RAG Powered LLMs for QA: Evolution, Challenges, Applications, and Future Directions

Hafiz Muhammad Ali Zeeshan , Muhammad Faizan, Usman Zia, Abdullah Gohar
National University of Sciences and Technology (NUST) Islamabad, Pakistan
hzeeshan.phdee24seecs@seecs.edu.pk, mfaizan.rime22smme@student.nust.edu.pk,
usman.zia@sines.nust.edu.pk, abdullahgohar01@gmail.com

2025 International Conference on Communication Technologies (ComTech) | 979-8-3315-1533-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/ComTech65062.2025.11034531

***Abstract*—Large language models (LLMs) are evolving to excel in challenging tasks such as text generation, mathematical reasoning, code generation, question answering, text summarization, etc. However, the responses generated by LLMs are prone to hallucinations, out-of-the-date knowledge, and non-transparent and untraceable reasoning. Retrieval augmented generation (RAG) addresses these shortcomings by incorporating external knowledge in the LLM prompt. RAG combines parametric knowledge of LLM with non-parametric knowledge from external databases by efficient retrieval techniques. This comprehensive review paper provides a detailed overview of the significance and emergence of RAG. Moreover, the building blocks of RAG are thoroughly discussed along with the evolution, challenges, and current research trends of deploying RAG-based applications. Finally, we explore the RAG performance enhancement strategies while also underlining the potential future research directions.**

***Index Terms*—Retrieval Augmented Generation (RAG), Large Language Models (LLM), Question Answering (QA), RAG Applications and Challenges**

I. INTRODUCTION

Large Language Models (LLMs) have shown impressive performance in various natural language processing tasks such as text generation, context understanding, and context-aware coherent response. Nevertheless, LLMs have several shortcomings, one of which is the difficulties in solving problems that rely on a large amount of domain-specific or expert knowledge. These difficulties frequently appear as "hallucinations" in which models produce content – that sounds accurate but is inaccurate, specifically when addressing queries for which the relevance of their training data is more specific or needs current information. A broader class of models that can handle these scenarios consists of Retrieval Augmented Generation (RAG) approaches. RAG overcomes the hallucination problem by grounding responses generated by an LLM in verifiable external sources, improving accuracy and reliability of LLM-generated content, and integration of external knowledge bases via semantic similarity calculations. [1].

RAG combines the generative capability of LLMs with retrieval mechanisms over external data repositories that are simultaneously virtually unlimited and highly dynamic. Such a hybrid approach not only avoids generating inaccurate information but also enables on-the-fly knowledge updating, making RAG in particular ideal for real-world applications that require current first-hand information [2]. In this way, RAG has been widely adopted in different applications like Chatbot and Automatic Customer Support, where it provides useful and accurate context during generation to largely complement LLM performance [3].

Despite many exciting breakthroughs, the RAG field has not been comprehensively synthesized to chart its broader development and future directions. We fill this gap with a survey that provides an in-depth look at LLMs based on RAG for question answering (QA). This paper reviews the literature on RAG and brings to light the RAG paradigms, from the original Naive RAG to two more sophisticated frameworks, Advanced RAG and Modular RAG. It also goes into the basics of RAG — retrieval, generation, and augmentation, and provides a review of the recent advances in each part [4].

The objective of this survey is to produce a high-quality and well-organized summary of the most recent RAG systems, more precisely, of end-to-end RAG systems that may be jointly trained with LLMs. To this end, this paper surveys key technical paradigms, research best practices, and evaluation frameworks to provide a broad foundation for research. Furthermore, it talks about the present difficulties they face and the directions in which future research should go to enable the capabilities of LLMs for QA and other knowledge-intensive tasks, given their implication to daily life [5].

In this paper, we provide a comprehensive review of the most recent LLM models in the RAG methods, along with how they are situated in the wider milieu of LLM. It also presents an overview of relevant technologies enabling the RAG process, such as retrieval, generation, and augmentation techniques. Moreover, the paper reviews the existing evaluation protocols for RAG, including different types of tasks, datasets, benchmarks, and techniques. It ends with some speculation on the future of RAG, a list of improvements for the existing problems, and some suggestions about integration with LLM fine-tuning approaches. This survey seeks to understand the genesis, use cases, and prospective significance of RAG-powered LLMs.

