GraphRAG: Leveraging Graph-Based Efficiency to Minimize Hallucinations in LLM-Driven RAG for Finance Data

Mariam Barry, Gaëtan Caillaut, Pierre Halftermeyer, Raheel Qader, Mehdi Mouayad, Dimitri Cariolaro, Fabrice Le Deit, Joseph Gesnouin

GraphRAG: Leveraging Graph-Based Efficiency to Minimize Hallucinations in LLM-Driven RAG for Finance Data

Mariam Barry[1], Gaëtan Caillaut[2], Pierre Halftermeyer[3], Raheel Qader[2], Mehdi Mouayad[1], Dimitri Cariolaro[4], Fabrice Le Deit [5], Joseph Gesnouin [6],
[1]BNP Paribas - IT Group - AI & IT Innovation, [2]Lingua Custodia, [3]NEO4J,
[4]Graph Thinking Consulting, [5]BNP Paribas - IT Group - Production, [6]BNP Paribas - Cash Management

Abstract

This study explores the integration of graph-based methods into Retrieval-Augmented Generation (RAG) systems to enhance efficiency, reduce hallucinations, and improve explainability, with a particular focus on financial and regulatory document retrieval. We propose two strategies—FactRAG and HybridRAG—which leverage knowledge graphs to improve RAG performance. Experiments conducted using Finance Bench, a benchmark for AI in finance, demonstrate that these approaches achieve a 6% reduction in hallucinations and an 80% decrease in token usage compared to conventional RAG methods. Furthermore, we evaluate HybridRAG by comparing the Digital Operational Resilience Act (DORA) from the European Union with the Federal Financial Institutions Examination Council (FFIEC) guidelines from the United States. The results reveal a significant improvement in computational efficiency, reducing contradiction detection complexity from $O(n^2)$ to $O(k \cdot n)$—where n is the number of chunks—and a remarkable 734-fold decrease in token consumption. Graph-based retrieval methods can improve the efficiency and cost-effectiveness of large language model (LLM) applications, though their performance and token usage depend on the dataset, knowledge graph design, and retrieval task.

1 Introduction

Generative Artificial Intelligence (GenAI), exemplified by Large Language Models (LLMs) such as OpenAI's GPT series (Brown et al., 2020; OpenAI, 2023), Meta's LLaMA models (Touvron et al., 2023), and Mistral's Mixtral (AI, 2023), has gained prominence in various fields, including healthcare, finance, and education. These models, while highly capable of producing coherent and contextually relevant responses, face challenges in generating factually accurate content—a phenomenon referred to as hallucination (Ji et al., 2023; Bang et al., 2023).

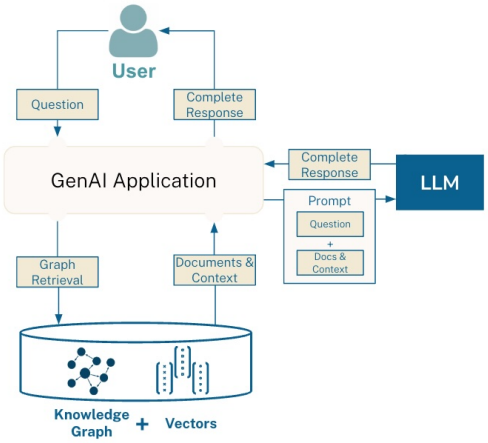

Figure 1: Graph RAG Pattern (Rathle)

Hallucination arises from LLMs' reliance on potentially outdated or domain-general training data, leading to inaccuracies in real-world applications where precision is critical (Dziri et al., 2022).

To address the issue of hallucinations, *Retrieval-Augmented Generation* (RAG) has emerged as a promising approach. Introduced by Lewis et al. (2020), RAG combines a retriever that identifies relevant documents and a generator that creates coherent responses from this information. By combining LLMs with external knowledge bases, RAG systems can enhance response accuracy and relevance by dynamically incorporating up-to-date and verifiable information into generated outputs (Kang and Lee, 2023). The original approach suggested dividing documents into 100-word disjoint chunks, but this can disrupt semantics and lead to hallucinations, as noted by Qian et al. (2024). To mitigate these issues, enhancements like sliding window chunking, sentence-level splitting with surrounding context, and incorporating metadata such as document titles have been proposed to improve the

quality and relevance of the generated outputs (Gao et al., 2023).

While RAG often improves the relevance of language model outputs, it faces notable limitations in real-world applications:

1. **Neglect of Structured Relationships**: Traditional RAG focuses on textual relevance and often overlooks structured relationships critical in domains like citation networks, limiting its effectiveness for complex, interconnected data (Yao et al., 2021).

2. **Redundancy and Lengthy Contexts**: Concatenated text snippets in RAG can lead to redundancy and excessively lengthy inputs, causing the model to lose focus and obscure key information (Longpre et al., 2021).

3. **Limited Global Context**: RAG's restricted retrieval scope hinders its ability to capture broader contexts necessary for tasks like query-focused summarization (Lewis et al., 2020).

These limitations highlight the need for advanced approaches, such as GraphRAG (Figure 1), to incorporate structured relationships and provide richer, more contextually accurate information, while being more efficient in terms of token usage. To overcome these limitations, we explore several distinct approaches to enhancing RAG systems with graphs, including:

1. **FactRAG**: We propose a graph-based approach, "FactRAG," for a question-answering search engine that is more efficient in term of tokens and reduces hallucinations compared to classical RAG.

2. **KG-RAG**: We introduce a knowledge graph-enhanced technique, "KG-RAG," for document comparison tasks, that significantly improves token efficiency and reduces computational complexity from $O(n^2)$ to $O(k \cdot n)$ in LLM-driven retrieval tasks, where n is the number of chunks/nodes and k the number of clusters in the KNN algorithms specifically for detecting contradictions between documents.

