Graph-Enhanced RAG: A Survey of Methods, Architectures, and Performance

Sara Sherif Daoud Saad, Stephanie Silva Vieira Gomes
Universitat Politècnica de Catalunya
Barcelona, Spain
{saracherifsaad, stephanievgomes}@gmail.com

***Abstract*—Large Language Models (LLMs) face significant challenges, including knowledge cutoffs, factual inaccuracies, and hallucinations, particularly in knowledge-intensive tasks. While Retrieval-Augmented Generation (RAG) addresses some limitations by incorporating external knowledge, traditional RAG systems rely on flat data structures that restrict complex reasoning capabilities. This survey examines the integration of Graph-Structured Knowledge (GSK) into RAG architectures to enhance factual accuracy and reduce hallucinations. GSK enables context-rich information retrieval and multi-hop reasoning by representing knowledge as interconnected entities and relationships. We analyze eight representative systems (GraphRAG, G-Retriever, GRAG, LightRAG, ToG-2, KG²RAG, KG-RAG, HippoRAG) across key architectural dimensions, including temporal integration, graph roles, and pipeline placement. These graph-enhanced approaches represent a promising direction for developing more reliable and factually grounded language models in critical domains such as finance, healthcare, and legal research. We also implement a prototype of a financial graph RAG system that illustrates the usefulness of GSK in a real-world setting in order to validate these methods. With a unique company name-to-symbol mapping mechanism, our system combines market data, financial statements, and company information into a graph structure that enhances query comprehension and data retrieval precision. This demonstrates how graph-enhanced methods can handle financial queries with accuracy and consistency.**

***Index Terms*—Graph-structured knowledge, retrieval-augmented generation, large language model**

I. Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities in understanding and generating human language for diverse Natural Language Processing (NLP) tasks, including translation, summarization, and text generation [1]. Trained on vast datasets, LLMs often produce outputs that are human-like for complex queries. However, inherent limitations persist. Specifically, LLMs suffer from a knowledge cut-off, lacking access to up-to-date factual information beyond their training data [2]. Moreover, despite their extensive training, they frequently exhibit factual inaccuracies and generate plausible but false information, known as hallucinations, particularly in knowledge-intensive or domain-specific contexts [3]. These persistent challenges highlight critical gaps in the current capabilities of LLMs, necessitating the development of effective external knowledge integration strategies.

In 2020, Lewis et al. [4] introduced the Retrieval-Augmented Generation (RAG) architecture, which optimizes LLMs' performance by dynamically retrieving and integrating external knowledge. This mechanism allows LLMs to access information beyond their initial training cutoff, significantly enhancing their ability to generate grounded and up-to-date responses. RAG has proven highly effective in mitigating core LLMs shortcomings by improving factual accuracy, reducing hallucinations, and enabling the incorporation of dynamic information updates [4], [5]. However, traditional RAG models, often relying on flat data structures and simple keyword matching for retrieval, still present inherent limitations. These include restricted capabilities for complex reasoning, retrieval of irrelevant or fragmented information, and an inability to capture the intricate relationships within a knowledge base [6].

Graph structures are recognized as an effective method for capturing relationships within data. In this representation, knowledge is organized as nodes, representing entities, and edges, representing the links or relationships between these entities. Graph-Structured Knowledge (GSK) enables the retrieval of context-rich information and more accurate data [7]. Furthermore, GSK enhances reasoning capabilities, facilitating the capture of more relevant and complex data. Consequently, recent studies have leveraged GSK to enhance RAG models and, by extension, LLMs, capitalizing on the powerful benefits that graph structures offer for knowledge-intensive tasks [7].

This paper surveys recent research on integrating GSK into RAG models, addressing the persistent challenge of ensuring factual accuracy and mitigating hallucinations in LLMs' outputs, particularly in knowledge-intensive tasks. The proposed solution leverages graph-structured knowledge to improve RAG models, thereby enhancing LLMs' ability to generate factually consistent and reliable responses. This work reviews current methodologies, examines key architectural innovations and their applications, and identifies remaining challenges and promising future directions in this emerging field.

The remainder of this paper is organized as follows: Section II provides essential background on RAG and GSK. Section III delves into the various methods for integrating GSK into RAG architectures. Section IV compares current key approaches focusing on their impact on enhancing factual accuracy and mitigating hallucinations. Section V exemplifies how graph-enhanced RAG works in a real-life context. Section VI identifies current challenges and outlines promising future research directions in this field. Finally, Section VII concludes the paper by summarizing key findings and contributions.

II. Background and Preliminaries

A. Retrieval-Augmented Generation (RAG)

1) Definition: RAG is a retrieval method that utilizes external sources of knowledge to ground a model's output, improving its relevancy, accuracy, and usefulness [4]. Its core characteristic is the integration of real-time or domain-specific data to augment the LLM's generation process, complementing its inherent parametric memory (the model's capability of storing knowledge in its neural network). RAG operates by combining both parametric memory and non-parametric memory (the ability to store information in an external database, retrieved when needed). It first accesses relevant information from the non-parametric memory, then parses this knowledge for the model to utilize alongside its parametric memory.

2) Architecture and Workflow: RAG architecture can be defined by four main components: ingestion, retrieval, augmentation, and generation.

