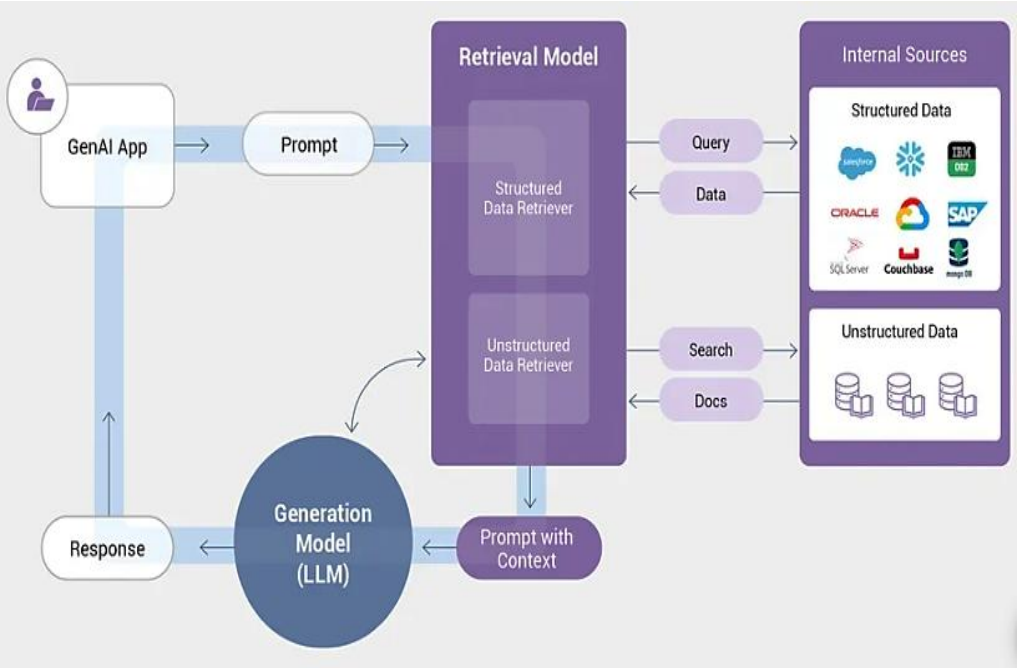

Fig. 1. RAG architecture.

2.2. Components of RAG systems: Retriever and Generator

The RAG is a hybrid method that provides better, more context-driven results by combining the LLM's generating capabilities with information retrieval approaches. RAG models consist of two primary parts:

Retriever: This system determines and retrieves applicable documents, passages or snippets of information in a large corpus or database using the search query. The retrieval process can be performed using various techniques:

Dense Retrieval: Embeds neural representations to locate similar semantic meaning documents. DPR (Dense Passage Retrieval) is a good example as it uses dense vector representations to match queries with relevant passages.

Sparse Retrieval: Utilizes the time-tested associated algorithms such as TFIDF and Best Matching 25 (BM25) to locate the documents that have shared keywords.

Generator: A response is generated by this component using the collected information. The generator, often an LLM, takes the input question and the retrieved documents and uses them to produce a more precise and contextually improved result.

2.3 Types of retrievers: Sparse vs Dense retrieval

Dense retrieval uses neural embeddings to find documents with similar semantic meanings. It involves the following techniques:

1. Dense Passage Retrieval (DPR)

DPR makes use of queries and passages represented by dense vectors [8]. A neural network encodes queries and passages into high-dimensional vectors, and relevant passages are retrieved based on vector similarity measures (e.g., cosine similarity).

2. Neural Retrieval Models

These models, such as Sentence-BERT and USE (Universal Sentence Encoder), generate embeddings for sentences or passages, allowing for efficient retrieval based on semantic similarity.

3. Sparse Retrieval

Sparse retrieval applies the traditional method of searching using keywords.

Key Methods Include

4. TermFrequency-InverseDocumentFrequency (TF-IDF):

This method evaluates the importance of words in documents relative to a corpus, enabling retrieval based on keyword matching.

5. BM25

A more advanced probabilistic search model that builds upon TF-IDF, saturation of frequency of words in a document, and normalization of document length by document length, thus suitable in keyword-based searches.