3. **HybridRAG**: We provide open-source code for a graph-based Hybrid RAG, which integrates symbolic and sub-symbolic retrieval for flexible question-answering.

2 Related Work

Knowledge Graphs (KGs) play a crucial role in enhancing the interpretability and factual accuracy of large language models (LLMs) by structuring information as entities and relationships (Hogan et al., 2022; Rosin et al., 2022). The integration of graph structures within Retrieval-Augmented Generation (RAG) frameworks has shown significant improvements in model performance. Zhao et al. (2023) demonstrate that Graph-based Retrieval-Augmented Generation enhances contextual accuracy by allowing systems to retrieve relevant entities and relationships from KGs. Similarly, Yasunaga et al. (2022) highlights the benefits of Graph-based Retrieval-Augmented Language Models for fact verification and knowledge enrichment, ensuring that generated outputs are relevant and accurate. Liu et al. (2022) introduces the concept of Graph Retrieval Augmentation, which enhances contextual and semantic understanding in LLMs, resulting in more coherent and pertinent responses. Furthermore, the work of Guu et al. (2020) illustrates how training in a language model with augmented retrieval with knowledge graphs can improve the accuracy and depth of the answer by using KG for the retrieval and structured data. Graph RAG stands out by retrieving graph elements from a pre-constructed knowledge graph, thereby enriching LLM-generated responses with structured knowledge (Rosin et al., 2022). This structure allows Graph RAG to capture semantic nuances, maintain contextual coherence, and reduce verbosity, making it particularly effective for applications in question-answering, recommendation systems, and complex information retrieval tasks relying on structured knowledge.

3 Problem Statement

Large Language Models (LLMs) augmented by Retrieval-Augmented Generation (RAG) systems have advanced the ability to generate contextually relevant responses. However, despite RAG's potential to reduce inaccuracies by integrating external knowledge, hallucinations – defined as the generation of factually incorrect or fabricated information – remain a significant issue. This paper explores the relevance of a graph-based approach to reduce hallucinations and optimize token consumption in language models.

3.1 Graph-based technique

We identify two key approaches to minimizing hallucinations in RAG systems: **pre-generation** and **post-generation**(Agrawal et al., 2023). The pre-generation approach enhances the input context with high-quality, semantically relevant passages to help the model produce accurate outputs. The post-generation approach, on the other hand, validates and corrects factual accuracy using verification processes(Sansford et al., 2024).

Knowledge graphs (KGs) support both approaches by providing structured knowledge. In pre-generation, KGs can insert accurate facts into the input context. In post-generation, KGs help validate generated content for factual correctness.

However, post-generation faces challenges such as converting text to graph representations and the computational costs of iterative LLM calls (Cabot et al., 2023). Additionally, post-generation corrections may introduce further errors, and the iterative calls to LLMs increase computational costs, making large-scale applications impractical. Given these limitations, our work focuses on the pre-generation approach, aiming to reduce hallucinations by using KGs to provide more accurate and reliable context before generation, improving both factual accuracy and efficiency in RAG systems.

3.2 Problem formalization - Building a RAG system with minimal hallucinations

In Retrieval-Augmented Generation (RAG) systems, hallucinations occur when the language model generates content that is factually incorrect or unsupported by retrieved documents. Given a set of documents $D = \{d_1, d_2, \ldots, d_n\}$ and a set of user queries $Q = \{q_1, q_2, \ldots, q_m\}$, the goal is to design a RAG system that minimizes the generation of hallucinated responses while maximizing response accuracy.

We define a RAG system as a function $RAG : Q \to A$, where each query $q \in Q$ is mapped to an answer $a \in A$ based on retrieved context $C \subseteq D$. Let $C(q) = \{c_1, c_2, \ldots, c_k\}$ represent the set of retrieved documents for a query q, where $C(q) \subseteq D$. Es et al. (2023) define the below evaluation measures:

1. **Hallucination Score**, $H(a)$, for each answer $a \in A$ as the proportion of information in a that is unsupported by $C(q)$:

$$H(a) = \frac{\text{Unsupported Information in } a}{\text{Total Information in } a}$$

2. **Faithfulness Score**, $F(a)$, for each answer $a \in A$ as the proportion of information in a that is directly supported by the retrieved context $C(q)$:

$$F(a) = \frac{\text{Supported Information in } a}{\text{Total Information in } a} = 1 - H(a)$$

Objective We aim to minimize the overall hallucination rate $H(A)$ across all answers $A = \{a_1, a_2, \ldots, a_m\}$ while ensuring that each $a \in A$ remains relevant to the query q. This objective can be formulated as:

$$\min_{RAG} H(A) = \frac{1}{m} \sum_{i=1}^{m} H(a_i)$$

subject to:

$$F(a_i) \geq \delta \quad \forall a_i \in A$$

where δ is a predefined faithfulness threshold (e.g., 0.9), ensuring that each generated answer is primarily supported by the retrieved context $C(q)$.

4 Proposal: Graph-based RAG System

We propose to enhance the classical RAG system with knowledge from graph databases instead of raw texts. To this aim, we build a traditional text RAG system to serve as a baseline, as well as two graph-flavored variants, which we call Facts and KG-RAG in the following.

Reproducibility Code for both text and facts RAG are available on Github[1].

4.1 Text RAG

Our text baseline is very classical, we set up a standard RAG pipeline. We relied on the `unstructured`[2] Python package to extract non-overlapping chunks of approximately 500 characters and we used the `all-MiniLM-L6-v2` model provided by `sentence-transformers`[3] (Reimers and Gurevych, 2019) to embed them. We stored the chunks and their embedding inside a `chromadb`[4] database.