- **Ingestion:** This initial phase involves preparing the external knowledge base for efficient retrieval. It typically includes collecting raw data, cleaning and pre-processing documents, segmenting them into manageable chunks (e.g., passages or documents), and then indexing these chunks, often by creating vector embeddings for fast similarity search [8].
- **Retrieval:** Given a user query, this component is responsible for searching the indexed external knowledge base to identify and retrieve the most relevant documents, passages, or knowledge snippets. This is commonly performed using semantic search techniques based on vector similarity [8].
- **Augmentation:** In this step, the retrieved context from the non-parametric memory is integrated into the input prompt of the Large Language Model. The retrieved information, along with the original user query, is effectively combined to provide the LLM with external knowledge to ground its response [8].
- **Generation:** This is the final stage where the LLM processes the augmented prompt (containing both the user query and the retrieved context) to synthesize a coherent, accurate, and contextually relevant output [8].

Figure 1 illustrates the general workflow from a question-answer context where the LLM model does not have sufficient access to domain-specific information, and therefore, the use of RAG assists in the retrieval of external knowledge to enhance the output. This is an example of a Naive RAG.

RAG has evolved significantly over the past years, leading to three main paradigms: Naive RAG, Advanced RAG, and Modular RAG. Figure 2 illustrates these RAG paradigms.

- Naive RAG: represents the earliest methodology, following a traditional "Retrieve-Read" process that includes indexing, retrieval, and generation [8].
- Advanced RAG: builds upon Naive RAG by introducing specific improvements, primarily focusing on enhancing retrieval quality through pre-retrieval and post-retrieval strategies to overcome its limitations [8].

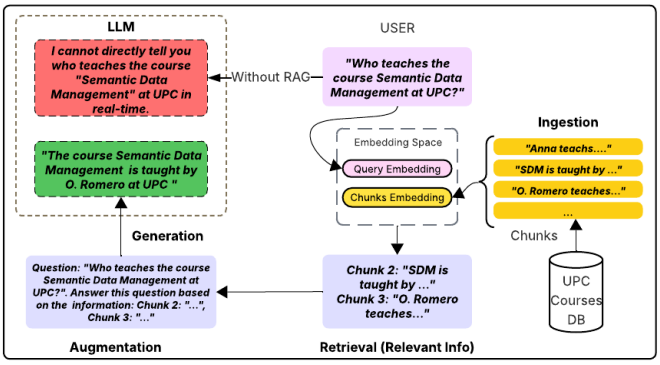

Fig. 1: Conceptual Example of Retrieval-Augmented Generation Workflow

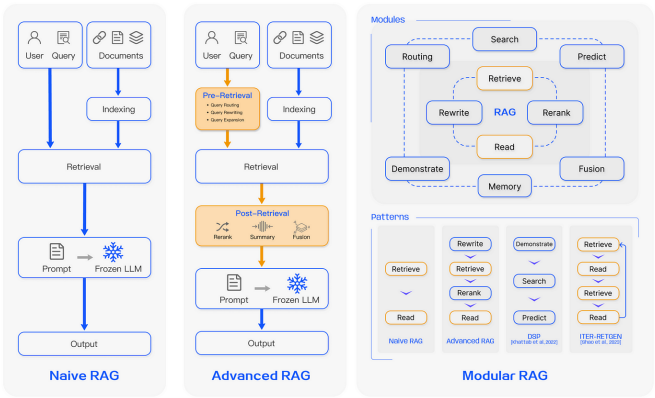

Fig. 2: Three Paradigms of Retrieval-Augmented Generation [8]

- Modular RAG: introduces multiple specific functional modules and allows for the replacement or rearrangement of existing ones, supporting processes that are not limited to sequential retrieval and generation but can include iterative and adaptive retrieval [8].

B. Graph-Structured Knowledge (GSK)

GSK encompasses any data that can be represented by entities (nodes) and their relationships (edges). This includes structures such as knowledge graphs (KGs) and property graphs, as well as more flexible graph representations derived from textual content (textual graphs) and graph transformations of structured sources, such as relational databases [9], [10].

The primary advantage of GSK lies in its ability to explicitly represent complex relationships, enabling richer semantic understanding and facilitating multi-hop reasoning that is challenging for flat data structures. GSK provides context-rich information, making facts explicit and traceable, which significantly enhances the accuracy and interpretability of retrieved knowledge for downstream tasks like question answering and recommendation systems [9]–[11].

Notable examples of GSK include large-scale, general-purpose knowledge bases such as Wikidata, and DBpedia,

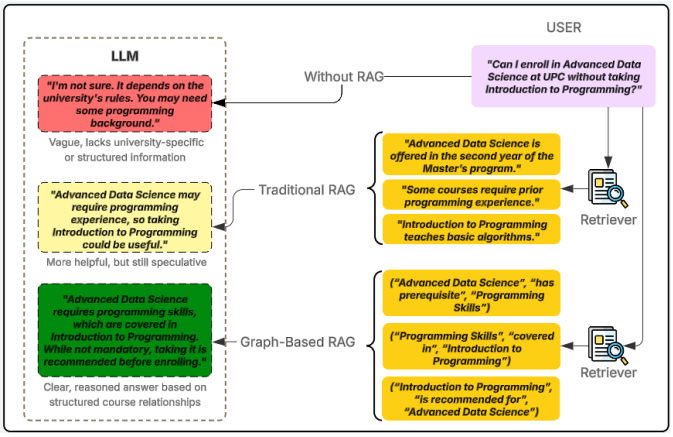

Fig. 3: Different Approaches to Answer a Question using an LLM

which integrate a wide spectrum of encyclopedic facts, as well as domain-specific knowledge graphs constructed for specialized applications in biomedical research (e.g., Bio2RDF), finance, or legal informatics [12]–[14].