3. Retrieval Techniques in RAG

RAG addresses critical shortcomings of standalone LLMs by incorporating external, non-parametric knowledge via a retrieval component. The foundation of generation on retrieved documents makes RAG more factual and enhances the coverage of knowledge as well as allows dynamic access to dynamic information. The success of RAG system strongly relies on the type of retrieval strategy that is used because the quality and relevancy of retrieved information directly affect the output of the system.

3.1 Sparse Retrieval

Sparse retrieval algorithms are based on lexical correspondence between the query that is entered and the documents in the corpus. Methods like the TF-IDF and BM25 are used to represent documents with the help of sparse vectors depending on the frequency of appearance of words. TF-IDF highlights the words appearing often in a document and seldom in the corpus, and BM25 extends this methodology and incorporates the frequency saturation of terms and document length normalization [9]. The sparse retrieval algorithms are insightful, interpretable, and do not need training at all, hence are applicable to large-scale and low-latency applications. They, however, depend on the overlap of words, which are precise or close enough to the words to be considered, which limits their capacity to represent semantic similarity. This weakness is more noticeable in other fields like healthcare and clinical text analysis, where the use of abbreviations, synonyms, and heterogeneous terms is prevalent.

3.2 Dense Retrieval

Dense Retrieval (DPR, Sentence Transformers) consists of sentences and phrases that are densely combined to retrieve pertinent information. Dense Retrieval (DPR, Sentence Transformers) comprises sentences and phrases that are densely packed together in order to seek out the relevant information [10]. The dense retrieval methods map queries and documents to a common continuous embedding space by neural encoders, allowing semantically similar matches even without lexical overlap [11]. Other models like Dense Passage Retrieval (DPR) and Sentence Transformers are trained with

contrastive learning goals to learn contextual and semantic relationships between text fragments. Dense retrieval particularly works well with paraphrasing queries, synonyms, and context-rich information needs. Dense retrievers are useful in medical and knowledge-intensive fields to enhance recall and relevance through matching semantically related information despite surface-wording differences [12]. Nevertheless, they are more costly in terms of computation since generation and similarity of vectors are encoded in these methods and labeled or weakly supervised training data is required to train them, and are less interpretable than sparse retrieval methods, as shown in Table 1 of Comparison of Sparse and Dense Retrieval Methods below.

Table 1. Comparison of Sparse and Dense Retrieval Methods

Key Points	**Sparse Retrieval**	**Dense Retrieval**
Retrieval Mechanism	Lexical token overlap	Learned semantic similarity
Input Representation	Bag-of-Words (BoW) vectors	Contextual neural embeddings
Similarity Metric	TF-IDF, BM25 (exact or weighted match)	Dot product or cosine similarity
Training Requirement	No training required	Requires supervised or weakly supervised training
Speed	Fast (index-based lookup)	Slower (approximate nearest-neighbor search)
Semantic Matching	Low (sensitive to term variation)	High (captures semantic context)
Memory Usage	Low (compact inverted index)	High (large vector storage)
Interpretability	High (term-level explanation)	Low (black-box embeddings)
Common Tools	TF-IDF, BM25, Elasticsearch	DPR, Sentence Transformers
Suitability for Healthcare	Effective for structured queries and known terminology	Effective for unstructured clinical text, synonyms, and abbreviations

3.3 Hybrid Retrieval Approaches

Hybrid retrieval approaches combine sparse and dense retrieval techniques to use the capabilities of each in terms of spelling and semantic interpretation. Sparse retrievers are typically used to quickly reduce the number of candidate documents, and dense retrieval or fusion-based ranking is applied to make the results more relevant. The hybrid strategies provide a sensible trade-off between efficiency and accuracy, which is especially appealing to the enterprise and domain-specific RAG systems [13], [14]. Hybrid retrieval supports healthcare and clinical settings by addressing terminological variation while maintaining acceptable latency and scalability, thereby improving the quality of responses to diverse query types.

3.4 RAG vs Fine-Tuning

LLMs have demonstrated the ability to be fine-tuned to better fit task requirements and data characteristics, thereby improving domain adaptation, especially in on-premises applications. Fine-tuning enables the use of domain knowledge, customization to structured data formats, and control over the style of output (often, publicly available datasets (e.g., Hugging Face) can be a good starting point).

