Comparative Analysis of RAG and GraphRAG Retrievers for Knowledge Retrieval in Diverse Datasets

Jiayi Chen
Ying Wu College of Computing
New Jersey Institute of Technology
Newark, USA
jc2693@njit.edu

***Abstract*—This study presents a comparative analysis of retriever part of traditional Retrieval-Augmented Generation (RAG) system, and GraphRAG, a graph-enhanced RAG approach. By leveraging datasets across multiple domains, including SQuAD, PubMedQA, and arXiv, the research evaluates retrieval quality, efficiency, and the impact of graph structures on performance. The experiments show that while GraphRAG improves retrieval precision and recall in scenarios that has heavy context, it comes at the cost of higher computational latency and struggle to handle large data samples. The results provide insights into the optimal use cases for graph-based retrieval systems and propose guidelines for selecting between traditional and graph-based RAG architectures based on specific requirements.**

***Index Terms*—RAG, GraphRAG, Knowledge Retriever**

I. Introduction

Retrieval-Augmented Generation (RAG) systems [1] have emerged as a powerful approach for combining pre-trained language models with external knowledge sources to improve performance across a variety of tasks, including question answering and document retrieval. Traditional RAG systems (BaseRAG) rely on dense retrieval methods that compute embedding similarities between queries and documents. However, these methods may overlook relationships between documents that can be exploited using graph-based approaches.

GraphRAG [2] introduces graph structures into the retrieval process, connecting documents based on their semantic or structural relationships and employing algorithms such as PageRank to enhance retrieval quality. While promising, the performance and computational trade-offs between RAG and GraphRAG remain underexplored across diverse datasets and domains.

This study aims to address this gap by conducting a systematic comparison of retrieval capacity between BaseRAG and GraphRAG. The evaluation spans multiple datasets, including SQuAD (general knowledge), PubMedQA (medical domain), and arXiv (scientific literature), focusing on retrieval quality, efficiency, and the influence of graph-based enhancements. By analyzing results across different domains, this work seeks to identify scenarios where graph-based methods provide advantages and to offer practical guidelines for selecting between RAG architectures.

The report is structured as follows: the methodology details the implementation of BaseRAG and GraphRAG, including RAG Architecture construction, dataset preparation and per formance metrics. The experimental results provide a quantitative comparison of the two systems. Finally, the discussion highlights the strengths and limitations of graph-based retrieval and proposes recommendations for future research and application.

II. Methodology

This study evaluates the retrieval performance of two Retrieval-Augmented Generation (RAG) systems—**BaseRAG** and **GraphRAG**—on multiple datasets. The methodology encompasses system design, dataset preparation, evaluation metrics, and experimental configurations to systematically compare the retrievers.

A. System Design

1) BaseRAG: BaseRAG implements traditional dense retrieval using embedding similarity. Documents and queries are encoded into vector representations using pre-trained embedding models (e.g., `multi-qa-mpnet-base-dot-v1` and `all-MiniLM-L6-v2`). The top-K documents are retrieved by ranking their cosine similarity to the query vector. Figure 1 demonstrates the main workflow of BaseRAG.

2) GraphRAG: GraphRAG enhances retrieval by integrating graph-based modeling:

- **Graph Construction:** A k-nearest neighbors (k-NN) graph is constructed where each node represents a document, and edges represent semantic relationships based on cosine similarity between document embeddings. Edge weights are assigned based on the similarity scores, and a threshold ensures meaningful connections.
- **Community Detection:** The graph structure is analyzed using community detection algorithms (e.g., Louvain method) to identify clusters of semantically related documents, which inform retrieval.