This paper makes the following main contributions:

- Frameworks for RAG systems.
- Reviews the current study and evaluation procedures for RAG, to ensure objective and accurate evaluation.
- Current problems and future research directions for better leveraging LLM capabilities in QA and other Knowledge-intensive task

- Discusses the future of RAG, both in terms of enhancements to fix some of its issues and in terms of its potential integrations, use cases, and a potential future role of RAG-powered LLMs.

The rest of the paper is organized as follows. Section II introduces the RAG core components. In Section III, we describe the evolution of RAG topologies. Section IV discusses the technical evaluations of RAG. Section V explores some RAG applications. Section VI focuses on key challenges and future directions in RAG. The paper is concluded in Section VII.

II. Core Components of RAG

Retrieval-Augmented Generation (RAG) has three main parts, Retrieval, Generation, and Augmentation. Combining these stages improves the performance and capabilities of Large Language Models (LLMs) with external knowledge.

A. Retrieval

In RAG, retrieval is especially important since successful retrieval implies that the information being input to LLM will be relevant and of high quality. The retrieval could be from unstructured, semi-structured, or structured data. The information retrieval could be categorized into three main categories, each of which is essential for the most relevant context extraction for the LLM prompt. These categories can be divided into pre-retrieval, retrieval, and post-retrieval. Next, each of these categories will be explored in depth.

1) Pre-Retrieval: Pre-retrieval is an important step for accurate and the most relevant data preparation before using a retriever. The process of pre-retriever begins with the indexing which enables fast and efficient context retriever. The specificity of indexing relies on the application. For instance, sentence-based indexing may be suitable for question answering while document-level indexing may be more appropriate for summarizing the concept or idea present in the document.

The traditional indexing method is based on inverted indexing, where each document is associated with a list of vocabulary. Therefore, keywords in the query are matched with the document based on the word's position and weight. However, such an approach is not effective for indexing the relevant semantic details to user queries. To address the limitation, retrieval techniques using dense vector embeddings using deep neural networks provide an alternative approach for accurately relating words and documents, enabling for more adaptive and accurate retriever. These dense vector-based indexing methods can be classified into three major types: graphs, locality-sensitive hashing (LSH) [6], and product quantization [7].

2) Retrieval: The category combines search and ranking. It focuses on selecting and assigning priority scores to each document to maximize the quality of generation. This stage combines search algorithms that extract content that is most relevant to the user query. Several works proposed novel approaches to advance search and ranking. For instance, Atlas and AAR [8] are used to enhance the relevance of retrieved-context to the user query. Atlas focuses on refining the retriever's ability to choose semantically relevant documents, in particular with low data scenarios, by using a few shot techniques such as perplexity distillation and attention distillation. On the other hand, AAR adjusts preferences to better align with the requirement posed by the LLM, improving the retriever generalizing across different tasks by training a small separate model.

3) Post-Retriever: The post-retriever technique focuses on improving the response quality by refining the retrieved documents. These are categorized under re-ranking and filtering.

The re-ranking process involves re-evaluating, scoring, and reordering the previously retrieved documents. Its purpose is to better emphasize documents most relevant to the query while reducing the significance of less relevant ones. This step often integrates additional metrics or external knowledge to improve accuracy. Pre-trained models with high accuracy but lower efficiency are particularly useful in this stage, given the small number of candidate documents involved [9].

B. Generation

The context window of LLM has a limited word limit, it can't handle long contextual prompts. Therefore, RAG chunks the context and feeds relevant context to the generator. In the generation stage, the retrieved information is further processed to generate the final coherent and relevant answers. Inputting all the content directly copied to the LLM is not a good idea because the content may be redundant but there's a mistake that it may ignore the middle, this is what is called "Lost in the middle" in [10]. Curating the context ranks and compresses documents discovered to make retrieved content more relevant and less noisy, methods like small language models (SLMs) identify and discard irrelevant tokens, rendering the text in a fashion that is more interpretable by LLMs [11].

C. Augmentation

The RAG augmentation reifies retrieval by enriching the retrieval process with supplementary mechanisms for complex multimodal, multi-relational, and multi-step reasoning tasks. Another example utilizing the system of iterative retrieval is alternating between retrieval and generation to expand the knowledge base [12]. Recursive retrieval, or querying recursively to refine the query, is called breaking down into subproblems, and search relevance feedback loop [13]. Adaptive retrieval where LLMs decide when and what to retrieve, using e.g., "reflection tokens" and confidence thresholds to gate retrieval processes is discussed in [14]. This adaptive system reduces cognitive time to respond which improves speed and with less rigor processes thinking through the well-rehearsed process.

