applied sciences

MDPI

Article

DualGraphRAG: A Dual-View Graph-Enhanced Retrieval-Augmented Generation Framework for Reliable and Efficient Question Answering

Mengqi Li and Rufu Qin *

State Key Laboratory of Marine Geology, Tongji University, Shanghai 200092, China; lmq1129@tongji.edu.cn
* Correspondence: qinrufu@tongji.edu.cn

Abstract

Graph-enhanced Retrieval-Augmented Generation (RAG) frameworks, such as GraphRAG, improve large language model (LLM)-based question answering (QA) by constructing and leveraging structured, knowledge-condensed graph information. However, they still face challenges in complex multi-hop reasoning tasks and often incur substantial time and resource costs, resulting in low efficiency. To address these limitations, we propose DualGraphRAG, a dual-view graph-enhanced RAG framework designed to achieve both high QA performance and computational efficiency for complex reasoning over open-domain corpora. Specifically, DualGraphRAG constructs a knowledge graph (KG) by automatically extracting triples from unstructured text using LLMs, and embeds KG nodes with unified text embeddings. For each query, multiple types of KG nodes are generated through a dedicated query enhancement module. Based on these nodes, DualGraphRAG employs a dual-view retrieval strategy to retrieve both one-hop triples that capture local context and shortest paths that compress global connectivity information, thereby facilitating answer generation. Experimental results show that, compared with NaiveRAG, GraphRAG, and LightRAG, DualGraphRAG achieves the best or competitive performance on benchmark datasets and significantly improves efficiency. Overall, DualGraphRAG organizes and exploits KG information in a dual-view manner, leveraging triples and shortest paths to offer a reliable and efficient framework for open-domain QA with complex multi-hop reasoning.

Keywords: retrieval-augmented generation; knowledge graph; large language models; question answering; triples; shortest paths; multi-hop

Check for updates

Academic Editor: Ming Liu

Received: 24 January 2026
Revised: 20 February 2026
Accepted: 24 February 2026
Published: 25 February 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

In recent years, large language models (LLMs) [1–4] have advanced rapidly and demonstrated remarkable performance on a wide range of tasks, including question answering (QA) [5,6], sequence labeling [7,8], text summarization [9,10], and machine translation [11,12], significantly transforming the field of natural language processing. Despite their strong generalization capabilities [13,14], LLMs still present inherent limitations due to the lack of up-to-date information and domain-specific knowledge, suffering from issues such as hallucinations [15,16]. Retrieval-Augmented Generation (RAG) has been proposed to address these challenges by enhancing LLMs with externally retrieved knowledge, enabling them to answer questions beyond their parametric memory [17]. As a result, LLMs coupled with RAG achieve improved performance on knowledge-intensive tasks [18].

Theoretically, conventional RAG approaches primarily retrieve information from unstructured corpora, where the retrieved text chunks are often treated as independent and

weakly connected units [19]. Consequently, this retrieval paradigm fails to support effective cross-source information aggregation and explicit capture of dependencies among facts, posing significant challenges for multi-hop reasoning tasks [20]. For instance, when a query involves implicit relational chains spanning multiple facts, conventional RAG approaches tend to prioritize surface-level similarity to the query, resulting in the retrieval of isolated chunks while failing to capture intermediate relations connecting them.

To address this limitation, recent studies have explored the integration of knowledge graphs (KGs) into RAG frameworks [21]. KGs provide explicit structured representations of entities and relations, facilitating graph-based relational composition and multi-hop reasoning, while also offering strong interpretability [22,23]. However, their reliance on symbolic representations, predefined schemas, and complex graph traversal often complicates direct integration with LLMs, necessitating additional alignment or reasoning mechanisms [24,25]. In this context, RAG offers an effective compromise by leveraging KGs' strengths in relational traversal and interpretable chain construction, thereby enabling LLMs to handle queries that require compositional reasoning over multiple related facts.