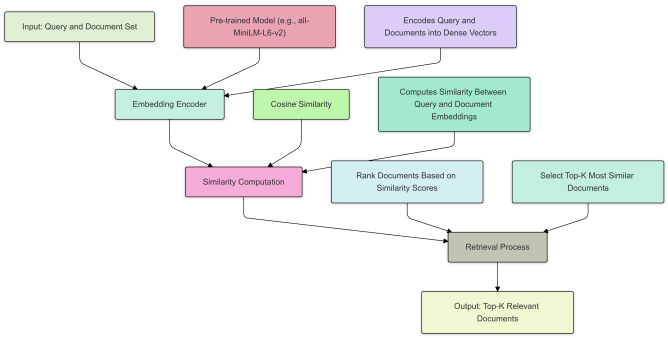

Fig. 1. BaseRAG Flowchart. The workflow of the BaseRAG system begins with inputting a query and a collection of candidate documents. The query and documents are encoded into dense vector representations using a pre-trained embedding model (e.g., all-MiniLM-L6-v2). These embeddings are processed in the similarity computation module, where cosine similarity is used to calculate relevance scores. Documents are then ranked based on these scores, and the top-K most similar documents are selected and returned as the output.

- **Ranking Mechanism:** Retrieval incorporates multiple factors:
  - Cosine similarity between query and document embeddings.
  - PageRank scores to capture document importance within the graph.
  - Community information to boost scores for documents in the same cluster as the query's most relevant document.

Graph parameters such as *k_neighbors*, *sim_weight*, and *cluster_weight* are tuned to optimize retrieval quality, balancing local similarity, graph connectivity, and community relevance. Figure 2 illustrates the workflow of GraphRAG.

B. Datasets

Three datasets spanning diverse domains are used for evaluation:

1) **SQuAD (Wikipedia-based question-answering):** General knowledge questions and their corresponding document passages.
2) **PubMedQA (Medical domain):** Questions related to medical research with abstracts from PubMed as the retrieval pool.
3) **arXiv (Scientific literature):** Research papers, with queries centered on technical or scientific topics.

Each dataset provides questions (queries) and relevant documents (ground truth) for benchmarking.

C. Evaluation Metrics

The systems are evaluated on both retrieval quality and efficiency:

- **Retrieval Quality:**
  - *Hits:* Proportion of queries for which relevant documents appear in the top-K results.
  - *Mean Reciprocal Rank (MRR):* A measure of the ranking of the first relevant document.
  - *Precision@K:* Proportion of retrieved documents in the top-K that are relevant.
  - *Recall@K:* Proportion of all relevant documents captured in the top-K results.
- **Efficiency:**
  - *Average Retrieval Time:* Time taken to retrieve the top-K results for a query.

D. Experimental Configuration

- **Embedding Models:** Pre-trained models `multi-qa-mpnet-base-dot-v1` and `all-MiniLM-L6-v2` are used for encoding queries and documents.
- **Parameter Tuning (GraphRAG):**
  - *k_neighbor:* Controls the number of nearest neighbors for each node in the graph.
  - *sim_weight:* Weights the similarity score between nodes.
  - *cluster_weight:* Adjusts the influence of graph-based clustering on retrieval.
- **Sample Sizes:** The datasets are sampled at varying sizes (e.g., 1000, 5000, 10,000, etc.) to analyze scalability and robustness of the systems.
- **Top-K Retrieval:** The experiments evaluate retrieval quality for different values of K (e.g., 5, 10, 15).

E. Procedure

1) **Data Preprocessing:** Documents and queries are preprocessed to remove noise and inconsistencies. Queries are mapped to their relevant documents.
2) **Graph Construction (GraphRAG):** Semantic relationships between documents are established using cosine similarity of embeddings. Clustering and ranking algorithms refine retrieval outputs.
3) **Retrieval and Evaluation:** Both BaseRAG and GraphRAG retrieve top-K documents for each query. The retrieved results are compared to the ground truth using the evaluation metrics.
4) **Comparison and Analysis:** Results are analyzed across datasets, sample sizes, and parameter configurations. Trade-offs between retrieval quality and efficiency are assessed.

This methodology ensures a systematic and fair comparison of BaseRAG and GraphRAG, offering insights into their performance across diverse tasks and domains.

III. Experimental Setup

This section outlines the configurations, datasets, and parameters used to evaluate the performance of BaseRAG and GraphRAG.