C. The Challenge of Hallucinations and Factual Accuracy

Large Language Models frequently exhibit hallucinations, generating plausible but incorrect or fabricated information. A primary reason is their dependency on static training data, leading to a knowledge cut-off that prevents access to up-to-date information. When queried beyond their training scope, LLMs may default to manufacturing answers that are semantically coherent but factually false [3]. Furthermore, despite extensive training on vast datasets, LLMs can encounter inconsistencies within their diverse training data, leading to contradictory internal knowledge. This tendency to prioritize fluency and coherence over strict factual correctness contributes significantly to the problem of hallucinations [3].

Factual accuracy is of extreme importance, particularly in knowledge-intensive NLP tasks. In critical domains such as healthcare, finance, and legal research, the generation of misinformation by LLMs can have severe consequences, ranging from incorrect diagnoses and misguided financial advice to propagation of false claims [15], [16]. For applications like question answering, summarization, and report generation, factual correctness is not only a quality metric but an indispensable requirement for the utility, safety, and ethical deployment of LLMs [17].

III. Methods for Integrating Graph-Structured Knowledge into RAG

In the literature, there are plenty of methods to integrate GSK into RAG, but Figure 3 describes the overall idea and highlights the limitations of both LLM and traditional RAG, as they fall short in domain-specific knowledge and lack reasoning for complex relationships [7].

A. Architecture Overview

As much as there are vast ways to implement the architecture of a graph-enhanced RAG, there are critical points that fundamentally shape the capabilities and performance of these methods. This section will enlighten the understanding of the architectural choices usually encountered in the key works being developed in literature right now.

1) Static vs. dynamic graph integration: The temporal nature of the GSK integration, whether the graph structure is static or dynamic. Static graph integration involves utilizing a fixed, pre-indexed snapshot of knowledge. This approach offers simplicity in implementation and predictability in retrieval, as the graph's structure and content remain constant during runtime.

In contrast, dynamic graph structure integration enables the graph to be updated in real-time or through continuous learning processes. This allows the RAG system to access the most current information and adapt to evolving knowledge domains, supporting continual learning capabilities.

2) Graph-as-Context vs. Graph-as-Reasoner: The role the graph structure plays within the RAG pipeline. It can act primarily as a source of context or as an active reasoning engine. When functioning as Graph-as-Context, the GSK primarily serves as a repository from which relevant facts, triplets, or subgraphs are extracted.

On the other hand, when operating as Graph-as-Reasoner, the GSK actively participates in performing complex logical inferences, multi-hop analysis, or deriving conclusions directly from its structured relationships. The graph structure itself guides or executes parts of the "thinking" process, and the results of this reasoning are then provided to the LLM [18].

3) Placement of GSK Integration within the RAG Pipeline: The specific stage(s) where GSK is integrated into the RAG workflow. There might be multiple stages implemented for the same method.

- Pre-Retrieval (Query Enhancement): Here, the graph structure is leveraged to enrich, expand, or reformulate the user's input query before it proceeds to the main retrieval phase. This can involve adding related entities or concepts to improve search relevance.
- During Retrieval (Graph-based Retrieval): In this configuration, the graph structure serves as the primary data source for retrieval. Graph traversal algorithms, graph embeddings, or graph pattern matching are directly applied to the GSK to identify and extract relevant structured information.
- Post-Retrieval (Contextualization and Reranking): In this stage, the graph structure is used to further process, contextualize, or re-rank initially retrieved text chunks (which may or may not have been retrieved from a GSK initially).
- During Generation (Graph-Aware Generation): At this point, the graph's structured information or the outcomes of graph-based reasoning are directly fed into or guide the LLM's generation process. This can involve the LLM

becoming "graph-aware" through specific prompting or even fine-tuning.
- Iterative/Hybrid Integration: More advanced architectures integrate the GSK across multiple stages or through iterative feedback loops. In these systems, an LLM's intermediate output or reasoning step might trigger further graph-based retrieval or refinement, creating a dynamic and interactive flow of knowledge.

B. Technical Components

To illustrate the key technical approaches employed in graph-augmented RAG, this section details how structured graph representations address the limitations of conventional RAG approaches.

1) Graph Construction Approaches: One of the fundamental advantages of GSK over flat documents is that it enables more nuanced representation of relationships and patterns. Therefore, one of the core aspects of graph-augmented RAG is how to represent data in the GSK format.

A straightforward method is the direct transformation of textual data into structured graph representations, wherein entities and relationships are systematically extracted and mapped to nodes and edges, respectively. This methodology involves the application of established text mining techniques and natural language processing tools to identify and extract semantic relationships from unstructured text, subsequently organizing this information into a coherent graph-based knowledge structure [19], [20]. Notably, the utilization of LLMs in these processes can enhance the quality and scalability of the generated graph structures, due to the LLMs' capabilities of reasoning [21].

