G-Retriever in the context of the RAG research field

Yannick Lang
Matriculation Number: 1995498
DS-SemRAG-M
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University of Bamberg

Submission date: February 20, 2025

Abstract

The research field of Retrieval-Augmented Generation (RAG), a subfield of Natural Language Processing (NLP), combines the strengths of retrieval-based and generation-based systems and has gained significant traction in recent years. This paper provides an overview of key concepts, contributions, and approaches in RAG, with a particular focus on the graph-based G-Retriever model. It examines the similarities and differences between G-Retriever and selected other models, highlighting its unique characteristics and potential advantages.

Contents

While Large Language Models (LLMs) have demonstrated strong performance across various Natural Language Processing (NLP) tasks, including question answering, they are prone to hallucinations and rely on static, potentially outdated information (Marcus, 2020). LLMs encode knowledge within their model weights (Petroni et al., 2019), but their capacity is constrained by model size, and their knowledge remains fixed after training. Updating or modifying this knowledge is computationally expensive and requires full or partial retraining (Guu et al., 2020).

Retrieval-Augmented Generation (RAG), a subfield of NLP, addresses these limitations by integrating retrieval-based and generation-based approaches. By retrieving relevant information from external knowledge sources and incorporating it into the generation process, RAG enhances output accuracy and adaptability.

1 Retrieval Augmented Generation

This section provides an overview of the research field of Retrieval-Augmented Generation (RAG), focusing on its origins and key contributions leading to the G-Retriever model.

RAG, a subfield of Natural Language Processing (NLP), has gained significant traction since its introduction in 2020. It combines retrieval-based and generation-based models to enhance text generation quality by incorporating information from retrieved documents.

The foundational work in this field was introduced by (Lewis et al., 2020), who proposed the RAG framework. Their key innovation was enabling language models to access external knowledge sources during generation, allowing them to condition outputs on retrieved information in addition to the static knowledge embedded in their weights. The authors demonstrated that external knowledge sources could be updated, expanded, and refined independently of the model, facilitating access to more up-to-date and diverse information without the computational cost of retraining large language models.

The core mechanism is straightforward: a retrieval component selects relevant documents based on the query, and this retrieved information is integrated into the generation process. This can be done by simply prepending the retrieved text to the query or through more sophisticated mechanisms that enable end-to-end training (Ram et al., 2023).

Given its effectiveness in knowledge-intensive tasks, RAG is widely used in question answering. The choice of knowledge source depends on the application, with common options including Wikipedia and scientific literature. Some approaches leverage the internet for real-time information, benefiting from search engines like Google or Bing. However, this also introduces potential noise and misinformation compared to curated sources. If dedicated search engines are not used, document relevance can be determined using vector similarity methods.

A crucial design choice in RAG models is the number of retrieved documents, typically ranging from three to five per query. Some models perform a single retrieval step per query, conditioning the entire response on a fixed set of documents, while others retrieve documents iteratively—up to once per generated token (Lewis et al., 2020). Models also vary in what parts of the system are static or trainable, with some freezing the retrieval component and others fine-tuning it jointly with the generative model (Guu et al., 2020). Some systems allow for off-the-shelf language models to be used without any training, thus enabling the use of pre-trained models via API access.

Adding retrieval results to the generation process has been shown to improve model performance, often matching or surpassing significantly larger models that lack external knowledge access (Ram et al., 2023). Moreover, retrieval enhances response quality, mitigates the limitations of small context windows, and reduces computational resource requirements (Xu et al., 2023).

However, there is no standardized dataset for benchmarking RAG models, as authors often create their own datasets from scratch or by combining existing ones. Additionally, there is no clear consensus on evaluation metrics. While accuracy and exact match are commonly used, some studies also — or exclusively — consider alternative metrics such as perplexity, (Q-)BLEU, or (K)F1.