A. Hardware and Environment

The experiments were conducted on the following setup:

- **Hardware:**
  - GPU: NVIDIA A100

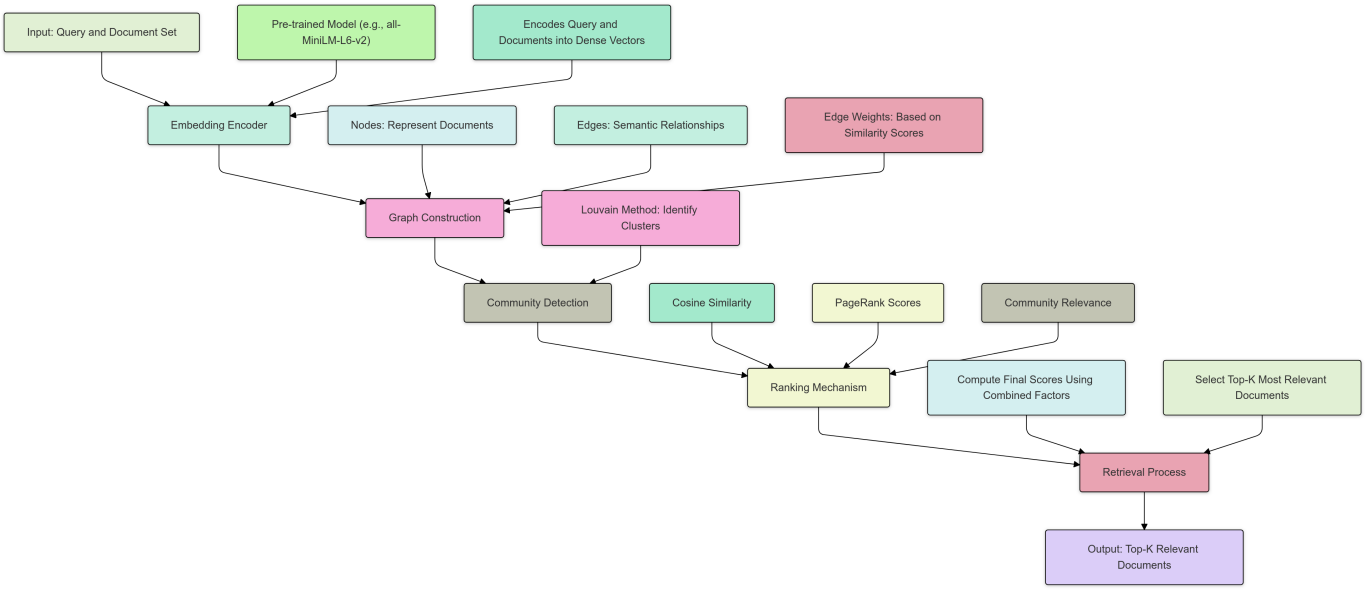

Fig. 2. GraphRAG Flowchart. The GraphRAG workflow begins by encoding the input query and document set using a pre-trained embedding model (e.g., all-MiniLM-L6-v2) to generate dense vector representations. A graph is constructed with documents as nodes, edges representing semantic relationships, and edge weights based on similarity scores. Community detection is performed using the Louvain method to identify clusters of related documents. The ranking mechanism incorporates cosine similarity, PageRank scores, and community relevance to compute final scores for all documents. Based on these scores, the top-K most relevant documents are selected in the retrieval process and returned as the output.

  - CPU: Intel Xeon
  - Memory: 32 GB RAM
- **Software:**
  - Python 3.9
  - Libraries: PyTorch, HuggingFace Transformers, NetworkX

Graph-based computations utilized efficient matrix operations for scalability.