Building on these insights, a number of concrete methods have been proposed, and the detailed descriptions can be found in a previous review study conducted by Peng et al. [26]. Technically, these methods differ in how KG information is organized for LLM-based answer generation. Although promising results have been achieved, it remains challenging to balance effective organization of KG information with overall efficiency. In practice, methods that emphasize richer graph structures or more comprehensive graph retrieval tend to improve QA performance but incur higher computational costs. Conversely, efficiency-oriented designs simplify graph usage, which can limit the quality of the retrieved information and lead to performance degradation. In particular, most of these methods fail to systematically exploit both local context and global connectivity information within the graph, making it difficult to simultaneously achieve strong performance and high efficiency on complex reasoning QA tasks.

To explicitly address these challenges, the goal of this work is to develop a general-purpose graph-enhanced RAG framework that effectively organizes and balances KG information, thereby providing reliable and efficient answers in the context of complex questions. In this study, we propose DualGraphRAG, a unified graph-enhanced RAG framework that integrates automated knowledge graph construction, query enhancement, and dual-view retrieval to provide structured guidance for LLM-based QA. The main contributions are summarized as follows. First, a LLM is employed to perform one-pass triple extraction from unstructured corpora, and the extracted triples are subsequently used to construct the KG, which is stored in a graph database for efficient retrieval. Second, a query enhancement module is then introduced to align queries with relevant nodes in the knowledge graph (KG nodes), providing precise and complete reasoning anchors. Third, a dual-view strategy is designed to jointly retrieve one-hop triples centered on each node and shortest paths between nodes, representing local context and global connectivity information.

Experimental results on benchmark datasets demonstrate that, compared with baselines, DualGraphRAG achieves the best or highly competitive QA performance, while significantly improving computational efficiency. This indicates that leveraging triples and shortest paths is crucial for achieving a dynamic balance between performance and efficiency within a graph-enhanced RAG framework.

The remainder of this paper is structured as follows. Section 2 reviews related work on RAG and graph-enhanced RAG frameworks. Section 3 introduces the implementation of DualGraphRAG framework, including knowledge graph construction, knowledge graph retrieval, and LLM-based QA. Section 4 describes the details of the experiments and

presents an in-depth analysis of the results. Finally, Section 5 concludes this work and outlines directions for future work.

2. Related Work

RAG framework has been widely adopted to enhance LLM generation with external knowledge. This framework typically consists of two core components, a retriever and a generator, and operates by encoding and matching the input query against an external knowledge base to generate semantically coherent and linguistically fluent answers [18,27,28].

In early studies, conventional RAG frameworks primarily focused on strategies to model, coordinate, and train the retriever and the generator. The goal was to effectively retrieve and leverage external information for LLM-based generation. These efforts explored both how the components are designed and optimized, as well as how their interactions are influenced by different training strategies [29,30].

To realize these goals, researchers investigated a range of techniques, including DPR [31], a typical dense retrieval method that employs dual encoders optimized via contrastive learning to capture semantic similarity between queries and documents. In contrast, sparse retrieval methods, such as TF-IDF and BM25 [32], generally construct inverted indices based on term frequency to achieve computationally efficient matching. Hybrid retrieval methods like ColBERTv2 [33] integrate sparse and dense retrieval through weighting or cascading strategies. Beyond these studies, RAG variants further investigated strategies for integrating retrieved information and coordinating components. Fusion-in-Decoder [18] adopts a delayed fusion strategy by independently encoding retrieved documents to enable more effective utilization of multiple data sources. REALM [34] introduces retrieval mechanisms during pretraining, enabling the models to learn when and how to retrieve relevant external information as needed. REPLUG [35] improves the efficiency of RAG by training only the retriever while reusing a frozen generator.

Despite notable improvements in QA performance, the aforementioned methods primarily focused on modeling, coordination, and training of core components to better exploit unstructured textual documents, without fundamentally altering how information is organized or presented to downstream LLMs. Specifically, these approaches rely heavily on surface-level lexical or semantic matching and lack explicit modeling of entity relations or structural dependencies within the documents [31]. As a result, they often struggle in scenarios involving multi-entity relational integration, multi-hop reasoning, and aggregation of redundant or fragmented evidence.