III. RAG Topologies

RAG is a revolutionary innovation in the age of Large Language Models (LLMs) which better enables external knowledge sources to be accessed in the language generation process. Through generative RAG, LLMs can be further improved, and the performance and robustness concerning knowledge-heavy tasks and prone-to-hallucination tasks,

979-8-3315-1533-1/25/$31.00 ©2025 IEEE

by utilizing external databases for the most up-to-date and domain-specific knowledge.

Over time, RAG has developed in stages, each one with its robustness and shortcomings:

A. Naive RAG

The oldest approach, depicted in Figure 2(a), consists of a simple pipeline of indexing, retrieval, and generation. This method is a Retrieve-Read framework [15]. During indexing, we could clean various-style method data and convert it into plain text. The text is therefore divided into portions and transformed into vectors to carry out an efficient similarity search.

When it receives a query, the RAG system converts the query into a vector and then retrieves the k most similar chunks based on semantic similarity. These are mashed with the original question to produce a full LLM prompt. Although simple, Naive RAG has limitations, such as poor retrieval precision and hallucination/redundant content.

B. Advanced RAG

Advanced RAG extends the idea of Naive RAG. This is divided into 2 parts: one for Pre-retrieval and one for Post-retrieval. Before the retrieval phase, methods for refinement of the indexing such as sliding windows and metadata integration, and techniques for optimization of the query like query rewriting and query expansion are implemented. The whole illustration for Advanced RAG is shown in Figure 2(b).

C. Modular RAG

The newly developed Modular RAG is the advancement of all, meeting the much-demanded adaptability and flexibility. It consists of a search module built on generic inputs tailored to search engines, databases [16], and query languages generated by LLM approaches of equivalent energy. The original work borrowed the memory of the LLM for an infinite memory pool in the memory module, making the memory-to-text distribution more compatible via recursive self-improvement [17]. Predict module removes any redundancy and noise as it allows context generation directly with an LLM and on the other hand Task-specific training: Task Adapter module specializes RAG to multiple downstream tasks [18]. Modular RAG is illustrated in the Figure 2(c).

Furthermore, Modular RAG is highly adaptive as it can simply substitute or change modules to manage specific tasks, extending its usage to diverse tasks. Such innovations as the Rewrite-Retrieve-Read model, in which retrieval queries are further refined with a rewriting module and an LM-feedback mechanism [15], enhance task results. In the Generate-Read, traditional retrieval is replaced with LLM learned content [19], and in the Recite-Read, retrieval is based on pulling knowledge out of model weights [20].

IV. Evaluation of RAG

Owing to how newly RAG pipelines have been developed and because of how widespread their applications have become in the NLP community, evaluating RAG models is one of the key research topics in the LLM community. The primary goal of this Litmus test is to aid visual and performance enhancements for the use of RAG models. In particular, this section will present major downstream tasks of RAG, datasets used for comparison, and how to evaluate RAG systems.

A. Evaluation Targets

Here are the major goals of an assessment:

1) Retrieval Quality: To evaluate whether the context fetched by the retriever component is effective or not, one must evaluate the retrieval quality. To evaluate the performance of the RAG retrieval module, we use common metrics from the search engines, recommendation systems, and information retrieval domains. This can be done to calculate one of the following metrics: Hit Rate, MRR, NDCG, etc., [21].

2) Generation Quality: The evaluation of generation quality is based on the generator's ability to produce coherent and appropriate answers given the retrieved context. This evaluation can be separated based on the goals of the content: either unlabeled or labeled content. When the content is not labeled the evaluation includes how faithful, relevant, and non-harmful the answers are. An example for each case conversely, the focus here is on the accuracy of the information churning out of the model for the labeled content in English [22]. Both retrieval and generation quality can be evaluated either manually or automatically as discussed in [22]

B. Evaluation Aspects

The core quality scores and four abilities, which relate to the evaluation of the two central targets of a RAG model (retrieval and generation), are key metrics of current evaluation practices.

1) Quality Scores: Context Relevance, Answer Faithfulness, Answer Relevance; Furthermore, these quality scores assess the efficiency of the RAG model from various aspects of information retrieval, and generation [23].

2) Required Abilities: Four abilities indicative of the adaptivity capacity and efficiency of the RAG evaluation are noise-robust, negation rejection, information integration, and counterfactual robustness [24]. These abilities determine how the model fares against several adversities and complicated scenarios, and as a result affect the quality scores.