B. Datasets

Three datasets from diverse domains were used to test retrieval systems:

1) **SQuAD:** A question-answering dataset sourced from Wikipedia, consisting of general knowledge questions and their corresponding passages.
   - Total queries: 130,319
   - Relevant documents per query: 1
2) **PubMedQA:** Medical domain dataset consisting of research questions with associated abstracts from PubMed.
   - Total queries: 1,000
   - Relevant documents per query: 1
3) **arXiv:** A collection of scientific papers with queries focusing on research topics.
   - Total queries: 100,000
   - Relevant documents per query: 1

To analyze scalability, subsets of these datasets were created at varying sample sizes: 1,000, 5,000, 10,000, and 20,000 queries.

C. Embedding Models

Pre-trained embedding models were used to encode both documents and queries:

- **`multi-qa-mpnet-base-dot-v1`:** A model optimized for dense retrieval tasks.
- **`all-MiniLM-L6-v2`:** A lightweight model providing a trade-off between computational cost and performance.

Each document and query was represented as a fixed-length embedding vector.

D. Parameters

For GraphRAG, the following parameters were tuned:

- **k_neighbor:** Controls the number of nearest neighbors per node in the graph. Values tested: $\{5, 10\}$.
- **sim_weight:** Adjusts the weight of the similarity score in the graph's edge calculation. Values tested: $\{0.6, 0.7\}$.
- **cluster_weight:** Defines the influence of clustering algorithms during graph traversal. Values tested: $\{0.1, 0.2, 0.3, 0.5, 0.7\}$.

E. Evaluation Metrics

Both BaseRAG and GraphRAG were evaluated using the following metrics:

- **Hits@K:** Measures whether the relevant document is among the top-K retrieved documents.
- **Mean Reciprocal Rank (MRR):** Evaluates the rank position of the first relevant document in the retrieval results.
- **Precision@K:** Calculates the proportion of relevant documents in the top-K retrieved results.

- **Recall@K:** Measures the proportion of relevant documents retrieved out of the total relevant documents.
- **Average Retrieval Time:** Captures the time taken to retrieve top-K results per query.

F. Procedure

1) **Data Preprocessing:** Queries and documents were extracted, cleaned and tokenized.
2) **Graph Construction (GraphRAG):** Semantic relationships between documents were established using cosine similarity of embeddings. Clustering and ranking algorithms refined retrieval outputs.
3) **Retrieval and Ranking:** BaseRAG retrieved documents based on dense embedding similarity. GraphRAG refined retrieval using graph-based algorithms, such as PageRank, on the constructed graph.
4) **Comparison:** Results for BaseRAG and GraphRAG were compared across all datasets and sample sizes to analyze the trade-offs between retrieval quality and computational cost.

IV. Results and Analysis

This section presents the results of the experiments and analyzes the performance of BaseRAG and GraphRAG across the datasets (SQuAD, PubMedQA, and arXiv) in terms of retrieval quality, efficiency, and scalability.

A. Benchmark Tables Overview

The results of the experiments are summarized in two tables:

- **Table I:** Evaluates BaseRAG and GraphRAG across three datasets (SQuAD, PubMedQA, arXiv) using different configurations of sample sizes, *k*, *k_neighbor*, *sim_weight*, and *cluster_weight*. Metrics such as Hits@K, MRR, Precision@K, Recall@K, and average retrieval time are reported.
- **Table II:** Focuses specifically on the PubMedQA dataset, comparing the two methods for larger sample sizes (5,000, 10,000, 20,000) under different parameter settings, highlighting retrieval quality and efficiency trade-offs.

B. Retrieval Quality

1) SQuAD (General Knowledge):

- **Performance:** Both BaseRAG and GraphRAG achieved high Hits@K (˜0.98–0.99) and MRR (˜0.92–0.94) across sample sizes (Table I). For example, at a sample size of 1,000, BaseRAG achieved a Hits@K of 0.9840, while GraphRAG slightly outperformed it at 0.9850.
- **Precision and Recall:** GraphRAG showed marginal improvements in precision and recall at smaller sample sizes (e.g., Precision@K of 0.0985 for GraphRAG vs. 0.0984 for BaseRAG at 1,000 queries in Table I). These differences became negligible as the sample size increased.
- **GraphRAG Insights:** Graph-based enhancements provided limited benefits for general knowledge datasets, where document relationships play a minor role.