2 G-Retriever

This section introduces the 2024 G-Retriever model, outlining its motivation, architecture, and evaluation results. In the following section, G-Retriever is compared to other retrieval-augmented models to highlight key differences

2.1 Motivation

G-Retriever applies the RAG approach to text-based graphs. Introduced in 2024 by (He et al., 2024), it is designed for question-answering tasks using text-based graphs — i.e., graphs in which both nodes and edges are associated with textual labels — as knowledge sources. Such graphs are prevalent in various domains, including knowledge graphs, recommendation systems, and scene graphs, the authors argue.

Given that these graphs, particularly knowledge graphs, can be large and contain only a subset of nodes and edges relevant to a specific task, G-Retriever aims to retrieve the most relevant subgraph for each generation task. This approach enhances generation efficiency, reduces the required context window length, and improves the quality of generated text by focusing on the most relevant information.

2.2 Architecture

The G-Retriever process consists of four key steps: (1) indexing, (2) retrieval, (3) subgraph construction, and (4) generation.

(1) Indexing Node and edge attributes are converted into vector representations using a pre-trained language model, such as SentenceBERT. The resulting vectors are stored in a nearest-neighbor data structure to facilitate efficient retrieval.

(2) Retrieval The query is encoded using the same pre-trained language model, and the top-k most relevant nodes and edges are retrieved based on embedding similarity. The value of k varies for nodes and edges (3 and 5, respectively, in the authors' experiments, with these values shown to yield optimal results). Previous approaches would directly use the retrieved results as model input, but the authors argue that this is suboptimal, as the most relevant nodes and edges may not be interconnected in the original graph, potentially omitting relevant relationships. To address this, they introduce a subgraph construction step to preserve the structural advantages of graphs over unstructured text.

(3) Subgraph Construction The retrieved nodes and edges are used to form a subgraph that maximizes relevant information while minimizing the inclusion of unnecessary nodes and edges. This is achieved using a modified version of the Prize-Collecting Steiner Tree (PCST) algorithm.

PCST is a well-known optimization problem in graph theory, where the objective is to find a prize-maximizing and cost-minimizing subgraph that connects a set of nodes. In its standard form, PCST assumes that all valuable information resides in nodes, with edges serving solely as connections. However, in the case of text-based graphs, edges can also contain meaningful textual attributes. To accommodate this, the authors modify PCST to assign both prizes and costs to edges. If an edge's prize is less than or equal to its cost, it is treated as a reduced-cost edge. If an edge's prize exceeds its cost, a virtual node is introduced to connect the original nodes. The prize of this node is set to the difference between the prize and cost of the original edge. The new edges introduced by this transformation are assigned a cost of zero.

By adapting PCST in this manner, the same optimization algorithm can be applied while preserving valuable edge information.

(4) Generation Once the optimal subgraph has been constructed, it is passed to the generative model. The subgraph is incorporated in two ways:

1. **Graph Encoding:** The subgraph is processed using a graph encoder, projected to the appropriate dimension via a projection layer, and supplied to the LLM for generation.

2. **Textual Representation:** The subgraph is converted into a structured text format (similar to CSV) and prepended to the query.

The first approach enables soft prompt tuning, while the second provides explicit context to the model. The authors note that although their chosen textual representation may not be optimal, it performs well in practice. An ablation study confirms that both encoded and textualized graphs contribute significantly to performance, with the projection layer having the least impact, serving primarily to align dimensions between the graph encoder and LLM.

2.3 Evaluation and Results

The authors evaluate G-Retriever on a dataset of text-based graphs, queries, and expected answers. This dataset is created by converting existing benchmarks — ExplaGraphs (debate stance prediction, (Saha et al., 2021)), SceneGraphs (visual question answering, (Hudson and Manning, 2019)), and WebQSP (large-scale knowledge graphs, (Yih et al., 2016)) — into a unified format.

The dataset represents three distinct evaluation scenarios:

- Common sense reasoning: Determining whether common sense related arguments support or refute each other (evaluated using accuracy).

- Scene Graph Question Answering: Answering open-ended questions about spatial relationships within a scene graph (evaluated using accuracy).

- Knowledge based Question Answering: Answering queries based on a knowledge graph, potentially requiring multi-hop reasoning (evaluated using Hit@1).