3) Evaluation Benchmarks and Tools: RAG is evaluated with a set of benchmarks and tools. These tools give numerical evaluation criteria, and output many RAG model goals that measure how well (but also how rubbery) the model works on several evaluation objectives. RGB, RECALL, and CRUD are top benchmarks for evaluating the core capabilities of RAG models [25]. Some of the recent automated benchmarks like RAGAS, ARES [26], and TruLens aggregate these quality scores concurrently using LLMs. These tools and benchmarks offer a robust foundation for systematically assessing various RAG models.

979-8-3315-1533-1/25/$31.00 ©2025 IEEE

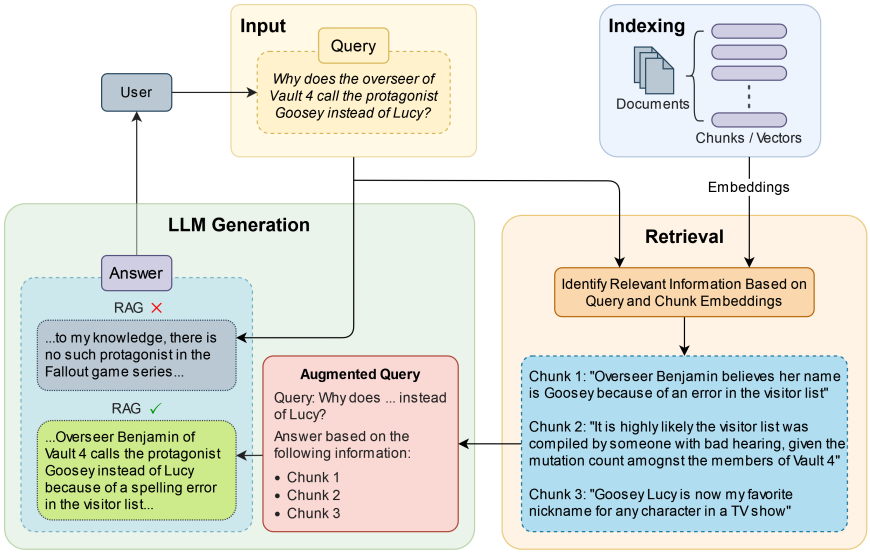

Fig. 1. Illustration of the usefulness of RAG in improving large language models (LLMs) by combining parameterized information from language models but at the same time, using non-parameterized information from external knowledge bases.

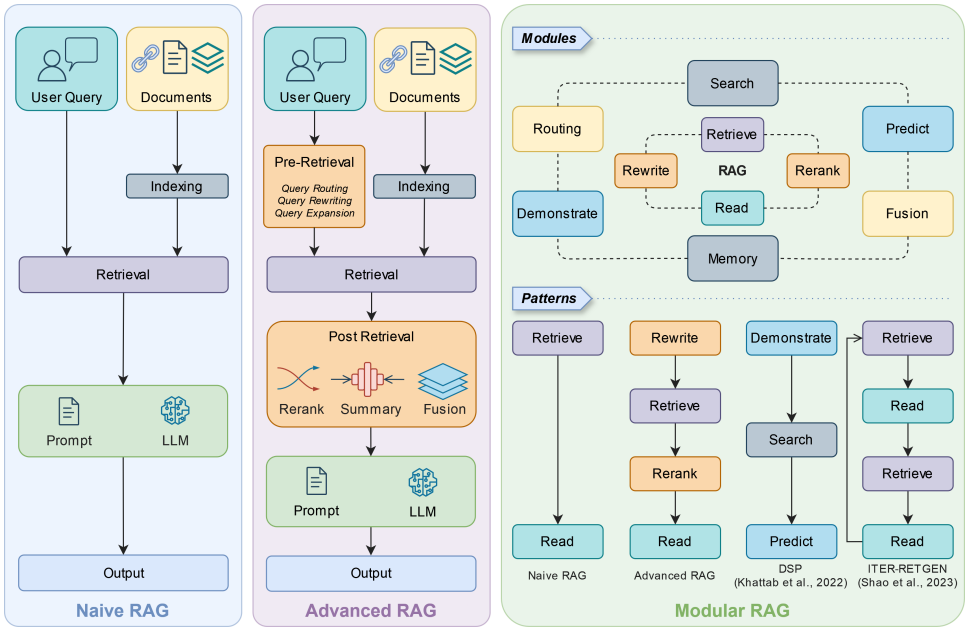

Fig. 2. A general overview of the various RAG development paradigms - naive (earliest), advanced, and modular (latest). (a) Naive RAG (b) Advanced RAG (c) Modular RAG