2) PubMedQA (Medical Domain):

- **Performance:** GraphRAG consistently outperformed BaseRAG in precision and recall (Table II). For example, at 20,000 queries, GraphRAG achieved a Precision@K of 0.1303 compared to BaseRAG's 0.1294. Moreover, GraphRAG outperformed BaseRAG on PubMedQA in terms of Hits@K and MRR, consistently achieving higher values across varying configurations. For example, in Table II, at a sample size of 5,000, GraphRAG achieved a Hits@K of 0.9130 and an MRR of 0.6446, compared to BaseRAG's 0.9060 and 0.6437, respectively, demonstrating the benefits of its graph-based retrieval approach for domain-specific tasks.
- **Graph Benefits:** The graph structure effectively captured complex inter-document relationships, leading to better precision and recall. For instance, GraphRAG achieved a Recall@K of 0.9130 at a sample size of 5,000 compared to 0.9060 for BaseRAG (Table II).
- **Consistency:** Hits@K and MRR remained comparable, with both systems maintaining consistently high scores across varying configurations in Table II.

3) arXiv (Scientific Literature):

- **Performance:** Both systems performed exceptionally well, with Hits@K exceeding 0.96 and MRR around 0.90–0.95 across sample sizes (Table I). For example, at a sample size of 1,000, both BaseRAG and GraphRAG achieved a Hits@K of 0.9860.
- **GraphRAG Insights:** Minimal improvements in precision and recall suggest that the added graph complexity offers limited benefits for scientific literature retrieval, which relies more on direct textual matching.

C. Efficiency

- **BaseRAG:** Demonstrated significantly faster retrieval times, making it more efficient for real-time applications. For example, on PubMedQA with 5,000 queries, BaseRAG achieved an average retrieval time of 0.0157 seconds compared to GraphRAG's 0.0612 seconds (Table II).
- **GraphRAG:** Incurred higher computational costs, particularly for larger datasets, due to graph construction and traversal. On SQuAD (Table I), GraphRAG took ˜4–5 times longer than BaseRAG at larger sample sizes.
- **Trade-off Example:** The efficiency gains of BaseRAG come at a minor cost to retrieval quality, especially in domain-specific tasks (Tables I and II).

D. Scalability

- **Performance Trends:** Both systems experienced slight declines in Hits@K and MRR as sample sizes increased, reflecting the increased complexity of the retrieval task (Tables I and II).
- **GraphRAG Overhead:** Non-linear scaling of computational costs, driven by higher *k_neighbor* and graph complexity, impacted efficiency significantly at larger scales (Table II).

TABLE I
BENCHMARK RESULTS FOR BASERAG AND GRAPHRAG ACROSS DATASETS WITH VARYING CONFIGURATIONS.