To address these limitations, recent studies have increasingly explored incorporating structured knowledge, particularly KGs, into RAG frameworks. By combining the relational reasoning capabilities of KGs with the expressive power of LLMs, these approaches aim to provide more effective and interpretable reasoning over multiple related facts [21,25]. Beyond component-level and training-oriented optimizations, such graph-based approaches enable reorganizing knowledge itself, rather than only improving the operations on unstructured textual documents. By organizing information into explicit relational structures, KGs facilitate more efficient and precise information selection than surface-level lexical or semantic matching and help reduce redundancy by aggregating fragmented textual evidence into compact factual units. Moreover, the explicit relational paths encoded in KGs naturally form coherent reasoning trajectories that align with the stepwise inference. Consequently, retrieval mechanisms enhanced by knowledge graphs can provide more structured, concise, and inference-oriented evidence, thereby improving reasoning capabilities in LLMs [36,37].

Recent studies have explored various methods for integrating KGs into RAG frameworks. In this work, we classify these methods into three categories based on the form

in which KG information is organized, as this distinction reflects the type of knowledge provided to downstream LLMs and highlights their respective strengths and limitations.

The first category focuses on incorporating fine-grained KG information, such as triples or local subgraphs, as context for generation. KG-RAG [38] aligns entities mentioned in the query with KG nodes, retrieves relevant triples, converts them into natural language, and prunes them via vector similarity before being fed into the generator. This design reduces token consumption and improves domain-specific performance. However, it heavily relies on curated domain-specific KGs and overlooks the utilization of global graph structure. SubgraphRAG [39] efficiently retrieves query-relevant local subgraphs from the KG and adapts their scale according to the downstream LLM's capabilities. Although this method improves QA accuracy and efficiency through a lightweight retrieval mechanism, its use of the KG remains largely limited to the acquisition and optimization of local subgraphs. KA-RAG [40] constructs a course KG and employs an agent to perform both subgraph retrieval and vector-based retrieval, achieving high retrieval accuracy and semantic consistency, while remaining limited to a single domain and exhibiting constrained multi-hop reasoning capabilities. In summary, methods in this category preserve fine-grained information effectively but typically lack explicit modeling of global connectivity information, limiting their support for complex multi-hop reasoning.

The second category organizes KG information into abstract structures for large-scale, open-corpora scenarios. GraphRAG [41], as a pioneering approach, automatically constructs a KG from unstructured corpora and introduces global query mechanisms. It extracts information via LLMs, applies specialized clustering algorithms to form communities, and generates reports, thereby obtaining multi-level graph abstractions. However, repeated LLM invocations introduce substantial token costs, and the complex design can lead to latency under different query patterns. LightRAG [42] adopts a lighter-weight strategy by linking extracted entities and relations directly to original documents, achieving improved retrieval efficiency and supporting incremental updates. Nevertheless, it does not explicitly leverage the modeled KG structures, which diminishes interpretability. In general, abstraction-based methods offer greater flexibility and scalability but may potentially sacrifice graph structure information and interpretability. HippoRAG 2 [43] explicitly organizes retrieved knowledge into hierarchical and relational abstractions that resemble the functional components of human memory. Building upon the Personalized PageRank mechanism used in HippoRAG [44], HippoRAG 2 enhances passage-level integration and combines offline indexing with online retrieval. These biologically inspired abstractions capture both local factual relations and broader associative connections, enabling richer semantic integration and more robust multi-hop reasoning.

The third category aims to extract explicit chain or path structures from KGs to guide multi-hop reasoning. HyKGE [45] constructs multiple types of reasoning chains based on hypothesized confirmed output nodes, and applies re-ranking strategies before passing the selected chains to the LLM. This framework balances diversity and relevance of retrieved knowledge, but its re-ranking module incurs additional computational overhead and the multi-stage, repeated LLM invocations further add to more costs. THINK-ON-GRAPH [46] guides the LLMs to iteratively and autonomously explore entities and relations within the KG, performing reasoning over the resulting explicit chains to answer the query. This method offers enhanced reasoning depth and traceability compared with static retrieval models, yet requires significant computational resources due to graph searches with substantial breadth and depth. Overall, chain-based methods provide clear reasoning evidence, but the additional planning and search processes incur non-negligible costs.