V. Applications of RAG

A. Single-hop and Multi-hop QA

RAG-based models excel at answering simple questions that can be resolved in a single retrieval step from an external knowledge source, enabling the QA system to provide accurate and truthful responses. For instance, in the Natural Questions (NQ) corpus, RAG systems can effectively extract the particular passages from the huge amount of data to respond to a given query [27]. For questions in multi-hop QA that need the information to be aggregated across documents, the RAG model is effective for collecting and integrating the data from multiple documents. This ability is important in datasets such as HotpotQA in which answering questions often requires understanding the connection of some information in one context to some other information in another [28].

B. Long-form QA

Long-form QA tasks require detailed responses that synthesize information from several places. Due to their ability to retrieve large amounts of information and generate specific responses, RAG models are well-suited for these purposes. For example, on ELI5 data RAG models provide detailed responses spanning over diverse facets of a topic, thus trying

979-8-3315-1533-1/25/$31.00 ©2025 IEEE

to maximize both depth and coverage [29]. The key step for RAG is to solve complex queries and combine various information sources in long-form QA, allowing for more in-depth explanations and discussions.

C. Domain-specific QA

RAG models excel at domain-specific QA, i.e., for questions related to specialized domains like medicine, law, technology, etc. BioASQ dataset: with access to domain-specific databases, RAG models can return accurate and contextually relevant answers compared to traditional approaches. Similarly, RAG models for medical QA datasets like COVID-QA can answer health-related questions by retrieving and synthesizing the latest research papers and guidelines [30]. This specialization means that the answers, in addition to being highly relevant, are also state-of-the-art in terms of the knowledge of the domain.

D. Multi-choice QA

RAG models are also useful for multi-choice QA tasks, where the system needs to pick a response from a set of available options, by facilitating the retrieval of contextually relevant facts, which could help in making the decision. Data sets, such as ARC (AI2 Reasoning Challenge), benefit from RAG's ability to background check answer choices against a large knowledge base hence making an even better guess of why an answer is correct [31]. In particular, the retrieval part in RAG makes sure that the generated context is relevant to let the model discriminate the right answer among the distractors.

E. Other Tasks

In addition to QA and dialogue generation, RAGs are used for many other tasks, such as reasoning, text generation, and text classification. RAG systems deliver retrieval-based logic inference performance in reasoning tasks by fetching relevant data to assist the process of reasoning. RAG models use the external information to generate a coherent and contextually consistent text, or in text classifier, to improve the classification rate by closing the gap of discrete information.

VI. Challenges and Future Directions

A. Robustness to Noise and Adversarial Inputs

A major issue in the implementation of RAG systems is that they are too vulnerable to noise and attacks. If the retrieved documents contain irrelevant or inaccurate information, then the responses may also be poor-quality answers. This is especially troublesome when the retrieval produces a set of documents that are in the ballpark of the query but don't see any meat. For instance, [32] showed that accuracy could rise by more than 30% by keeping irrelevant documents in the dataset, which indicates that this is a complex task to solve. For improvement of robustness, we can investigate how to create more effective retrieval mechanisms that can decide if the documents are relevant or not more accurately, perhaps by adding additional filtering or re-ranking stages into current retrieval mechanisms as pointed out [33].

B. Integration with Long-Context Models

This changes as new language models are released and due to how they evolve, their capacity to handle longer beholding of the context increases significantly. This indicates that modern LLMs are proficient enough to handle contexts larger than 130, 000 tokens. This ability leads to the natural question, if LLMs are freed up from context length, do we need RAG systems anymore? While these advances have helped, RAG is still required here to maintain efficiency and also to ensure the validity of the answers produced. When providing a context over 165,000 bytes in a single request the inference times are significantly slower whereas chunked retrieval allows for on-demand input, speeding up the entire process. Not only this, but RAG providing original references to the retrieved evidence, allows for verification, making the process of retrieval and reasoning more interpretable than a long-context model in a black-box fashion. In the future, new RAG methods with extended context in an efficient manner should also be explored [34].