4.2 Facts RAG

Our second system rely on LLM to automatically extract entities and relations from raw text, then

[1] https://github.com/gcaillaut/financebench-graph-rag

[2] https://github.com/Unstructured-IO/unstructured

[3] https://github.com/UKPLab/sentence-transformers

[4] https://github.com/chroma-core/chroma

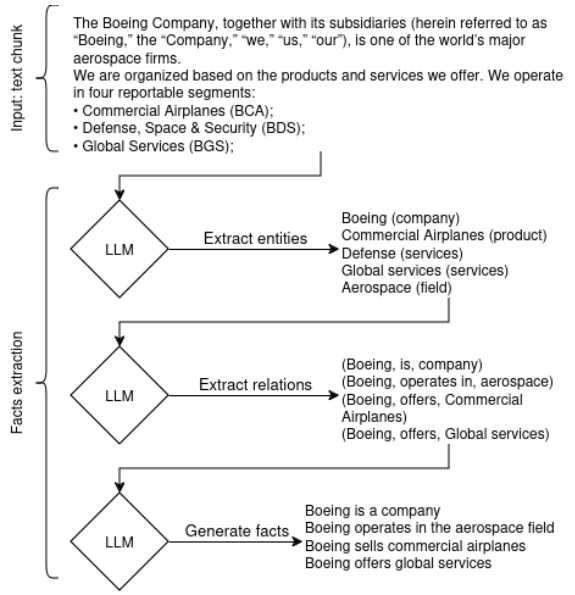

Figure 2: Facts extraction process. We first extract entities from a raw and potentially noisy text. Then we build triples using the text and the extracted entities. Finally, we generate textual description of the triples, which we call *facts*.

convert these triples into short sentences. We call these sentences *facts*. Hence, this system is very similar to our baseline, the difference being an additional step to convert relevant chunks into facts. Practically speaking, we first ask an LLM to extract all entities inside a relevant chunk, then we ask for the relations between them. Finally, the LLM generate triples and a short sentence (the fact) describing the triple in natural language inside a JSON array. The resulting sentences are much more concise, contain less noise, and are more direct. We then provide these generated facts to the LLM instead of the raw chunks. The complete prompt we used to generate these facts is given in Appendix C and the overall process is described in Figure 2.

The purpose of this system is to validate the relevance of LLM-based knowledge graph extraction methods (Zhang and Soh, 2024; Carta et al., 2023) in the context of RAG. While we pointed out the limitations of these approaches in the previous section, we also believe that extracting graphs from text is a powerful summarization and noise filtering tool, as it removes all uninformative tokens and the facts generated are very clear and easy to understand.

4.3 KG-RAG (Knowledge Graph based RAG)

Our third system is based on a graph representation of the document to be queried. The document is processed to extract a knowledge graph, given a predefined graph schema. Then, text chunks and their embeddings are stored inside a node and linked to the entities they contain. More specifically, we rely on the `llm-graph-builder` tool from Neo4j[5] to extract a knowledge graph from pdf files. The tool can also automatically generate a graph schema from raw text, so we use the questions in our dataset (more details in Section 5) to extract a schema suiting our target task.

Finally, we use the user's query to find the most relevant chunks using traditional embedding similarity, then we explore the graph using the chunks as seed to retrieve potentially useful entities and relations, in the form of triples. We limit the exploration of the graph to the direct neighbours of the relevant chunks, but more sophisticated exploration strategies are possible, such as re-ranking documents using graph-based algorithms like PageRank.

4.4 Hybrid RAG

The last system we experimented with aims to leverage explicit and implicit relationships from the knowledge graph using an hybrid architecture. As illustrated in Figure 3.

We introduce a GraphRAG framework that combines explicit (symbolic) and implicit (subsymbolic) retrieval methods to enhance retrieval-augmented generation (RAG) systems. Our approach allows for adaptive retrieval based on the nature of the user question, with Explicit RAG using text-to-Cypher translation for structured queries, while Implicit RAG leverages vector similarity to find k-nearest neighbours. The system employs an LLM to determine the optimal retrieval method, utilizing the retrieved context to generate precise answers, offering a versatile solution for better knowledge retrieval tasks.

Our approach to HybridRAG is tested with a comparative analysis of two regulatory documents: the Digital Operational Resilience Act (DORA) from the European Union and the Federal Financial Institutions Examination Council (FFIEC) guidelines from the United States.

The HybridRAG system is designed to optimize the retrieval and contradiction identification process in large regulatory documents using a knowledge graph-based KNN clustering approach. Traditional Retrieval-Augmented Generation (RAG)

[5] https://llm-graph-builder.neo4jlabs.com/

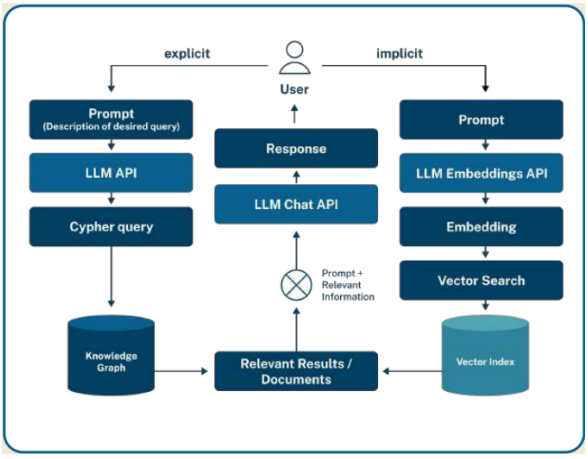

Figure 3: **Hybrid Graph RAG**: Explicit vs. Implicit RAG, *e.g* Symbolic vs. Sub-Symbolic Retrieval. The framework offers two retrieval strategies based on the nature of the user's query.

methods often suffer from high token consumption due to the computationally expensive Cartesian product of document segments. To overcome this, HybridRAG uses KNN clustering to group similar document segments, reducing computational costs. The system captures the context for each document node, generates text embeddings, and clusters nodes based on cosine similarity.