2) Enhanced Retrieval Strategies: Retrieving relevant information from the GSK in an efficient manner is not an easy task, as graph structures can have complex connections and large sizes. Nonetheless, some strategies implement sophisticated navigation and selection mechanisms to traverse and identify the most important information within these complex networks.

- Graph Transversal and Community Detection: one of the approaches is graph transversal algorithms, the backbone of retrieval strategies. Advanced systems employ algorithms to convert textual graph structures into hierarchical descriptions [22], [23]. Community detection algorithms partition large-scale graphs into hierarchical communities of related entities, guiding subsequent summarization and retrieval processes [24].
- Multi-hop Reasoning Strategies: enables systems to discover complex relationships spanning multiple nodes and edges [25]. These strategies use path-finding algorithms that traverse multiple hops to uncover indirect connections between entities.
- Subgraph Extraction and Ranking: focus on identifying and prioritizing relevant graph portions based on query context and semantic similarity. These approaches utilize scoring functions considering both structural properties (centrality measures, connectivity patterns) and semantic features (entity embeddings, relationship types) [26].

3) Knowledge Integration and Generation: To effectively use knowledge from graph structures, it is crucial to transform this information into formats suitable for LLM processing and answer generation. The preservation of structural information during this transformation represents a fundamental step in graph-augmented RAG architectures. Some of the techniques used for enabling the smoothness of this process include:

- Graph Linearization: basic approaches employ graph flattening techniques that transform retrieved graphs/subgraphs into textual formats by systematically representing node and edge attributes in sequential form. Advanced linearization methods convert complex graph topologies into hierarchical text descriptions that serve as structured prompts, effectively preserving topological information in textual representations [27].
- Context Integration: Other than graph linearization, alternative strategies utilize multi-modal integration frameworks that maintain separate representations for textual and structural information, allowing models to process both semantic content and topological features simultaneously [4].

IV. Analysis of Methods

A. Representative Systems Analysis

Since the initial RAG model in 2020 [4], numerous other systems have been developed, aiming to address the shortcomings of the traditional RAG model already mentioned. In this paper, eight graph-enhanced RAG models were selected based on their relevance and innovation value. Details of their architecture, strengths, and performance will be provided. Table IV presents a concise overview of the system's architecture.

- GraphRAG [28]: uses a static global strategy where the graph acts as a contextual framework to support thorough document understanding. Its architecture centers on hierarchical community detection using the Leiden algorithm, creating multi-level summaries that enable "sensemaking" queries requiring global document comprehension. The system's strength lies in its ability to provide comprehensive, diverse responses for complex analytical queries, while its weakness is the high computational cost and static nature that requires full reconstruction for updates.
- G-Retriever [29]: implements a flexible framework tailored for textual graph question answering. Its architecture standardizes textual graphs and leverages Graph Attention Networks (GAT) combined with Multilayer-Perceptron (MLP)-based alignment for effective node selection and subgraph extraction. The system demonstrates strong benchmark performance across various graph datasets, consistently outperforming traditional approaches. However, its limitations include reliance on static graph processing and the need for extensive preprocessing to standardize input graphs.

- GRAG [30]: introduces a dynamic approach where graphs function as reasoning engines rather than mere context providers. Its architecture incorporates dual-view frameworks combining hard prompts for hierarchical reasoning and soft prompts for GNN-based processing. The system's strength is in multi-hop reasoning and hallucination reduction, though it faces complexity challenges in balancing the dual reasoning paradigms.
- LightRAG [31]: represents the most recent advancement, focusing on efficiency and adaptability. Its architecture implements dual-level retrieval with incremental update capabilities, using LLM-powered extraction with key-value profiling and deduplication mechanisms. The system's primary strength is computational efficiency and dynamic adaptability, while maintaining competitive performance across diverse datasets.
- Think-on-Graph 2.0 (ToG-2) [32]: creates a hybrid RAG framework that tightly connects unstructured and structured knowledge sources. The architecture includes knowledge graph-guided context retrieval and iterative retrieval from text corpora and knowledge graphs. With its deep and faithful reasoning capabilities, this plug-and-play, training-free system raises smaller models like LLAMA-2-13B to GPT-3.5 performance levels while achieving state-of-the-art performance on six of seven knowledge-intensive datasets.
- KG²RAG [33]: makes use of knowledge graphs to enhance the coherence and diversity of results obtained by establishing fact-level relationships between chunks. Through structured relationship mapping, the system provides improved retrieval quality by implementing a KG-based chunk organization and KG-guided chunk expansion process. Although it has issues with semantic retrieval seed dependency and KG construction overhead, it performs better on HotpotQA variants with better response quality and coherence.
- KG-RAG [34]: integrates knowledge graphs to build a bridge between structured knowledge and creative generation. By integrating structured knowledge into creative generation processes, the architecture emphasizes the balance between knowledge and creativity. The system tackles the problem of preserving factual accuracy while allowing for creative output, but it has limitations specific to a given domain and evaluation issues when it comes to gauging the balance between creativity and knowledge.
- HippoRAG [35]: uses a memory system based on the hippocampus indexing theory that is inspired by neurobiology. For non-parametric learning and ongoing knowledge integration, the architecture makes use of personalized PageRank. Despite memory overhead and graph construction costs, this method is 6–13 times faster and 10–30 times more cost-effective than iterative methods, allowing for continuous learning with up to 20% performance improvement.