RAG systems' fine-tuning can be optimized jointly with retrieval components by matching retriever and generator preferences using contrastive learning, feedback-based reinforcement learning, or knowledge distillation with more powerful models, such as RA-DIT, to improve performance by integrating retriever-generator scoring, enabling more consistent, grounded generation when access to large, proprietary models is not readily available, as shown in Table 2. The comparison of RAG vs Fine Tuning is below.

Table 2. Comparison of RAG vs Fine-Tuning

Aspect	**Retrieval-Augmented Generation (RAG)**	**Fine-Tuning (FT)**
Core Idea	Augments LLMs with external knowledge retrieved at inference time	Updates model parameters by training on task- or domain-specific data
External Knowledge Requirement	High – relies on external knowledge sources (documents, databases, APIs)	Implicit – knowledge is embedded into model weights during training
Model Adaptation Requirement	No parameter updates to the base model	High – requires parameter updates and retraining
Knowledge Update Mechanism	Dynamic; supports real-time knowledge updates	Static; requires retraining to incorporate new knowledge
Customization Capability	Moderate – behavior influenced by retrieved context	High – enables deep customization of model behavior, style, and structure
Latency	Higher due to retrieval and re-ranking steps	Low at inference time
Computational Cost	Moderate (embedding, retrieval, indexing overhead)	High (dataset preparation, training, and compute resources)
Interpretability	High – retrieved documents provide transparent evidence	Low – knowledge embedded in opaque model weights
Hallucination Reduction	Strong – grounded generation using retrieved evidence	Moderate – reduced for known patterns, weaker for unseen data
Handling New / Unseen Knowledge	Excellent – can retrieve entirely new knowledge	Weak – LLMs struggle to learn new facts via unsupervised fine-tuning

Suitability	Knowledge-intensive and dynamic information retrieval tasks	Tasks requiring replication of specific styles, formats, or behaviors
Ethical & Privacy Concerns	Higher – concerns related to data sources and retrieval	Moderate – depends on training data provenance
Representative Analogy	Providing a tailored textbook during problem-solving	A student internalizing knowledge through prolonged study
Empirical Performance	Consistently outperforms FT on knowledge-intensive tasks	Limited gains for factual knowledge learning

4. Augmentation Process in RAG for LLM

The common pattern in the field of RAG with LLMs is that there is a single (one-time) retrieval step and then the text is generated, this may be inefficient and not sufficient to handle complex tasks that involve multi-step reasoning, as it gives a narrow contextual scope. The parametric knowledge of Large Language Models is regularly not sufficient to meet domain-specific and dynamically evolving information demands, although they have powerful generative and reasoning processes. As a result, the optimization and iterative refinement of the retrieval augmentation process to improve the integration of retrieved evidence within the reasoning ability of LLMs has been the subject of a number of studies as summarized in Fig. 2.