5 Experiments and evaluation

We conducted experiments to address the following two research questions on datasets related to the financial domain.

1. **RQ1:** Does a graph-based RAG system reduce hallucinations compared to a classical RAG system for a question-answering task?

2. **RQ2:** How efficient is Graph/Hybrid RAG in terms of token consumption for retrieval tasks involving document comparison?

5.1 Datasets in Finance Domain

The datasets used in this study include FinanceBench (Islam et al., 2023), a benchmark for evaluating AI systems in finance, which contains various financial documents like regulatory reports and financial statements. Additionally, we utilized DORA (Digital Operational Resilience Act)(European Parliament and Council of the European Union, 2022), a European Union regulation on managing IT risks in the financial sector, and the FFIEC IT Handbook(Federal Financial Institutions Examination Council, 2019), which offers guidelines for IT management in U.S. financial institutions. These data sets were used to assess the performance of the proposed methods in financial document retrieval and contradiction detection.

5.2 Metrics

We will use the metrics of Deep Eval (AI, 2024) to evaluate RQ1. Since we are not interested in assessing the overall RAG quality or the retrieval mechanism, but rather the probability to hallucinate, we focused on the two hallucinations measures available in DeepEval. The Faithfulness Metric first uses an LLM to extract all claims made in the actual_output, before using the same LLM to classify whether each claim is truthful based on the facts presented in the retrieval_context. The Hallucination metric employs an LLM to evaluate each context in a set of contexts, determining whether there are any contradictions with the *actual_output*.

The metrics are calculated according to the following equation:

$$\text{Faithfulness} = \frac{\text{Number of Truthful Claims}}{\text{Total Number of Claims}} \quad (1)$$

the *Number of Truthful Claims* represents the count of accurate statements, and the *Total Number of Claims* represents the overall number of statements evaluated.

$$\text{Hallucination} = \frac{\text{NB of Contradicted Contexts}}{\text{Total Number of Contexts}} \quad (2)$$

5.3 RQ1: Hallucinations of RAG systems (FactRAG & KG-RAG)

Recent studies (Kamalloo et al., 2023; Tan et al., 2023) on LLM show that they are good at answering mainstream, general domain-related questions, without needing any kind of knowledge injection, such as RAG. Hence, we chose to experiment on the Financebench (Islam et al., 2023) dataset, as it contains questions on financial documents from the filings of public companies[6], which are less likely to have been seen and memorized by currently available LLM. The dataset is comprised of 150 questions and 84 documents. For each pair (question, document) in this dataset, we retrieved the 8 most relevant chunks and asked an LLM to

[6] https://www.sec.gov/search-filings

	faithfulness ↑	hallucination ↓
Llama 3.2 3B		
Text RAG	0.844	0.704
Facts RAG	**0.937**	0.679
KG-RAG	0.790	**0.660**
Llama 3.1 8B		
Text RAG	0.843	0.659
Facts RAG	**0.891**	0.658
KG-RAG	0.890	**0.532**
Qwen 2.5 32B		
Text RAG	0.954	**0.395**
Facts RAG	**0.970**	0.594
KG-RAG	0.963	0.407

Table 1: DeepEval scores with GPT4o as a judge.

generate an answer. Finally, we relied on DeepEval[7] to perform the evaluation. DeepEval relies on a *strong* LLM to automatically score RAG systems, we chose `GPT-4o` since it has been reported as being one of the most accurate. In order to evaluate the propensity to hallucinate, we report in the following the *faithfulness* measures from DeepEval, as it quantifies the consistency of the generated answer given contextual information. This is a good proxy for hallucination because we expect the LLM's response to be aligned with the retrieved chunks, and it is also the recommended way to measure hallucination. We also report the *hallucination* measure from DeepEval for completeness.

For each RAG system, we experimented with three LLMs: `Llama 3.2 3B`, `Llama 3.1 8B` (Dubey et al., 2024) and `Qwen 2.5 32B` (Team, 2024). We used these same LLM to generate facts during the facts-RAG experiments.

The results of the DeepEval evaluation are shown in Table 1. We observe an increase in faithfulness when switching from text to graph-based RAG systems, except with the smaller model. This observation fits our prior hypothesis stating that providing contextual information from KG can reduce hallucinations.

We also observe that the gap between Text and Facts RAG is higher with smaller, and supposedly less powerful, models. Since this measure quantifies the consistency between the retrieved context and the generated answer, we conclude that smaller models have some difficulties to filter out noise in raw texts, thus providing cleaner facts help them generating more appropriate answers; while larger models have better reasoning capabilities and can filter irrelevant information on their own.

[7] https://github.com/confident-ai/deepeval

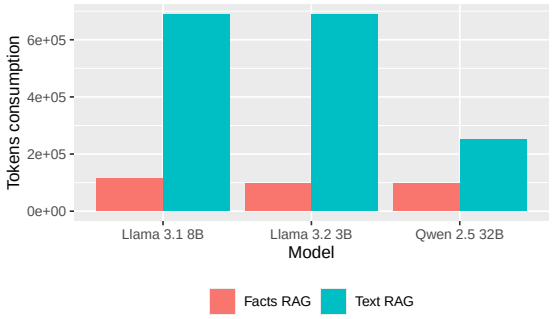

Figure 4: Total number of input tokens consumed during our experiments, for all 150 questions. Facts RAG uses dense and effective prompts while producing less hallucinations than Text RAG.