B. Comparative Performance Assessment

1) System Capabilities and Improvements: GraphRAG demonstrates strong performance on global understanding queries, achieving 72–83% win rates in comprehensiveness through hierarchical community detection [28]. G-Retriever introduces the first retrieval-augmented generation (RAG) framework for textual graphs, showing a 40.6% improvement over prompt tuning via GAT-based processing and reducing token usage by 99% [29]. GRAG extends this approach with a dual-view framework, achieving a 74.45% gain on WebQSP by enhancing multi-hop reasoning [30]. LightRAG emphasizes efficiency, requiring fewer than 100 tokens versus GraphRAG's 610.000 while maintaining competitive performance [31].

ToG-2 demonstrates the efficacy of hybrid structured-unstructured knowledge coupling by extending global understanding capabilities with state-of-the-art performance on six of seven knowledge-intensive datasets. [32]. Through fact-level relationships, KG²RAG enhances retrieval diversity and coherence, demonstrating superior performance on HotpotQA variants. [33]. Knowledge-creativity balance in generation tasks is specifically addressed by KG-RAG [34]. HippoRAG maintains up to 20% performance improvement while achieving remarkable efficiency gains, being 6–13 times faster and 10–30 times more cost-effective than iterative methods. [35].

2) Hallucination Reduction Effectiveness: G-Retriever achieves a 54% reduction in hallucinations, with 77% valid nodes and 62% fully valid graphs compared to the baseline's 31% and 8%, respectively [29]. GRAG further improves entity validation with 79% valid entities [30]. While GraphRAG and LightRAG do not explicitly report hallucination metrics, their graph-based architectures inherently promote factual consistency via structured validation and community-level summarization [28], [31].

KG-RAG lowers hallucination rates from 30% to 15%, demonstrating how well dynamic knowledge graphs work to improve answer faithfulness. Most remarkably, Think-on-Graph 2.0 (ToG-2) uses structured reasoning and iterative KG-guided retrieval to drastically reduce the hallucination rate from 72.7% with GPT-3.5 + CoT-SC to just 9.7%.

TABLE I: Hallucination Reduction Comparison

System	Valid Entities	Valid Nodes	Valid Graphs	Hallucination Rate
G-Retriever	71%	77%	62%	38%
GRAG	79%	–	–	–
KG-RAG	–	–	–	15%
ToG-2	–	–	–	9.7%
Baseline LLM	62%	31%	8%	92%

3) Factual Accuracy Improvements: G-Retriever achieves 70.49 Hit@1 on WebQSP, reflecting a 45.81% improvement over prompt tuning, and 85.16% on ExplaGraphs [29]. GRAG surpasses this with 72.36 Hit@1 and 92.23% accuracy, showcasing its multi-hop reasoning strength [30]. GraphRAG records 72–83% win rates in comprehensiveness across podcast and news datasets [28], while LightRAG achieves an

84.8% win rate on legal domains with significantly reduced computational load [31].

ToG-2 demonstrates state-of-the-art performance across 6 out of 7 knowledge-intensive datasets, elevating smaller models like LLAMA-2-13B to GPT-3.5 performance levels [32]. HippoRAG shows up to 20% performance improvement on multi-hop QA datasets including MuSiQue, 2WikiMultiHopQA, HotpotQA, and Bamboogle [35]. KG²RAG demonstrates superior performance on HotpotQA variants with enhanced diversity and coherence metrics [33].

TABLE II: Factual Accuracy Performance

System	Metric(s)	Highlights
G-Retriever	WebQSP: 70.5 ExplaGraphs: 85.2	Baseline
GRAG	WebQSP: 72.4 ExplaGraphs: 92.2	+2.6% WebQSP, +8.3% ExplaGraphs, +8pp valid refs
GraphRAG	Win Rate: 72–83%	Strong multi-hop, comprehensive
LightRAG	Win Rate: 84.8%	+5.6pp legal win, 99.98% token ↓, incremental
ToG-2	SOTA on 6/7 sets	Boosts small models
HippoRAG	+20% (multi-hop)	Strong QA boosts
KG²RAG	HotpotQA SOTA	High factuality, diverse answers

4) Head-to-Head System Comparisons: The GRAG vs. G-Retriever comparison reveals GRAG's superior performance: 72.36 vs. 70.49 Hit@1 on WebQSP, 92.23% vs. 85.16% ExplaGraphs accuracy, and an 8pp advantage in valid entity reference (79% vs. 71%) [29], [30].