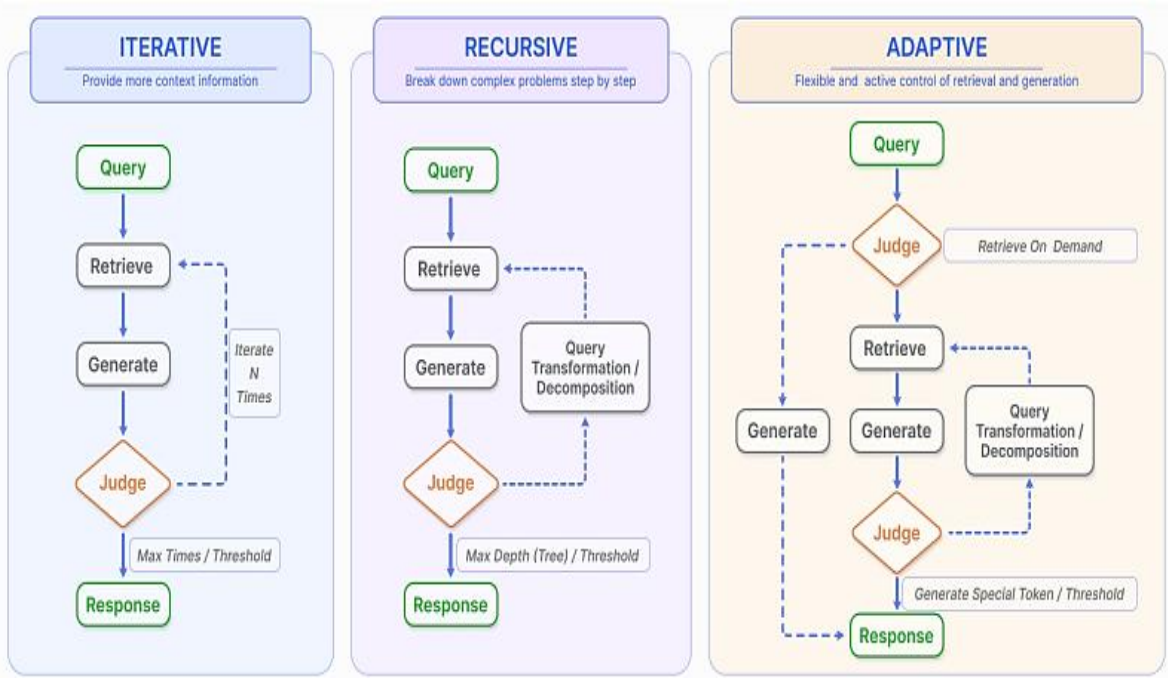

Fig. 2. Process of Augmentation in RAG.

4.1 Iterative Retrieval

Iterative retrieval uses the initial query and the previously prepared text to continuously search the database in order to construct a more Comprehensive Knowledge base for LLMs . This method has

been shown to enhance the resilience of future answer generation by providing additional contextual references throughout several retrieval cycles [15]. Nevertheless, it might be impacted by the buildup of superfluous data and semantic gaps. "Retrieval-enhanced generation" and "generation-enhanced retrieval" work together in a complementary fashion in ITERRETGEN, a synergistic strategy for data reproduction tasks. The method aids better responses in following rounds by retrieving relevant knowledge utilizing the information needed to finish the input job as a contextual foundation.

4.2 Recursive Retrieval

It is common practice in NLP and information retrieval to use recursive retrieval in order to refine and increase the breadth of search results. Search terms are refined repeatedly depending on results acquired from prior searches. Improving the search experience is the goal of Recursive Retrieval, which uses a feedback loop to progressively converge on the most relevant information. IRCoT improves the CoT using the outcomes of the retrieval process guided by chain-of-thought. In order to optimize the confusing sections of the Query, ToC develops a clarification tree. In complicated search situations where the user's demands are not immediately apparent or when the information being sought is very specialized or nuanced, it may be especially helpful. The recursive structure of the procedure allows for ongoing learning and modification to meet the needs of the user, which often leads to higher satisfaction with search results [16]. Combining recursive and multi-hop retrieval methods allows us to tackle unique data circumstances. A structured index is used in recursive retrieval to process and retrieve data hierarchically. This procedure might entail summarizing portions of a document or long PDF before retrieving the rest of the material based on this summary [17]. The recursive process is shown by a secondary retrieval inside the document that refines the search. Multi-hop retrieval, in its turn, is aimed at retrieving the related information in graph-structured data sources by getting deeper.

4.3 Adaptive Retrieval

Adaptive retrieval methods like Flare and Self-RAG enhance the RAG framework in the sense that they enable the LLMs to take proactive decisions concerning the most appropriate time and content they should retrieve, which makes the information retrieved more efficient and more relevant. These techniques are an example of the growing trend of LLMs using active judgement in their operations; this is also true of model agents like AutoVPN, Tool former, and Graph Tool former. For example, Graph-Tool former employs a three-step retrieval process: LLMs actively utilize retrievers, implement Self-Ask strategies, and start search queries with few-shot prompts [18]. This proactive approach enables the LLMs to determine when to find requisite information just like the agent makes use of tools. When retrieving the text, the generator performs a beam search at the fragment level on a series of paragraphs to produce the most coherent sequence. The design of Self-RAG does not require further classifiers or use of the Natural Language Inference (NLI) models, thereby simplifying the decision-making process of whether to involve retrieval mechanisms or enhancing the autonomous judgment abilities of the model in producing correct responses.

5. Literature Review

The literature on Retrieval-Augmented Generation (RAG) demonstrates a steady evolution from foundational architectures that combine parametric and non-parametric memory to more advanced, domain-adaptive, and context-aware retrieval mechanisms. Recent studies emphasize joint retriever–generator optimization, topic-aware retrieval, and efficient knowledge integration to improve factual grounding and scalability. Table III summarizes key contributions, methodologies, and findings from representative works in this area.