5.3.1 Ablation studies

We conducted ablation studies to measure the individual contribution of text and graph contexts. These experiments focus on our KG-RAG system, we removed either the text chunks or the triples extracted from the KG and we computed the faithfulness and hallucination measures from DeepEval . The results, shown in Table 2, shows that the faithfulness is always better when providing only triples from our KG.

We also observe that the relative differences between all setups (hybrid, no text and no graph) tend to decrease the larger the model is. This validates our previous assumption, large models can filter out irrelevant and useless information by themselves. However, we argue that letting the model do the filtering is suboptimal as it requires to provide every bits of available information to the LLM. We already showed that graph-based RAG improves the overall response by reducing hallucinations, and we show in the following that it also has the benefit of being a lot more efficient in terms of tokens consumption as illustrated in Figure 4.

5.4 RQ2: HybridRAG optimizing tokens usage in Graph-based RAG Systems

This experiment examines GraphRAG's capability to detect contradictions in regulatory language across jurisdictions, specifically between DORA (EU) and FFIEC (US) documents, demonstrating its efficiency in large-scale regulatory analysis.

Using a knowledge-graph-based KNN cluster-

	faithfulness ↑	hallucination ↓
Llama 3.2 3B		
text + graph	0.790	**0.660**
graph only	**0.940**	0.665
text only	0.807	0.690
Llama 3.1 8B		
text + graph	0.890	**0.532**
graph only	**0.965**	0.576
text only	0.866	0.592
Qwen 2.5 32B		
text + graph	0.963	0.407
graph only	**0.988**	0.619
text only	0.945	**0.365**

Table 2: DeepEval scores with GPT4o as a judge when removing text or graph contexts.

ing approach, HybridRAG minimizes token consumption by streamlining contradiction detection. Unlike traditional RAG methods that rely on costly pairwise comparisons ($O(n^2)$ complexity), HybridRAG clusters document segments with KNN ($O(k \cdot n)$ complexity), - where n being the number of chunks - reducing retrieval to targeted, contextually relevant nodes. This approach involves embedding each document node, clustering similar segments, and generating optimized LLM prompts for contradiction detection.

With this clustering method ($k = 10$), API calls decreased from almost 2 million (1 975 944 with the classical approach) to just 2 690, achieving a 734-fold reduction in token consumption. Eight potential contradictions were identified, underscoring GraphRAG's effectiveness in enhancing computational efficiency and cost-effectiveness in regulatory document retrieval.

6 Perspectives and Future Work

6.1 Limitations

This work focuses exclusively on the English language, and as such, we cannot confidently generalize our findings to other languages, even those with high resource availability.

Several limitations are associated with the DeepEval toolkit used for evaluating our RAG systems. Generally speaking, the use of LLM as a judge offers numerous advantages, such as ease of use and the ability to enable reproducible evaluations through hard-coded prompts and standard evaluation pipeline, it also presents some drawbacks. Firstly, it requires a lot of computing power and is impractical for large-scale evaluations due to high latency and potentially prohibitive costs. For instance, evaluating a single system (only 150 questions) necessitates processing approximately 4 million input tokens and 0.4 million output tokens. Secondly, the prompts utilized are often hard-coded in English, which renders the toolkit unsuitable for applications in other languages.

Lastly, even if we showed that introducing knowledge from KG enhances RAG systems, it is important to point out the difficulties of building and querying a graph that suits our target task. The underlying schema of existing KG might not fit the target use case, hence the KG often has to be either hand-crafted (extremely costly and difficult to maintain) or automatically generated (error-prone and compute-intensive). For instance, Mihindukulasooriya et al. (2023) show that precision and recall are very low even when the set of relation's types to extract is restricted.

6.2 Future work

Future work could extend graph-based RAG approaches to handle diverse datasets, including visually rich and multilingual documents, by integrating visual embeddings from Vision-Language Models (Faysse et al., 2024) and heterogeneous data (Sun et al., 2024). Fine-tuning language models with domain-specific knowledge could further reduce hallucination rates. Additionally, incorporating multimodal capabilities (e.g., text and images) could enhance contextual understanding and retrieval precision. Improving the scalability of knowledge graph construction and integrating external sources like ontologies could further reduce hallucinations (Agrawal et al., 2023). Exploring hybrid models combining symbolic reasoning with deep learning (Ambrogio et al., 2023) and advanced post-generation verification (Sansford et al., 2024) could also improve RAG systems.

6.3 Architecture design to industrialize RAG systems in production

We propose a design to smoothly deploy RAG systems in production. The architecture schema in the appendix of Figure 6 demonstrates how modular design can ensure scalability and system reliability.

The RAG logic is composed of most of the app that the user interacts with containing the RAG logic, the models hub (green) which is deployed on

its infrastructure mostly based on GPU, the Data Module (red) which consists of the Data ingestion layer of RAG, the Evaluation Module (light blue) which is responsible of all functional evaluations of the RAG and lastly one of the most important parts, the monitoring and the logging (orange), which ensure that our system works well, help with debugging, audits, updates, and gives the entire vision of the RAG. This setup allows each component to operate independently and cohesively, supporting efficient scaling, robust functionality, and clear traceability.

6.4 Lessons learned and best practices for deploying RAG systems in Production

We share some recommendations based on lessons learned in large-scale banking infrastructure.