The results demonstrate that G-Retriever outperforms baseline models across all three scenarios. The baselines include LLMs that receive no additional information, the entire graph as input, or the top-k retrieved nodes and edges without subgraph construction.

Additionally, the authors show that each component of G-Retriever contributes to its performance. Notably, G-Retriever exhibits reduced hallucination rates compared to the baselines, as evaluated in spot checks.

A particularly interesting aspect of G-Retriever is explainability: Since the model generates answers based on a well-defined subgraph, the reasoning process can be easily fact-checked by inspecting the nodes and edges involved. However, this aspect is not evaluated further in the paper.

3 Comparison to selected papers

This section contextualizes G-Retriever within the field of retrieval-augmented generation (RAG) by comparing it to selected models and highlighting key similarities and differences.

3.1 Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

The original RAG paper by Lewis et al. (2020) introduced the retrieval-augmented generation framework, enabling pre-trained language models to retrieve external knowledge during text generation.

Like G-Retriever, RAG retrieves relevant information based on embedding similarity. However, whereas G-Retriever utilizes text-based graphs as its knowledge source, RAG operates on chunked Wikipedia articles. Another key distinction is that RAG fine-tunes its retriever jointly with the generative model, whereas G-Retriever's retrieval component remains frozen during training. Both models primarily target question-answering tasks.

3.2 REALM: Retrieval-Augmented Language Model Pre-Training

REALM (Guu et al., 2020) is also one of the early retrieval-augmented language models. It employs masked language modeling for pre-training on large text corpora, followed by fine-tuning for open-domain question answering. Like RAG, it uses Wikipedia articles as its external knowledge source rather than structured graphs.

A key distinction from G-Retriever is that REALM fine-tunes its retriever model alongside the generative model, propagating gradients across all retrieved documents. Additionally, it leverages salient span masking, a technique not utilized in G-Retriever.

3.3 In-Context Retrieval-Augmented Language Models

This work (Ram et al., 2023) demonstrates that the core benefits of RAG can be achieved using a simple approach: retrieving documents and prepending them to the query before passing it to an off-the-shelf LLM—without requiring any additional training.

Most RAG-based papers, including G-Retriever, modify LLM architectures to integrate retrieval, with varying degrees of trainable components. In contrast, this in-context approach allows for immediate deployment and eliminates the need for training resources, making it well-suited for scenarios that rely on API-based LLMs.

In contrast to G-Retriever, which performs retrieval only once per query, this approach retrieves documents iteratively, potentially up to once per generated token. The authors also demonstrate that general-purpose retrievers, such as BM25, can outperform embedding-based retrievers like those used in G-Retriever, particularly when combined with additional reranking.

3.4 Internet-Augmented Dialogue Generation

Komeili et al. (2022) describe a dialogue-focused system that grounds its responses using real-time information retrieved from the internet. To achieve this, the system generates a query based on its context and submits it to a search engine such as Bing. The retrieved documents are then incorporated into the context for generating the model's response.

By leveraging live internet data, this approach ensures access to the most up-to-date information, similar to how a human might search for facts before answering a question.

Beyond the general concept of retrieving external knowledge to enhance generation quality, this approach shares little in common with G-Retriever. While the authors of G-Retriever (He et al., 2024) briefly mention the possibility of a "Chat-with-your-graph" system, they do not explore this idea in detail. As a result, G-Retriever is not primarily designed as a dialogue-oriented system.

3.5 Retrieval-Augmented Retriever Multimodal Language Modeling

RA-CM3 is a retrieval-augmented multi-modal model in which both the knowledge sources and generated outputs can be either text or images (Yasunaga et al., 2023).

Similar to other RAG approaches, RA-CM3 employs Maximum Inner Product Search (MIPS) to retrieve relevant documents from a candidate pool. However, it requires a specialized multi-modal encoder, such as CLIP, to process both textual and visual data.

The authors demonstrate that RA-CM3 improves generation quality while reducing training resource requirements. Additionally, they explore alternative use cases, such as controlled image generation and image editing.