Dataset	Method	Hits	MRR	Precision	Recall	Avg Time (s)
Sample Size: 1000, $k = 10$, $k_{neighbor} = 10$, $sim_{weight} = 0.7$, $cluster_{weight} = 0.1$ Embedding Model: multi-qa-mpnet-base-dot-v1						
SQuAD	BaseRAG	0.9840	0.9235	0.0984	0.9840	0.0228
	GraphRAG	0.9850	0.9237	0.0985	0.9850	0.0582
PubMedQA	BaseRAG	0.9430	0.6720	0.0943	0.9430	0.0257
	GraphRAG	0.9430	0.6715	0.0943	0.9430	0.1117
arXiv	BaseRAG	0.9820	0.9500	0.0982	0.9820	0.0234
	GraphRAG	0.9820	0.9498	0.0982	0.9820	0.0625
Sample Size: 5000, $k = 10$, $k_{neighbor} = 10$, $sim_{weight} = 0.7$, $cluster_{weight} = 0.1$ Embedding Model: multi-qa-mpnet-base-dot-v1						
SQuAD	BaseRAG	0.9428	0.8196	0.0943	0.9428	0.0261
	GraphRAG	0.9428	0.8197	0.0943	0.9428	0.2252
PubMedQA	BaseRAG	0.9430	0.6720	0.0943	0.9430	0.0259
	GraphRAG	0.9420	0.6714	0.0942	0.9420	0.1106
arXiv	BaseRAG	0.9640	0.8945	0.0964	0.9640	0.0254
	GraphRAG	0.9636	0.8945	0.0964	0.9636	0.2378
Sample Size: 10000, $k = 10$, $k_{neighbor} = 10$, $sim_{weight} = 0.7$, $cluster_{weight} = 0.1$ Embedding Model: multi-qa-mpnet-base-dot-v1						
SQuAD	BaseRAG	0.9115	0.7637	0.0912	0.9115	0.0366
	GraphRAG	0.9115	0.7637	0.0912	0.9115	0.4164
PubMedQA	BaseRAG	0.9430	0.6720	0.0943	0.9430	0.0262
	GraphRAG	0.9420	0.6714	0.0942	0.9420	0.1115
Sample Size: 1000, $k = 15$, $k_{neighbor} = 10$, $sim_{weight} = 0.7$, $cluster_{weight} = 0.1$ Embedding Model: multi-qa-mpnet-base-dot-v1						
SQuAD	BaseRAG	0.9890	0.9292	0.0659	0.9890	0.0246
	GraphRAG	0.9890	0.9290	0.0659	0.9890	0.0620
PubMedQA	BaseRAG	0.9550	0.6730	0.0637	0.9550	0.0269
	GraphRAG	0.9550	0.6725	0.0637	0.9550	0.1178
arXiv	BaseRAG	0.9830	0.9542	0.0655	0.9830	0.0245
	GraphRAG	0.9830	0.9541	0.0655	0.9830	0.0647
Sample Size: 1000, $k = 5$, $k_{neighbor} = 5$, $sim_{weight} = 0.6$, $cluster_{weight} = 0.2$ Embedding Model: multi-qa-mpnet-base-dot-v1						
SQuAD	BaseRAG	0.9760	0.9145	0.1952	0.9760	0.0242
	GraphRAG	0.9750	0.9136	0.1950	0.9750	0.0468
PubMedQA	BaseRAG	0.9180	0.6686	0.1836	0.9180	0.0273
	GraphRAG	0.9190	0.6686	0.1838	0.9190	0.0739
arXiv	BaseRAG	0.9860	0.9589	0.1972	0.9860	0.0242
	GraphRAG	0.9860	0.9590	0.1972	0.9860	0.0473
Sample Size: 2500, $k = 7$, $k_{neighbor} = 5$, $sim_{weight} = 0.6$, $cluster_{weight} = 0.3$ Embedding Model: all-MiniLM-L6-v2						
SQuAD	BaseRAG	0.9200	0.8093	0.1314	0.9200	0.0159
	GraphRAG	0.9208	0.8092	0.1315	0.9208	0.0718
PubMedQA	BaseRAG	0.9060	0.6437	0.1294	0.9060	0.0155
	GraphRAG	0.9090	0.6436	0.1299	0.9090	0.0605
arXiv	BaseRAG	0.9728	0.9168	0.1390	0.9728	0.0158
	GraphRAG	0.9724	0.9167	0.1389	0.9724	0.0752
Sample Size: 3500, $k = 7$, $k_{neighbor} = 5$, $sim_{weight} = 0.6$, $cluster_{weight} = 0.5$ Embedding Model: all-MiniLM-L6-v2						
SQuAD	BaseRAG	0.9106	0.7972	0.1301	0.9106	0.0172
	GraphRAG	0.9106	0.7970	0.1301	0.9106	0.1032
PubMedQA	BaseRAG	0.9060	0.6437	0.1294	0.9060	0.0152
	GraphRAG	0.9100	0.6438	0.1300	0.9100	0.0598
arXiv	BaseRAG	0.9637	0.9055	0.1377	0.9637	0.0171
	GraphRAG	0.9631	0.9054	0.1376	0.9631	0.1052

TABLE II
BENCHMARK RESULTS FOR BASERAG AND GRAPHRAG ON PUBMEDQA WITH VARYING CONFIGURATIONS.