Yu et al. (2022) The RAG approaches are becoming more visible in the NLP community as a whole due to their state-of-the-art performance on several downstream NLP tasks. Amazing benefits, such as

minimal training cost, robust scalability, and quick knowledge acquisition, are offered by RAG approaches in comparison to traditional pre-trained generation models. A lot of research has focused on knowledge-intensive NLP applications, such as conversation systems and open-domain quality assurance, and has used existing RAG models to extract unstructured material from Wikipedia. Identify and describe the existing challenges to knowledge retrieval from a homogenous corpus sourced from a single source [19].

Ahn et al. (2022) suggested retrieval-augmented models may provide a response using a number of documents; however, they completely disregard the provided subject and rely only on the conversational context. Therefore, they provide a new model for retrieval-augmented response generation that takes into account the subject and local context of a discussion to retrieve a variety of documents that may be used to generate a response that is anchored in knowledge. In order to provide different representations, their model first takes into account both the tokens before to the answer and the subject words that were taken from the whole discussion. After that, it compares the two sets of conversation representations with the matching set obtained from the document after choosing the representations of the first N tokens and one representation of each keyword from the conversation and document encoders. The model is trained using a novel data-weighting strategy that encourages it to produce knowledge-based replies that are not reliant on ground truth data [20].

Siriwardhana et al.(2022) propose RAG-end2end, a RAG extender that can adapt to a domain-specific Knowledgebase by updating all elements of the External Knowledgebase during training. Furthermore, it provides an additional training signal to add more data particular to the topic. In response to this supplementary signal, RAG-end2end retrieves relevant information from the External Knowledgebase and uses it to reconstruct a specific sentence. In contrast to RAG, their innovative RAG-end2end trains the generator and retriever in tandem for the last QA assignment and domain adaptation. Experiments on datasets by the COVID-19, news, and Conversational Domains show significant performance improvements over the original RAG model [21]

Singh and Mahmood et al. (2021) models created for NLP have shown impressive potential in several semantic and linguistic domains, such as Text Classification, machine translation, Cognitive Conversation Systems, NLU-based IR, NLP, and NLU. The ground-breaking Transformer architecture is largely responsible for this achievement; it gave rise to designs like BERT and GPT (I, II, III), among others. High computational expenses are associated with these large-size models, notwithstanding their remarkable results. Consequently, a number of recent NLP frameworks have reduced model sizes while maintaining almost identical performance to their forerunners via the use of techniques such as Knowledge Distillation, Transfer Learning, pruning, and quantization [22].

Lewis et al. (2020) explores a flexible method for optimizing RAG models that create languages using pre-trained parametric and non-parametric memory. It provides RAG models with a Dense Vector index of Wikipedia for Non-Parametric Memory, retrieved via a Pre-Trained Neural Retriever, and employs a pre-trained seq2seq model for Parametric Memory. It compares and contrasts two RAG formulations: one that doesn't permit token-specific passage modifications and the other that utilizes the same retrieved passages in the generated sequence at all times. It evaluates their models in comparison to parametric seq2seq models and task-specific retrieve-and-extract architectures, assessing their performance across multiple knowledge-intensive NLP tasks, and demonstrates that they surpass the advance results in three open-domain Qatasks [23].

Ambati and Chhadia (2019) incorporating information from other sources may enhance LLM performance in this matter. They want to investigate and evaluate different approaches to this improvement as they are often considered in isolation. A domain-adapted BERT base model is the focus of their investigation into RAG using generated knowledge graphs. It also looks at the Know BERT

architecture, a different method that directly incorporates knowledge into a language model. Their baseline is an unsupported knowledge graph-based domain-adapted BERT large model. Acc, prec, rec, F1score, confusion, and a factual memory test are the assessment metrics they use. They find that, on the whole, big language models benefit from information injection, which leads to increased accuracy [24].

Table 3. Retrieval-Augmented Generation (RAG) for Large Language Models.