In comparing GraphRAG and LightRAG, the latter demonstrates practical superiority. While GraphRAG focuses on sensemaking through deep graph summarization, LightRAG achieves a 52.8% overall win rate and 73.6% diversity win rate on legal datasets, requiring fewer than 100 tokens for retrieval compared to GraphRAG's 47.2% overall win rate, 26.4% diversity win rate, and 610.000 tokens [28], [31]. Similarly, KG2RAG performs better than GraphRAG in terms of retrieval efficiency and factual accuracy. In the fullwiki context, KG2RAG outperforms GraphRAG by +6.4 F1 scores on the Shuffle-HotpotQA benchmark. Additionally, it offers improved recall and retrieval precision while surviving KG quality degradation. Because KG2RAG uses fact-level structural reasoning with lower token and time costs than GraphRAG's summarization-heavy architecture, it is better suited for scalable, knowledge-intensive tasks [28], [33].

TABLE III: Head-to-Head System Comparisons

Comparison	Winner	Key Advantages
G-Retriever vs. GRAG	GRAG	+2.65% Hit@1 on WebQSP; +8.3% ExplaGraphs accuracy; +8pp valid entity reference improvement.
GraphRAG vs. LightRAG	LightRAG	+5.6pp win rate on legal dataset; +47.2pp diversity win rate; 99.98% token reduction in retrieval; supports incremental updates.
KG2RAG vs. GraphRAG	KG2RAG	+6.4 F1 on Shuffle-HotpotQA; better retrieval precision/recall; robust to incomplete KGs.

These results trace a clear evolution from G-Retriever to GRAG to LightRAG and KG2RAG, reflecting continuous gains in reasoning quality and computational efficiency. Future RAG systems are likely to further optimize this trade-off between performance and scalability.

V. Experimentation

We created a financial-focused prototype to verify the efficacy of incorporating Graph-Structured Knowledge into RAG systems. The prototype can be accessed via the link: GitHub Repository. The following crucial procedures and elements were part of our experiment:

A. System Overview

We developed a financial graph-based RAG system that integrates sophisticated retrieval and generation capabilities with structured financial data. The prototype stores and links market data, financial statements (income, balance sheet, and cash flow), and company information using a Neo4j graph database. This graph-based method facilitates multi-hop reasoning over interconnected financial entities and allows for context-rich retrieval.

B. Key Features and Implementation

- **Company Name-to-Symbol Mapping:**
  We put in place a mapping mechanism that automatically converts company names in user queries to their corresponding stock symbols in order to close the gap between structured data and natural language queries. Because financial datasets are usually indexed by symbol, this guarantees accurate retrieval.
- **Graph Data Integration:**
  To represent entities (companies) and their relationships (e.g., `reported`, `has_statement`, `traded_as`), we ingested a number of financial datasets (company information, income statements, balance sheets, cash flows, and stock prices) into the Neo4j graph.
- **Retrieval-Augmented Generation Pipeline:**
  The system retrieves pertinent context for a given query by using semantic search (through FAISS) over graph-extracted documents. Following retrieval, the data is fed into a Large Language Model (LLM) to generate responses, improving factual accuracy and lowering hallucinations.
- **Interactive Interfaces:**
  Users can ask natural language questions about businesses, financial metrics, and trends using the prototype's support for both a command-line interface and a Streamlit web application.

C. Experimentation Process

We tested the system with a variety of financial queries, such as:

- *Which companies operate in the technology sector?*
- *What is the net income of Microsoft Corporation in 2024?*
- *What does Microsoft do?*

The system produced logical, fact-based responses, successfully mapped company names to symbols, and extracted accurate financial data from the graph.

D. How the System Reduces Hallucinations

1) **Grounded Retrieval:**
   The structured graph, which is constructed from validated financial data sources, is the only source of information that the system retrieves. Instead of producing responses solely from its own (possibly out-of-date or insufficient) training data, this guarantees that the LLM is given actual, context-rich data to base its responses on.
2) **No Data, No Answer:**
   The system does not create an answer if the graph does not contain the pertinent information (for example, a company or a particular financial metric is missing). Rather, it informs the user that the requested information is not available by returning a message like "No relevant information found in the context."
3) **Explicit Context for Generation:**
   The graph provides explicit context that is used to prompt the LLM. It is not allowed to fabricate facts or hallucinate details because it is only told to respond based on this context.
4) **Traceable Sources:**
   Users can confirm the response's factual foundation by viewing the source documents or data snippets that were used to generate it.

Summary: *By tightly tying answer generation to the graph, the prototype lessens hallucinations. If the necessary data is missing, the system alerts the user clearly and concisely rather than attempting to create an answer.*

VI. Challenges and Future Directions

Although LLM limitations like hallucinations and forgetfulness are addressed by GSK-enhanced RAG models [32]–[35], significant obstacles still exist.

A. Principal Difficulties

Scalability: GSK construction from raw text is slow and expensive (e.g., KG-RAG has a latency of 30s, reduced to $\sim$3s with better hardware) [34]. Offline indexing further limits scalability [33].

Data Integrity and Quality: Poor input quality and errors in triple extraction degrade performance [34]. Integrating GSK with unstructured text remains complex [32]. GSK incompleteness and NER/OpenIE limitations reduce coverage [32], [33].

Gaps in Evaluation: Benchmarks like HotpotQA suffer from spurious signals [34]. EM metrics fail to capture semantic equivalence [32], [34], and domain-specific, unseen datasets remain scarce [32], [34].