To maintain efficient operation, several best practices must be followed when deploying RAG systems in production. First, a modular design should be implemented to allow easier maintenance, updates, and scalability (Zhang et al., 2021). Caching frequently accessed queries can help reduce latency and improve performance. Additionally, using an LLM gateway enables switching between models based on task requirements.

Real-time monitoring and logging mechanisms should track system health, latency, error rates, and performance metrics, enabling prompt issue resolution and continuous improvement (Smith and Roberts, 2022). Appropriate evaluation metrics should asses system accuracy and reliability, with faithfulness as a key metric to ensure the generated responses align with the intended outputs (Kumar et al., 2023). Finally, security measures such as input and output guardrails are necessary to maintain ethical boundaries (Patel and Gupta, 2024). Regular backups and audit logging ensure data integrity, traceability, and reproducibility across the system.

7 Conclusion

This study demonstrates three significant contributions of graph-based approaches in enhancing classical RAG systems:

1. **Reduction of Hallucinations**: The use of graph-based structures, such as Fact-RAG, significantly reduces hallucinations by linking contextually relevant information. This leads to more precise and complete responses, with experimental evaluations showing a 6% reduction in hallucinations while using 80% fewer tokens compared to text-only RAG.

2. **Efficiency and Cost Savings**: For document comparison use-case, GraphRAG improves efficiency by filtering out irrelevant data, reducing computational costs, and enhancing scalability. Using semantic clustering, it reduces the complexity of detecting contradictions from $O(n^2)$ to $O(k \cdot n)$ where n is the number of chunks and nodes in the graph.

3. **Enhanced Explainability and Traceability**: HybridRAG, using knowledge graphs, allows users to trace responses back to specific data sources and relationships as shown in Figure 5 (an example of the output of the demo using NEO4J). This transparency is crucial for sectors like finance and banking, enabling better governance, easier audits, and a more thorough understanding of the reasoning behind answers.

This efficiency demonstrates that graph-based retrieval methods can make large-scale LLM applications more cost-effective and accessible. However, their effectiveness depends on factors such as the dataset, knowledge graph modeling, and the specific retrieval task, highlighting that graph-based approaches are not always inherently more efficient.

Acknowledgments

We are grateful to Jean-Luc Billy and Philip Rathle for their insightful discussions on RAG and Knowledge Graphs, and to Virginie Chenal-Laze and Gael Marchand for sharing their domain knowledge on regulatory matters. We also acknowledge Lingua Custodia and BNP Paribas IT Group for providing the computing resources that supported the experiments on large language models (LLMs).

Reproducibility

We open-sourced the code to facilitate adoption and reproducibility of experiments.

HybridRAG Code for HybridRAG demo to compare DORA & FFIEC is available on Github [8].

FactRAG Code for both text and facts RAG used with FinanceBench Data are available on Github[9].

[8] https://github.com/halftermeyer/dora-ffiec-hybrid-rag-neo4j

[9] https://github.com/gcaillaut/financebench-graph-rag

References

Garima Agrawal, Tharindu Kumarage, Zeyad Alghamdi, and Huan Liu. 2023. Can knowledge graphs reduce hallucinations in llms?: A survey. *arXiv preprint arXiv:2311.07914*.

Confident AI. 2024. Deepeval: The evaluation framework for llms.

Mistral AI. 2023. Mixtral model card. `https://mistral.ai`.

M. Ambrogio, D. Garcia, and F. Rodriguez. 2023. Symbolic reasoning meets deep learning for improved nlp models. *Computational Intelligence Review*, 18(2):102–118.

Yejin Bang, Samuel Cahyawijaya, Bryan Wilie Lee, Wenliang Dai, Dan Su, Bryan Wilie, Pascale Fung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in neural information processing systems*.

Pere-Lluís Huguet Cabot, Simone Tedeschi, Axel-Cyrille Ngonga Ngomo, and Roberto Navigli. 2023. Redfm: a filtered and multilingual relation extraction dataset. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4326–4343.

Salvatore Carta, Alessandro Giuliani, Leonardo Piano, Alessandro Sebastian Podda, Livio Pompianu, and Sandro Gabriele Tiddia. 2023. Iterative zero-shot llm prompting for knowledge graph construction. *arXiv preprint arXiv:2307.01128*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Nouha Dziri, Alexander Milton, Tiezheng Yu, Mohit Yu, Kevin Bowden, Debanjan Ghosh, Andrea Madotto, and Pascale Fung. 2022. On the origin of hallucinations in conversational models: Is it the datasets or the models? *Transactions of the Association for Computational Linguistics (TACL)*, 10:730–746.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*.

European Parliament and Council of the European Union. 2022. Regulation (EU) 2022/2554 of the European Parliament and of the Council on Digital Operational Resilience for the Financial Sector (DORA). Accessed: 2024-11-13.

L. Faysse, J. Turner, and D. Patel. 2024. Colpali: Efficient document retrieval via vision-language models. Proceedings of the Conference on Document Retrieval.

Federal Financial Institutions Examination Council. 2019. Federal Financial Institutions Examination Council (FFIEC) IT Examination Handbook. Accessed: 2024-11-13.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. 2020. Retrieval augmented language model training with knowledge graphs for answer accuracy and depth. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, Jose Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2022. Knowledge graphs. *Synthesis Lectures on Data, Semantics, and Knowledge*, 12(2):1–257.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. Financebench: A new benchmark for financial question answering. *Preprint*, arXiv:2311.11944.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yanlin Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. A survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–35.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606.

Jihoon Kang and Minji Lee. 2023. Ever-growing knowledge: Integrating retrieval-augmented generation for continuous learning in language models. *IEEE Transactions on Neural Networks and Learning Systems*, 34(8):2591–2603.