C. Hybrid Approaches Combining RAG and Fine-Tuning

A very promising future direction seems to be in combination of generation with retrieval in an end-to-end, fine-tuned setup. Hybrid approaches might provide a good balance between these two as you can utilize RAG to generate dynamic information and further fine-tune the generated content with some handcrafting to provide more contextualization or customization. The big question is how to perform integration (in sequential processes, alternating retrieval-fine-tuning, or end-to-end joint training). For example, RA-DIT aligns the scoring functions between retrievers and generators under KL divergence, which provides an actionable way to reconcile these techniques [35]. Future work needs to investigate in more detail how these methodologies can be effectively combined to the advantages of strengthening model ability and application adaptiveness.

D. Development of End-to-End RAG Models

This is even a little bit better because right now we are in that kind of phase in RAG, which is that we want to build entire end-to-end models for RAG but we also want pre-trained models, especially for RAG tasks. These models have an inherent mechanism to combine the retrieval and generation phases and make the implementation of an RAG system less complex and more efficient. More recent work, such as RETRO++ [34], have broken through on the scalability of these models. However, the parameter numbers in RAG models still fall short of the LLMs, indicating concerns with the scalability and efficiency of RAG systems. Another direction for future investigation is that of inverse scaling laws: the idea of using smaller, more efficient models which, in some instances, may already outperform their larger equivalents [36].

979-8-3315-1533-1/25/$31.00 ©2025 IEEE

VII. Conclusion

In this survey, we provide a brief history and application of RAG and review its progress on three developmental paradigms: Naive, Advanced, and Modular RAG. The application domain of RAG is expanding to multimodal domains involving processing disparate data modalities like images, videos, and code. Given its clear practical insights, this extension generates interest across industry and academia, The growth of RAG-centric applications, and materials that aid them, shows the rise in the RAG ecosystem. As RAG applies to a larger blend of applications, new methodologies to assess refineries are needed to capture more accurately RAGs contribution. While RAG has brought LLMs forward quite a bit, improvements can still be made in robustness, long-context integration, hybrid approaches, end-to-end development processes, practical deployments in commercial settings, and multimodal capabilities. In future work, we aim address these aspects to improve the effectiveness of RAG and make it useful for practical situations.

References

[1] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, *et al.*, "Siren's song in the ai ocean: A survey on hallucination in large language models," *arXiv preprint arXiv:2309.01219*, 2023.

[2] Z. Liu, J. Li, M. Huang, Z. Su, and F. Wei, "Leveraging model-based retrieval-augmented generation models for fact extraction and verification," *arXiv preprint arXiv:2206.10341*, 2022.

[3] J. Pan, Y. Gao, Y. Xiong, M. Wang, and H. Wang, "Towards robust retrieval-augmented generation: Mitigating the impact of irrelevant documents in knowledge-intensive tasks," *arXiv preprint arXiv:2301.13117*, 2023.

[4] J. Sun, K. Jia, and X. Gao, "Rag-enhanced large language models for open-domain question answering," *arXiv preprint arXiv:2310.05451*, 2023.

[5] Y. Chen, J. Wu, and H. Wang, "Rag-c: Retrieval-augmented generation with cross-attention for knowledge-grounded dialogue systems," *arXiv preprint arXiv:2302.03412*, 2023.

[6] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*, pp. 253–262, 2004.

[7] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 1, pp. 117–128, 2010.

[8] Z. Yu, C. Xiong, S. Yu, and Z. Liu, "Augmentation-adapted retriever improves generalization of language models as generic plug-in," *arXiv preprint arXiv:2305.17331*, 2023.

[9] X. Huang and Q. Hu, "A bayesian learning approach to promoting diversity in ranking for biomedical information retrieval," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 307–314, 2009.

[10] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024.

[11] N. Anderson, C. Wilson, and S. D. Richardson, "Lingua: Addressing scenarios for live interpretation and automatic dubbing," in *Proceedings of the 15th Biennial Conference of the Association for Machine Translation in the Americas (Volume 2: Users and Providers Track and Government Track)*, pp. 202–209, 2022.

[12] Z. Shao, Y. Gong, Y. Shen, M. Huang, N. Duan, and W. Chen, "Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy," *arXiv preprint arXiv:2305.15294*, 2023.

[13] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, "Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions," *arXiv preprint arXiv:2212.10509*, 2022.

[14] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," *arXiv preprint arXiv:2305.06983*, 2023.