Method	Hits	MRR	Precision	Recall	Avg Time (s)
Sample Size: 5000, $k = 7$, $k_{neighbor} = 5$, $sim_{weight} = 0.6$, $cluster_{weight} = 0.7$ Embedding Model: all-MiniLM-L6-v2					
BaseRAG	0.9060	0.6437	0.1294	0.9060	0.0157
GraphRAG	0.9130	0.6446	0.1304	0.9130	0.0612
Sample Size: 10000, $k = 7$, $k_{neighbor} = 5$, $sim_{weight} = 0.6$, $cluster_{weight} = 0.7$ Embedding Model: all-MiniLM-L6-v2					
BaseRAG	0.9060	0.6437	0.1294	0.9060	0.0151
GraphRAG	0.9120	0.6440	0.1303	0.9120	0.0610
Sample Size: 20000, $k = 7$, $k_{neighbor} = 5$, $sim_{weight} = 0.6$, $cluster_{weight} = 0.7$ Embedding Model: all-MiniLM-L6-v2					
BaseRAG	0.9060	0.6437	0.1294	0.9060	0.0144
GraphRAG	0.9120	0.6441	0.1303	0.9120	0.0571

- **BaseRAG Scalability:** Maintained efficiency and stable performance, making it better suited for large-scale applications.

E. Parameter Impact (GraphRAG-Specific)

- **k_neighbor:** Increasing neighbors enhanced recall but proportionally increased computational demands, especially at larger sample sizes (Table I).
- **sim_weight:** Fine-tuning graph similarity weights improved retrieval quality (e.g., precision and recall), but gains diminished beyond a threshold (e.g., 0.7 in Table I).
- **cluster_weight:** Adjusting cluster weights enhanced retrieval for PubMedQA but had negligible effects on simpler datasets like SQuAD (Table II).

F. Comparative Insights

- **When GraphRAG is Beneficial:** GraphRAG excels in domain-specific contexts, such as PubMedQA, where capturing document relationships significantly improves retrieval precision and recall (Table II).
- **When BaseRAG is Preferable:** BaseRAG's simplicity and efficiency make it ideal for general-purpose datasets like SQuAD and large-scale datasets like arXiv, where retrieval quality is comparable to GraphRAG (Table I).

G. Summary of Findings

- **Retrieval Quality:** GraphRAG marginally improves retrieval metrics for domain-specific datasets but provides limited benefits for general knowledge or large-scale datasets.
- **Efficiency:** BaseRAG is significantly more efficient and scales better for larger datasets.
- **Use Case Guidelines:**
  - Use GraphRAG for tasks requiring high precision in domain-specific contexts (e.g., medical research).
  - Use BaseRAG for general knowledge retrieval or applications requiring low latency.

These results demonstrate the trade-offs between retrieval quality and efficiency, providing guidelines for selecting the appropriate system based on specific use cases.

V. DISCUSSION AND FUTURE WORK

The results of this study reveal valuable insights into the trade-offs and applications of BaseRAG and GraphRAG for retrieval-augmented generation systems. This section explores the implications of these findings, highlights the strengths and limitations of each system, and provides recommendations for their use in real-world scenarios.

A. Strengths of BaseRAG

BaseRAG demonstrates several strengths that make it an efficient and scalable choice for general retrieval tasks:

- **Efficiency:** BaseRAG consistently outperformed GraphRAG in terms of retrieval speed across all datasets. Its simplicity allows it to scale effectively, even for large datasets such as arXiv.
- **Comparable Retrieval Quality:** Despite its simplicity, BaseRAG achieved similar Hits@K, MRR, Precision@K, and Recall@K scores as GraphRAG in general-purpose datasets like SQuAD and arXiv.
- **Wide Applicability:** The dense retrieval approach of BaseRAG is suitable for applications where computational efficiency is a priority, such as real-time search engines or large-scale document retrieval.