P. Kumar et al. 2023. Metrics for evaluating large language models: Precision, recall, and faithfulness. *IEEE Transactions on AI*, 10(2):125–134.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

J. Liu, C. Xiong, and J. Callan. 2022. Graph retrieval augmentation for enhanced contextual and semantic understanding in large language models. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Shayne Longpre, Yi Lu, Mitchell Wortsman, Tianhao Xia, et al. 2021. Lost in the middle: How length of input affects language models' ability to understand context. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1001–1013.

Nandana Mihindukulasooriya, Sanju Tiwari, Carlos F Enguix, and Kusum Lata. 2023. Text2kgbench: A benchmark for ontology-driven knowledge graph generation from text. In *International Semantic Web Conference*, pages 247–265. Springer.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

N. Patel and R. Gupta. 2024. Security frameworks for ethical ai deployment. *Journal of AI Ethics*, 3(1):45–62.

H. Qian, M. Chen, Y. Liu, and S. Wang. 2024. Grounding retrieval-augmented generation: Mitigating hallucinations with contextual awareness. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Philip Rathle. The graphrag manifesto: Adding knowledge to genai.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Guy Rosin, Ranit Ben Aharon, Marina Litvak, and Daniel Cohen-Or. 2022. Knowledge graphs in natural language processing: A survey. *Journal of Artificial Intelligence Research (JAIR)*, 73:765–826.

Hannah Sansford, Nicholas Richardson, Hermina Petric Maretic, and Juba Nait Saada. 2024. Grapheval: A knowledge-graph based llm hallucination evaluation framework. *arXiv preprint arXiv:2407.10793*.

A. Smith and J. Roberts. 2022. Real-time monitoring and feedback for ai applications. *AI Systems Journal*, 8(3):148–159.

Qiang Sun, Yuanyi Luo, Wenxiao Zhang, Sirui Li, Jichunyang Li, Kai Niu, Xiangrui Kong, and Wei Liu. 2024. Docs2kg: Unified knowledge graph construction from heterogeneous documents assisted by large language models. *arXiv preprint arXiv:2406.02962*.

Yiming Tan, Dehai Min, Yu Li, Wenbo Li, Nan Hu, Yongrui Chen, and Guilin Qi. 2023. Can chatgpt replace traditional kbqa models? an in-depth analysis of the question answering performance of the gpt llm family. In *International Semantic Web Conference*, pages 348–367. Springer.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Roziere, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Yifan Yao, Zilu Zhang, Fei Fang, Min Zheng, Lei Li, Linhong Sun, and Jie Tang. 2021. Citationkg: Constructing a citation knowledge graph with contextual information and applications. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1234–1245.

M. Yasunaga, X. Wu, D. Radev, and J. Leskovec. 2022. Graph-based retrieval-augmented language models for fact verification and knowledge enrichment. In *Transactions of the Association for Computational Linguistics (TACL)*.

Bowen Zhang and Harold Soh. 2024. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. *arXiv preprint arXiv:2404.03868*.

Y. Zhang et al. 2021. Scalable architectures for large-scale ai systems. *Journal of AI Engineering*, 12(4):250–267.

W. Zhao, C. Wang, X. Zhang, and H. Lin. 2023. Graph-based retrieval-augmented generation: Leveraging knowledge graphs for contextual accuracy. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*.

A Reproducibility of HybridRAG demo

Reproducibility The code source for HybridRAG demo is open-sourced [10] to facilitate adoption.

A.1 Use-Case for Efficient Contradiction Detection in Regulatory Documents

The problem focuses on efficiently detecting contradictions between document segments, such as those from regulatory documents (DORA and FFIEC), using a Retrieval-Augmented Generation (RAG) system. The approach consists of two key steps:

1. **KNN Clustering**: A semantic similarity relation is used to group similar document segments based on cosine similarity, reducing the complexity from $O(n^2)$ to $O(k \cdot n)$ by considering only the top k-nearest neighbours for each node, where n is the number of chunks (nodes in the graph).

[10] https://github.com/halftermeyer/dora-ffiec-hybrid-rag-neo4j

2. **Contradiction Detection**: An LLM is used to detect contradictions between pairs of similar segments, reducing the number of LLM calls and the associated token consumption.

A.2 Approach: HybridRAG for Optimized Retrieval and Contradiction Identification

Traditional RAG systems often suffer from high token consumption due to the computationally expensive Cartesian product of document segments. HybridRAG addresses this by utilizing a streamlined pipeline that applies KNN clustering :

1. **Optimization of Retrieval and Contradiction Identification:** HybridRAG enhances efficiency in retrieving and identifying contradictions within large regulatory documents.
2. **Knowledge-Graph-Based KNN Clustering:** It utilizes KNN clustering with knowledge graph embeddings to group similar document segments, reducing computational costs.
3. **Context Capture:** Context for each document node is captured, incorporating its content, structural relationships, and citations.
4. **Embedding Creation:** Text embeddings are generated by concatenating contextual information, encapsulating the semantic essence of document segments.
5. **KNN Clustering:** Cosine similarity is applied to cluster document segments, creating labeled edges (e.g., `SIMILAR_TO`) for efficient comparison.
6. **Contradiction Discovery:** LLM prompts are used to assess contradictions between document segments, yielding a simple "Yes" or "No" answer.

B Architectural Design for RAG in Production

We propose a design schema in Figure 6 that demonstrates how modular design can ensure scalability and system reliability.