B. Prospects for the Future

Technical Advancements: Efficient, local GSK construction with open-source LLMs (e.g., Llama-3-70B) is becoming viable [34]. Further gains depend on hybrid retrieval frameworks and better traversal algorithms [32], [35], plus component-level tuning of NER, OpenIE, and ranking modules [35].

New Paradigms: Insights from hippocampal indexing and the "extended mind" model could reshape RAG design [34], [35]. Adaptive query strategies and modular, plug-and-play architectures are equally promising [32], [33].

Deployment Focus: Cheaper LLMs and hardware make real-time RAG feasible [34]. Domain-specific applications (e.g., finance) underscore practical value [32], while robustness to GSK quality remains essential for production use [33].

These directions aim to deliver scalable, accurate, and efficient RAG systems ready for complex real-world tasks.

VII. Conclusion

According to this survey, incorporating Graph-Structured Knowledge into RAG architectures is a significant step forward in the fight against factual errors and LLM hallucinations. Consistent gains over conventional RAG approaches are found through analysis of eight representative systems, including up to 79% valid entity reference (GRAG), a 54% decrease in hallucinations (G-Retriever), and 20% performance gains (HippoRAG). Graph-as-Context versus Graph-as-Reasoner paradigms and static versus dynamic integration are two examples of architectural innovations that allow multi-hop reasoning and traceable information retrieval, which are not possible with flat data structures. Our financial domain prototype validates these theoretical advancements by integrating market data, financial statements, and company information into a Neo4j graph structure and putting in place a company name-to-symbol mapping mechanism, . The prototype specifically prevents hallucinations by basing answers on validated data sources and returning "no information found" messages rather than fabricating answers when relevant data is unavailable, in addition to more accurately answering financial queries. Although there are still problems with scalability and real-time performance, graph-enhanced RAG systems offer a promising path toward reliable, fact-based AI systems required for critical domains where accuracy is essential.

REFERENCES

[1] M. Raza, Z. Jahangir, M. B. Riaz, and et al, "Industrial applications of large language models," *Nature Scientific Reports*, vol. 15, no. 1, pp. 1–15, 2025. [Online]. Available: https://doi.org/10.1038/s41598-025-98483-1

[2] J. Cheng, M. Marone, O. Weller, D. Lawrie, D. Khashabi, and B. Van Durme, "Dated data: Tracing knowledge cutoffs in large language models," *arXiv preprint arXiv:2403.12958*, 2024.

[3] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Trans. Inf. Syst.*, vol. 43, no. 2, p. 55, 2025. [Online]. Available: https://doi.org/10.1145/3703155

[4] 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.

[5] N. B. Team, "What is retrieval-augmented generation, aka rag?" *NVIDIA Blog*, August 2024, *(Supporting RAG effectiveness in mitigating LLM limitations)*. [Online]. Available: https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/

[6] J. Chen, H. Lin, X. Han, and L. Sun, "Benchmarking large language models in retrieval-augmented generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, 2024, pp. 17 754–17 762.

[7] Z. Zhu, T. Huang, K. Wang, J. Ye, X. Chen, and S. Luo, "Graph-based approaches and functionalities in retrieval-augmented generation: A comprehensive survey," *arXiv preprint arXiv:2504.10499*, 2025.

[8] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, vol. 2, no. 1, 2023.

[9] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier *et al.*, "Knowledge graphs," *ACM Computing Surveys (Csur)*, vol. 54, no. 4, pp. 1–37, 2021.

[10] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 2, pp. 494–514, 2021.

[11] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web*, vol. 8, no. 3, pp. 489–508, 2016.

[12] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[13] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer *et al.*, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic web*, vol. 6, no. 2, pp. 167–195, 2015.

[14] M. Dumontier, A. Callahan, J. Cruz-Toledo, P. Ansell, V. Emonet, F. Belleau, and A. Droit, "Bio2rdf release 3: a larger connected network of linked data for the life sciences," in *Proceedings of the 2014 international conference on posters & demonstrations track*, vol. 1272. Citeseer, 2014, pp. 401–404.

[15] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM computing surveys*, vol. 55, no. 12, pp. 1–38, 2023.

[16] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, "On faithfulness and factuality in abstractive summarization," in *Proceedings of the 58th*

[19] S. Kumar, "A survey of deep learning methods for relation extraction," *arXiv preprint arXiv:1705.03645*, 2017.

Annual Meeting of the Association for Computational Linguistics, 2020, pp. 1906–1919.

[17] C. Wang, X. Liu, Y. Yue, X. Tang, T. Zhang, C. Jiayang, Y. Yao, W. Gao, X. Hu, Z. Qi *et al.*, "Survey on factuality in large language models: Knowledge, retrieval and domain-specificity," *arXiv preprint arXiv:2310.07521*, 2023.

[18] X. Liu, D. Liu, B. Yang, H. Zhang, J. Ding, W. Yao, W. Luo, H. Zhang, and J. Su, "Kgr4: Retrieval, retrospect, refine and rethink for commonsense generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, 2022, pp. 11 029–11 037.

[20] T. Al-Moslmi, M. G. Ocaña, A. L. Opdahl, and C. Veres, "Named entity extraction for knowledge graphs: A literature overview," *IEEE Access*, vol. 8, pp. 32 862–32 881, 2020.