Authors	**Focus Area**	**Objectives**	**Approach**	**Key Findings**	**Future Work**
Yu et al. (2022)	RAG with homogeneous corpora	When obtaining information from a single-source, homogenous corpus, identify the constraints of RAG.	Analysis of existing RAG models primarily retrieving unstructured text from Wikipedia	Highlighted challenges in knowledge coverage, retrieval bias, and limited adaptability when relying on single-source corpora	Extend RAG to heterogeneous, multi-source knowledge bases and structured data
Ahn et al. (2022)	Topic-aware retrieval-augmented dialogue generation	Improve knowledge-grounded response generation by considering both topic and local context	Topic- and context-aware document retrieval with multi-representation matching and data-weighted training	Achieved more relevant and knowledge-grounded responses without requiring ground-truth knowledge annotations	Incorporate richer discourse modeling and multi-turn reasoning
Siriwardhana et al. (2022)	End-to-end domain-adaptive RAG	Enable RAG to adapt to domain-specific knowledge bases through joint training	Proposed RAG-end2end with joint retriever–generator training and auxiliary reconstruction signal	Demonstrated significant performance gains across COVID-19, news, and conversational datasets	Explore scalability to larger knowledge bases and continual domain adaptation
Singh & Mahmood (2021)	Efficient NLP model design	Reduce computational cost of large transformer-based models while	Transfer learning, pruning, quantization, and knowledge distillation	Showed that compact models can achieve near-parity performance with large LLMs	Combine efficiency techniques with retrieval-augmented and knowledge-aware models

		maintaining performance			
Lewis et al. (2020)	Foundational RAG architecture	Develop a general-purpose framework combining parametric and non-parametric memory	Introduced RAG-Sequence and RAG-Token with dense retrieval over Wikipedia	Achieved state-of-the-art results on multiple open-domain QA benchmarks	Extend RAG to new tasks, domains, and dynamic knowledge sources
Ambati & Chhadia (2019)	Knowledge injection in language models	Evaluate the impact of external knowledge integration on NLP performance	Compared RAG with knowledge graphs and KnowBERT on a domain-adapted BERT model	Found that injecting structured knowledge improves accuracy and factual recall	Improve integration of structured and unstructured knowledge sources

6. Conclusion and Future Work

RAG is an important development in the history of LLM in that it can successfully integrate information retrieval and generative capacity to address the shortcomings of parametric knowledge. RAG makes decisions more factual by basing them on documents outside of memory and also minimizes hallucinations and allows people to update their knowledge dynamically without expensive retraining of the model. This paper has discussed the basic structure of the RAG, its basic elements and various retrieval mechanisms such as the sparse, dense and hybrid mechanisms and their strengths and limitations. Moreover, more complex augmentation methods including iterative, recursive and adaptive retrieval were mentioned in solving multi-step, complex reasoning problems. As the comparison with fine-tuning showed, RAG is especially adapted to the field of knowledge-intensive and rapidly changing spheres. In general, RAG provides a scalable, interpretable, and efficient model of constructing stable AI systems that require a proper and situation-specific language comprehension.

Future studies have the potential to enhance retrieval efficiency, decrease computational overhead and increase the interpretability of dense retrievers. The investigation of multi-modal RAG, more robust retriever-generator co-training, and domain-specific adaptive retrieval approaches will extend the applicability of RAG to applications where safety is a crucial factor.

Funding source

None.

Conflict of Interest

None.

References

[1] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, “Large language models are zero-shot reasoners,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 35, 2022.

[2] M. Chen et al., “Evaluating large language models trained on code,” arXiv preprint arXiv:2107.03374, 2021.

[3] F. J. Teixeira, A. A. N. Castro, and A. C. Sant’Ana, “Investigating the origin of the raw material of rag paper by Raman spectroscopy,” Vib. Spectrosc., vol. 98, pp. 119–122, Sep. 2018, doi: 10.1016/j.vibspec.2018.08.003.

[4] E. Kristiani, V. K. Verma, and C.-T. Yang, “Deploying LLM transformer on edge computing devices: A survey of strategies, challenges, and future directions,” AI, vol. 7, no. 1, p. 15, Jan. 2022, doi: 10.3390/ai7010015.

[5] D. Li et al., “Large language models with controllable working memory,” in Findings of the Association for Computational Linguistics (ACL), pp. 1774–1793, 2023, doi: 10.18653/v1/2023.findings-acl.112.