C Facts extraction on finance data (Islam et al., 2023)

The prompt used to extract a knowledge graph from a text is given below. The first assistant answer is forced, the others are generated by the LLM and are not reported here.

```
## User
Please read the text below, I will ask you questions afterwards.

{{ INPUT_TEXT }}

## Assistant
I have read the text, I am ready to answer your questions.

## User
The end goal is to build a knowledge graph from the text. We will do it step by step. First, extract all named entities (persons, organizations, events, ...), dates (times and epochs too) and locations. Put them in a list.

## Assistant
<list of entities>

## User
Perfect, now generate a list of triples (subject, predicate, object). Subjects and objects must come from the list of entities you extracted beforehand. Predicates are very short text (up to 3 words) describing the relation between subjects and objects. Try to extract only *interesting* triples, do not report too obvious triples.

## Assistant
<list of triples>

## User
Great, now format the triples as a JSON list. Add a "text" attribute containing a sentence in natural language fully describing the fact held by the triple. Just write the JSON content.

## Assistant
<JSON content>
```

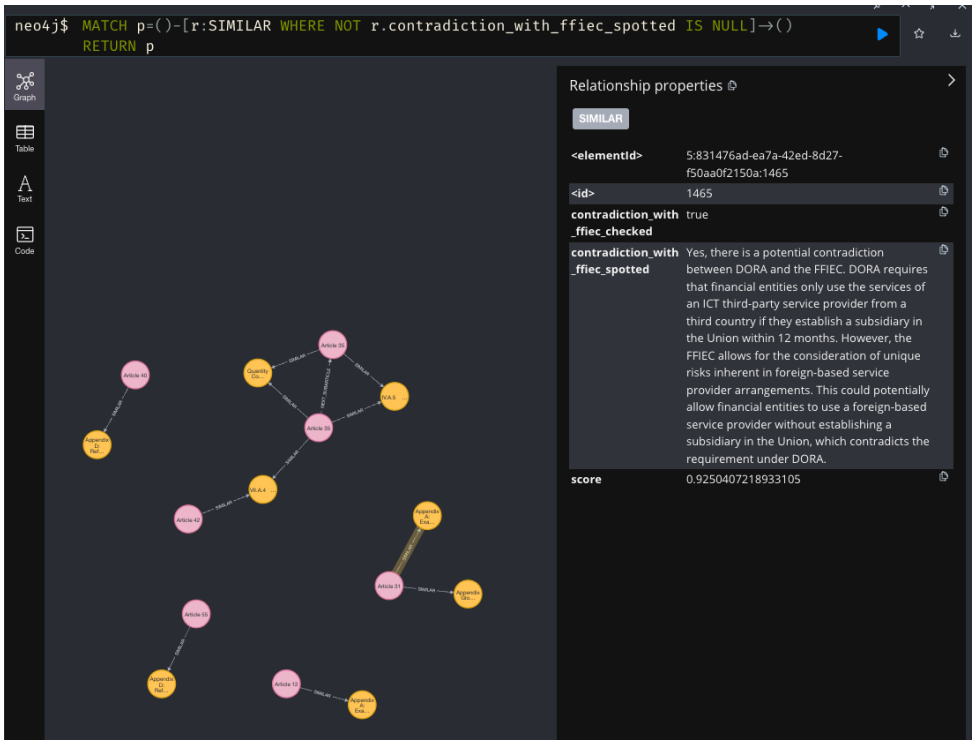

Figure 5: HybridRAG: Example of potential contradiction between articles in DORA and FFIEC

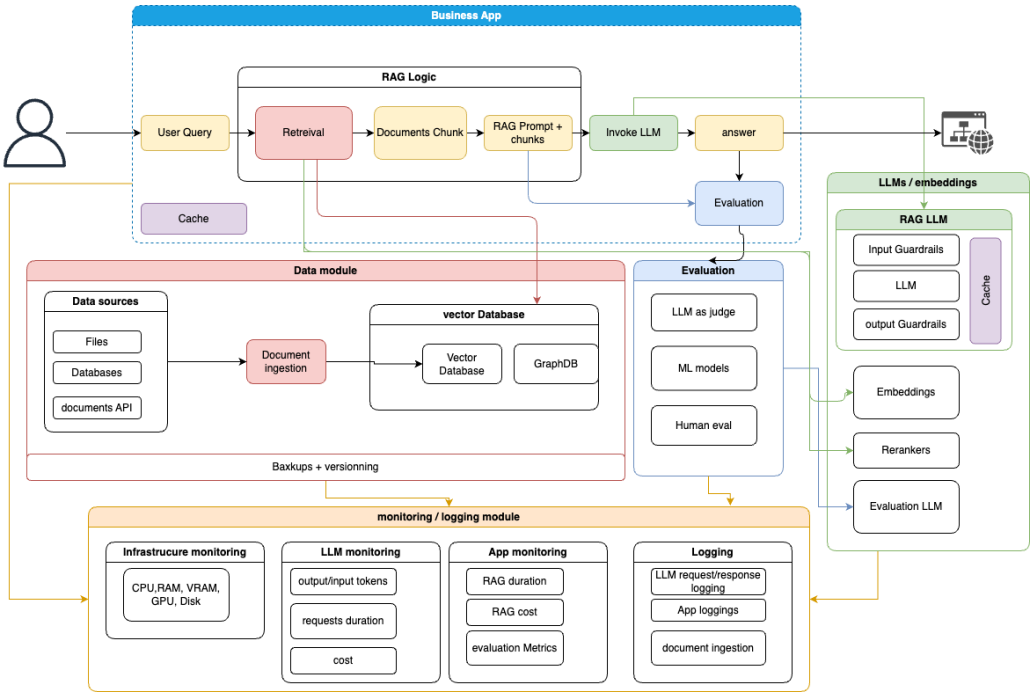

Figure 6: Architectural Design and Components to Deploy RAG in Production