B. Strengths of GraphRAG

GraphRAG leverages graph-based relationships between documents, providing advantages in domain-specific retrieval tasks:

- **Improved Precision and Recall:** GraphRAG showed marginal gains in retrieval quality, particularly for PubMedQA, where the structured relationships between medical abstracts contributed to more accurate retrieval.
- **Contextual Understanding:** By modeling relationships between documents, GraphRAG captures semantic context that may be missed by traditional dense retrieval

methods, making it useful for tasks requiring multi-hop reasoning or related-document retrieval.

C. Limitations of GraphRAG

While GraphRAG offers benefits in specific scenarios, it also has notable limitations:

- **High Computational Overhead:** The graph construction and traversal processes significantly increase retrieval latency, making GraphRAG less suitable for real-time applications or scenarios with stringent performance requirements.
- **Scalability Challenges:** As dataset size increases, the computational cost of maintaining and processing graph structures grows non-linearly, limiting its applicability for very large datasets.

D. Dataset-Specific Implications

The findings suggest that the choice between BaseRAG and GraphRAG depends on the dataset characteristics:

- **General Knowledge (SQuAD):** Both systems performed equally well, with BaseRAG being the preferred choice due to its lower latency.
- **Domain-Specific Knowledge (PubMedQA):** GraphRAG provided slight improvements in precision and recall, suggesting its utility in specialized domains where relationships between documents are critical.
- **Scientific Literature (arXiv):** The benefits of graph-based retrieval were minimal, indicating that BaseRAG is sufficient for large-scale scientific document retrieval.

E. Practical Recommendations

- **When to Use BaseRAG:**
  - Large-scale retrieval tasks where computational efficiency is critical.
  - Applications with limited computational resources or real-time constraints.
- **When to Use GraphRAG:**
  - Domain-specific tasks that benefit from structured relationships between documents (e.g., medical or legal retrieval).
  - Scenarios requiring improved precision for a smaller dataset or more complex queries.

F. Future Directions

While this study provides a comparative analysis of BaseRAG and GraphRAG, further research can explore:

- **Dynamic Graph Construction:** Optimizing graph structures dynamically to reduce computational overhead while retaining retrieval quality.
- **Hybrid Approaches:** Combining BaseRAG and GraphRAG for context-aware retrieval that balances efficiency and precision.
- **Application-Specific Studies:** Testing these systems in real-world applications, such as healthcare, law, and education, to understand their practical benefits and challenges.

VI. Conclusion

This study presented a comparative analysis of two Retrieval-Augmented Generation (RAG) systems—BaseRAG and GraphRAG—across datasets of varying domains, including SQuAD, PubMedQA, and arXiv. The results highlight the trade-offs between retrieval quality and efficiency, providing practical insights into the optimal use cases for each system.

BaseRAG, with its simplicity and computational efficiency, is well-suited for general-purpose and large-scale retrieval tasks. Its dense retrieval approach consistently delivered high retrieval quality while maintaining lower latency, making it ideal for applications requiring real-time performance or resource constraints.

GraphRAG, on the other hand, demonstrated advantages in domain-specific scenarios, particularly in datasets such as PubMedQA, where relationships between documents significantly influence retrieval quality. By leveraging graph structures, GraphRAG achieved marginal improvements in precision and recall. However, these benefits came at the cost of increased computational overhead, limiting its scalability for larger datasets.

The findings suggest that the choice between BaseRAG and GraphRAG depends on the application's domain and requirements. For tasks involving complex relationships and smaller datasets, GraphRAG is a promising choice. Conversely, BaseRAG remains the preferred option for general or large-scale applications.

This research provides a foundation for further exploration into retrieval-augmented systems. Future work includes but not limited to optimizing graph construction for scalability, developing hybrid retrieval approaches, and extending the analysis to application-specific scenarios. These advancements will enable more versatile and efficient RAG systems for a broader range of use cases.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[2] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, "From local to global: A graph RAG approach to query-focused summarization," *arXiv preprint arXiv:2404.16130*, 2024.