While G-Retriever is evaluated on spatial reasoning tasks, it does not support images as input or output. Instead, it relies on textual graph representations of scenes as its knowledge source.

3.6 Retrieval meets Long Context Large Language Models

How do retrieval augmentation and long context windows for LLMs compare, and how does a combination of both perform? These are the central questions investigated by Xu et al. (2023) in their 2024 study.

The authors evaluate their system on a question-answering dataset in which some queries require multi-hop reasoning, similar to the tasks used for G-Retriever. Their findings suggest that retrieval augmentation can match or even surpass the performance of models with significantly larger context windows while also reducing inference time.

Xu et al. (2023) further highlight that the lost-in-the-middle problem — where relevant information is buried within a long context window — can be mitigated through retrieval augmentation. This aligns with the findings of G-Retriever, which also demonstrates improved performance over baseline models by incorporating only the most relevant subgraphs rather than the entire graph.

3.7 Benchmarking Large Language Models in Retrieval-Augmented Generation

Chen et al. (2023) identify several key challenges faced by RAG models:

Noise robustness The ability to generate accurate responses when some retrieved documents are irrelevant.

Negative rejection The ability to refrain from answering when sufficient information is unavailable.

Information integration The ability to aggregate knowledge from multiple sources.

Counterfactual robustness The ability to detect and resolve conflicting information.

While G-Retriever is not among the systems that have been examined for this paper, these could still be relevant. One exception might be information integration, since G-Retriever only uses a single (sub-) graph to condition on.

3.8 G-Retriever

How does G-Retriever compare to these models?

The main distinction of G-Retriever is its focus on text-based graphs as knowledge sources, along with a subgraph construction step that is absent or not needed in other approaches.

Additionally, G-Retriever employs prompt tuning, which is not common in most RAG models. Unlike some retrieval-augmented systems, G-Retriever's retrieval component remains frozen during training. The language model itself is also frozen, with the graph encoder and projection layer being the only trainable parts.

While the authors mention the possibility of using the model in a dialog system, in this paper they use G-Retriever for question answering, which is a very popular use case for RAG systems.

4 Conclusion

The field of retrieval-augmented generation presents a promising approach for enhancing the quality of generated text by incorporating information from retrieved documents while maintaining lower resource requirements compared to scaling up language models.

While most RAG systems focus on unstructured text sources — such as news articles, scientific papers, or Wikipedia — G-Retriever distinguishes itself by leveraging structured graphs as its knowledge source. This approach enables the system to utilize both the structure of the graph and its textual content, potentially enhancing reasoning capabilities. However, it also introduces the challenge of designing a more complex retrieval mechanism compared to standard text-based approaches.

One promising yet underexplored avenue in RAG research is the use of retrieved information to explain the model's reasoning process. Enhancing transparency in this way could provide users with more contextual background, facilitate fact-checking, and improve the detection of potential errors in the model's output. Future work could further investigate this aspect, potentially contributing to more interpretable and trustworthy AI systems.

References

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2023. Benchmarking large language models in retrieval-augmented generation. In *AAAI Conference on Artificial Intelligence*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Xiaoxin He, Yijun Tian, Yifei Sun, N. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *ArXiv*, abs/2402.07630.

Drew A. Hudson and Christopher D. Manning. 2019. Gqa: a new dataset for compositional question answering over real-world images. *ArXiv*, abs/1902.09506.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2022. Internet-augmented dialogue generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8460–8478, Dublin, Ireland. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401.

Gary F. Marcus. 2020. The next decade in ai: Four steps towards robust artificial intelligence. *ArXiv*, abs/2002.06177.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Swarnadeep Saha, Prateek Yadav, Lisa Bauer, and Mohit Bansal. 2021. ExplaGraphs: An explanation graph generation task for structured commonsense reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7716–7740, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence C. McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2023. Retrieval meets long context large language models. *ArXiv*, abs/2310.03025.

Michihiro Yasunaga, Armen Aghajanyan, Weijia Shi, Rich James, Jure Leskovec, Percy Liang, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. Retrieval-augmented multimodal language modeling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.