[6] P.-M. Wong and C.-K. Chui, “Cognitive engine for augmented human decision-making in manufacturing process control,” J. Manuf. Syst., vol. 65, pp. 115–129, Oct. 2022, doi: 10.1016/j.jmsy.2022.09.007.

[7] S. Siriwardhana, R. Weerasekera, E. Wen, and S. Nanayakkara, “Fine-tune the entire RAG architecture (including DPR retriever) for question-answering,” arXiv preprint, Jun. 2021.

[8] P. Pathak, A. Shrivastava, and S. Gupta, “A survey on various security issues in delay tolerant networks,” J. Adv. Shell Program., vol. 2, no. 2, pp. 12–18, 2015.

[9] R. Saxena, S. A. Pushkala, and R. Carvalho, “Systems and methods for rapid processing of file data,” U.S. Patent 9,594,817, Mar. 2017.

[10] J. Chen, R. Zhang, J. Guo, Y. Fan, and X. Cheng, “GERE: Generative evidence retrieval for fact verification,” in Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval (SIGIR), 2022, pp. 2184–2189, doi: 10.1145/3477495.3531827.

[11] K. Murugandi and R. Seetharaman, “A study of supplier relationship management in global procurement: Balancing cost efficiency and ethical sourcing practices,” Int. J. Adv. Res. Sci. Commun. Technol., vol. 2, no. 1, pp. 724–733, 2022, doi: 10.48175/IJARSCT-7744B.

[12] W. Lin and B. Byrne, “Retrieval augmented visual question answering with outside knowledge,” in Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP), 2022, pp. 11238–11254, doi: 10.18653/v1/2022.emnlp-main.772.

[13] C. Zhang, Y. Lai, Y. Feng, and D. Zhao, “A review of deep learning in question answering over knowledge bases,” AI Open, vol. 2, pp. 205–215, 2021, doi: 10.1016/j.aiopen.2021.12.001.

[14] S. Gupta, N. Agrawal, and S. Gupta, “A review on search engine optimization: Basics,” Int. J. Hybrid Inf. Technol., vol. 9, no. 5, pp. 381–390, May 2016, doi: 10.14257/ijhit.2016.9.5.32.

[15] P. S. H. Lewis, “Improving neural question answering with retrieval and generation,” Ph.D. dissertation, Univ. College London (UCL), 2022.

[16] V. Thakaran, “A comparative study of piping stress analysis methods with different tools, techniques, and best practices,” Int. J. Adv. Res. Sci. Commun. Technol., vol. 2, no. 1, pp. 675–684, Oct. 2022, doi: 10.48175/IJARSCT-7868D.

[17] P. Loft, Y. He, I. Yevseyeva, and I. Wagner, “CAESAR8: An agile enterprise architecture approach to managing information security risks,” Comput. Secur., vol. 122, p. 102877, Nov. 2022, doi: 10.1016/j.cose.2022.102877.

[18] J. Kachhia, A. Patel, A. Vala, R. Patel, and K. Mahant, “Logarithmic slots antennas using substrate integrated waveguide,” Int. J. Antennas Propag., vol. 2015, 2015, doi: 10.1155/2015/629797.

[19] W. Yu, “Retrieval-augmented generation across heterogeneous knowledge,” in Proc. NAACL-HLT Student Res. Workshop, 2022, doi: 10.18653/v1/2022.naacl-srw.7.

[20] Y. Ahn, S.-G. Lee, J. Shim, and J. Park, “Retrieval-augmented response generation for knowledge-grounded conversation in the wild,” IEEE Access, vol. 10, pp. 131374–131385, 2022, doi: 10.1109/ACCESS.2022.3228964.

[21] S. Siriwardhana, R. Weerasekera, E. Wen, T. Kaluarachchi, R. Rana, and S. Nanayakkara, "Improving the domain adaptation of retrieval-augmented generation models for open-domain question answering," Trans. Assoc. Comput. Linguist., vol. 11, pp. 1–17, 2023, doi: 10.1162/tacl_a_00530.

[22] S. Singh and A. Mahmood, "The NLP cookbook: Modern recipes for transformer-based deep learning architectures," IEEE Access, vol. 9, pp. 68675–68702, 2021, doi: 10.1109/ACCESS.2021.3077350.

[23] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020.

[24] Ambati and N. Chhadia, "Knowledge-enhanced language models: A comparative study of RAG and embedding methods," 2019.