[15] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, "Query rewriting for retrieval-augmented large language models," *arXiv preprint arXiv:2305.14283*, 2023.

[16] X. Wang, Q. Yang, Y. Qiu, J. Liang, Q. He, Z. Gu, Y. Xiao, and W. Wang, "Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases," *arXiv preprint arXiv:2308.11761*, 2023.

[17] X. Cheng, D. Luo, X. Chen, L. Liu, D. Zhao, and R. Yan, "Lift yourself up: Retrieval-augmented text generation with self-memory," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[18] Z. Dai, V. Y. Zhao, J. Ma, Y. Luan, J. Ni, J. Lu, A. Bakalov, K. Guu, K. B. Hall, and M.-W. Chang, "Promptagator: Few-shot dense retrieval from 8 examples," *arXiv preprint arXiv:2209.11755*, 2022.

[19] W. Yu, D. Iter, S. Wang, Y. Xu, M. Ju, S. Sanyal, C. Zhu, M. Zeng, and M. Jiang, "Generate rather than retrieve: Large language models are strong context generators," *arXiv preprint arXiv:2209.10063*, 2022.

[20] Z. Sun, X. Wang, Y. Tay, Y. Yang, and D. Zhou, "Recitation-augmented language models," *arXiv preprint arXiv:2210.01296*, 2022.

[21] I. Nguyen, "Evaluating rag part i: How to evaluate document retrieval," 2023.

[22] J. Liu, "Building production-ready rag applications," 2023.

[23] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "Ragas: Automated evaluation of retrieval augmented generation," *arXiv preprint arXiv:2309.15217*, 2023.

[24] Y. Liu, L. Huang, S. Li, S. Chen, H. Zhou, F. Meng, J. Zhou, and X. Sun, "Recall: A benchmark for llms robustness against external counterfactual knowledge," *arXiv preprint arXiv:2311.08147*, 2023.

[25] D. Arora, A. Kini, S. R. Chowdhury, N. Natarajan, G. Sinha, and A. Sharma, "Gar-meets-rag paradigm for zero-shot information retrieval," *arXiv preprint arXiv:2310.20158*, 2023.

[26] J. Saad-Falcon, O. Khattab, C. Potts, and M. Zaharia, "Ares: An automated evaluation framework for retrieval-augmented generation systems," *arXiv preprint arXiv:2311.09476*, 2023.

[27] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, *et al.*, "Natural questions: a benchmark for question answering research," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 453–466, 2019.

[28] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning, "Hotpotqa: A dataset for diverse, explainable multi-hop question answering," *arXiv preprint arXiv:1809.09600*, 2018.

[29] A. Fan, Y. Jernite, E. Perez, D. Grangier, J. Weston, and M. Auli, "Eli5: Long form question answering," *arXiv preprint arXiv:1907.09190*, 2019.

[30] T. Möller, A. Reina, R. Jayakumar, and M. Pietsch, "Covid-qa: A question answering dataset for covid-19," in *ACL 2020 Workshop on Natural Language Processing for COVID-19 (NLP-COVID)*, 2020.

[31] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try arc, the ai2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018.

[32] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonellotto, and F. Silvestri, "The power of noise: Redefining retrieval for rag systems," *arXiv preprint arXiv:2401.14887*, 2024.

[33] X. Du and H. Ji, "Retrieval-augmented generative question answering for event argument extraction," *arXiv preprint arXiv:2211.07067*, 2022.

[34] B. Wang, W. Ping, P. Xu, L. McAfee, Z. Liu, M. Shoeybi, Y. Dong, O. Kuchaiev, B. Li, C. Xiao, *et al.*, "Shall we pretrain autoregressive language models with retrieval? a comprehensive study," *arXiv preprint arXiv:2304.06762*, 2023.

[35] X. V. Lin, X. Chen, M. Chen, W. Shi, M. Lomeli, R. James, P. Rodriguez, J. Kahn, G. Szilvasy, M. Lewis, *et al.*, "Ra-dit: Retrieval-augmented dual instruction tuning," *arXiv preprint arXiv:2310.01352*, 2023.

[36] B. Wang, W. Ping, L. McAfee, P. Xu, B. Li, M. Shoeybi, and B. Catanzaro, "Instructretro: Instruction tuning post retrieval-augmented pretraining," *arXiv preprint arXiv:2310.07713*, 2023.

979-8-3315-1533-1/25/$31.00 ©2025 IEEE