[21] X. Wei, X. Cui, N. Cheng, X. Wang, X. Zhang, S. Huang, P. Xie, J. Xu, Y. Chen, M. Zhang *et al.*, "Chatie: Zero-shot information extraction via chatting with chatgpt," *arXiv preprint arXiv:2302.10205*, 2023.

[22] L. Backstrom and J. Leskovec, "Supervised random walks: predicting and recommending links in social networks," in *Proceedings of the fourth ACM international conference on Web search and data mining*, 2011, pp. 635–644.

[23] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[24] S. Fortunato, "Community detection in graphs," *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[25] X. V. Lin, R. Socher, and C. Xiong, "Multi-hop knowledge graph reasoning with reward shaping," *arXiv preprint arXiv:1808.10568*, 2018.

[26] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[27] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Reasoning on graphs: Faithful and interpretable large language model reasoning," *arXiv preprint arXiv:2310.01061*, 2023.

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

[29] X. He, Y. Tian, Y. Sun, N. Chawla, T. Laurent, Y. LeCun, X. Bresson, and B. Hooi, "G-retriever: Retrieval-augmented generation for textual graph understanding and question answering," *Advances in Neural Information Processing Systems*, vol. 37, pp. 132 876–132 907, 2024.

[30] Y. Hu, Z. Lei, Z. Zhang, B. Pan, C. Ling, and L. Zhao, "Grag: Graph retrieval-augmented generation," *arXiv preprint arXiv:2405.16506*, 2024.

[31] Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, "Lightrag: Simple and fast retrieval-augmented generation," 2024.

[32] S. Ma *et al.*, "Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation," *arXiv preprint arXiv:2407.10805*, 2024.

[33] W. Hu *et al.*, "Knowledge graph-guided retrieval augmented generation," *arXiv preprint arXiv:2502.06864*, 2025.

[34] S. Fang, K. Ma, T. Zheng, X. Du, N. Lu, G. Zhang, and Q. Su, "Kg-rag: Bridging the gap between knowledge and creativity," *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024.

[35] B. J. Gutiérrez, Y. Shu, and Y. Su, "Hipporag: Neurobiologically inspired long-term memory for large language models," *Advances in Neural Information Processing Systems*, 2024.

TABLE IV: Comparative Analysis of Graph-Augmented RAG System Architectures

System	Temporal Integration	Graph Role	Pipeline Integration	Graph Construction	Retrieval Strategy	Knowledge Integration
GraphRAG	Static	Graph-as-Context	• Graph-based retrieval • Graph-aware generation	• LLM-driven entity/relationship extraction • Hierarchical community detection (Leiden) • Pre-generated summaries	• Map-reduce over communities • Global sensemaking aggregation • Community-based synthesis	• Multi-level summaries • Map-reduce composition • Hierarchical aggregation
G-Retriever	Static	Graph-as-Context	• Graph-based retrieval • Graph-aware generation	• Textual graph standardization • Node/edge preservation • Benchmark integration	• k-NN node selection • PCST optimization • Connected subgraphs	• GAT encoding • MLP alignment • Soft prompting
GRAG	Dynamic	Graph-as-Reasoner	• Graph-based retrieval • Post-retrieval reranking • Graph-aware generation	• PLM-based embedding • K-hop ego-graphs • Mean-pooling aggregation	• Divide-and-conquer strategy • Graph soft pruning • Multi-hop reasoning	• Dual-view framework • Hard prompts (hierarchical) • Soft prompts (GNN)
LightRAG	Dynamic	Graph-as-Context	• Graph-based retrieval • Graph-aware generation	• LLM-powered extraction • Key-value profiling • Deduplication mechanisms	• Dual-level paradigm • Entity/thematic matching • Neighboring incorporation	• Concatenated values • Multi-source unification • Direct integration
KG^{2}RAG	Static	Graph-as-Context	• Graph-based retrieval • Graph-aware generation	• LLM-based triplet extraction • Subgraph merging • Source-chunk linking	• Seed chunk semantic retrieval • Graph-guided expansion • MST filtering and ranking	• Structured paragraph composition • Self-consistent context blocks • Chunk-path alignment
KG-RAG	Dynamic	Graph-as-Reasoner	• Graph-based retrieval • Post-retrieval reranking • Graph-aware generation	• Few-shot LLM triple extraction • Triple hypernodes • Embeddings for graph elements	• Chain of Explorations (CoE) • Vector + Cypher traversal • Multi-hop inference	• KG-path constrained prompting • Faithful, context-aware output
HippoRAG	Dynamic	Graph-as-Context	• Graph-based retrieval • Graph-aware generation	• OpenIE-based schemaless KG • Synonymy edge augmentation • Dense phrase encoders	• Personalized PageRank • Node specificity (IDF-like) • Multi-hop in one step	• Memory-indexed QA • IRCoT-compatible retrieval • Long-term context integration
ToG-2	Dynamic	Graph-as-Reasoner	• Graph-based retrieval • Post-retrieval reranking • Graph-aware generation	• Wikipedia + Wikidata KGs • Co-occurrence & relation extraction • Optional manual KGs	• Graph-context alternation • Dense retrieval + pruning • KG-based exploration	• Triple-text prompting • Iterative clue summarization • Query re-optimization