Taken together, current studies on integrating KGs into RAG frameworks have explored a wide range of strategies for organizing and utilizing KG information, each mak-

ing positive progress in addressing specific challenges. Nevertheless, existing methods inevitably face trade-offs among graph expressiveness, reasoning effectiveness, and computational efficiency. Building on these insights, our study proposes a unified dual-view framework that leverages knowledge graph information collaboratively, aiming to provide reliable and efficient answers for complex multi-hop reasoning tasks.

3. Methods

3.1. Architecture of DualGraphRAG

Similarly to existing paradigms that integrate KGs into RAG, as illustrated in Figure 1, our method consists of three main stages: knowledge graph construction, knowledge graph retrieval, and LLM-based QA. Compared to conventional methods, using a KG instead of document chunks provides more structured and knowledge-condensed information for the LLM. In particular, our proposed DualGraphRAG alleviates both information disconnection and computational inefficiency, leading to improved performance in both single-hop and complex multi-hop QA.

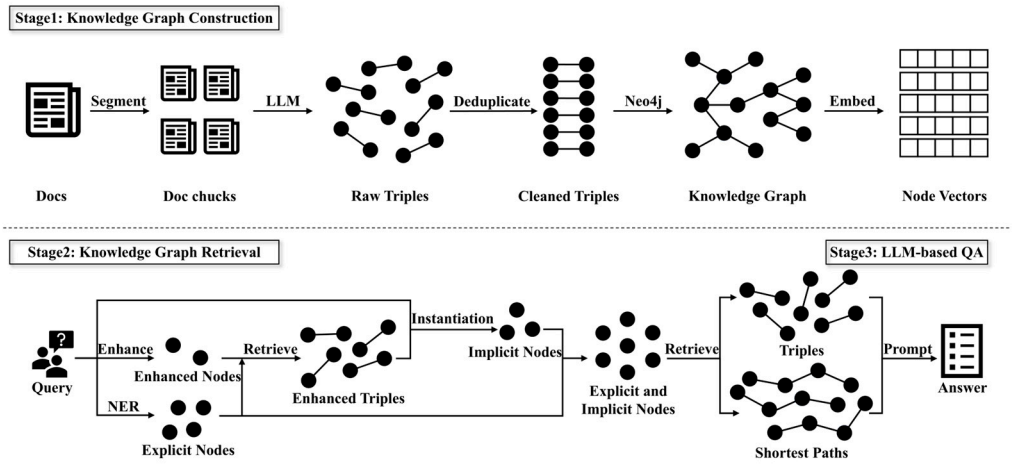


Figure 1. Overview of the DualGraphRAG framework.

- Stage 1: Knowledge Graph Construction

During the construction stage, we build a KG from open-domain corpora by extracting triples using LLMs. The process involves standard text chunking, where triples are directly extracted in their original textual form, preserving more fine-grained information. Triples are then connected and stored in a Neo4j graph database. To ensure accurate semantic representation, all KG nodes are exported and encoded as vectors using the BGE-M3 [47] text embedding model, and stored uniformly in a ChromaDB vector database for subsequent retrieval.

- Stage 2: Knowledge Graph Retrieval

During the retrieval stage, we obtain different types of nodes based on the given query and then retrieve relevant KG information. This stage consists of two main steps: Query Enhancement and Node-Based Retrieval.

Query Enhancement: Starting from entity extraction, we identify explicit entities via named entity recognition (NER) and generate enhanced entities to enrich the query. These entities are then encoded using the same embedding model as before and matched to corresponding KG nodes based on vector similarity computation. The query, together with the information linked to these nodes, is finally used to generate implicit nodes via LLMs, compensating for potential semantic gaps or inaccurate matching.

Node-Based Retrieval: Based on the explicit and implicit nodes obtained in the previous step, we retrieve both one-hop triples centered on each node and shortest paths between pairs of nodes from the KG. These two types of information provide complementary views and granularity, ensuring the knowledge required for downstream QA is adequately captured.

- Stage 3: LLM-Based QA

In the final stage, the retrieved information is organized into prompts and fed into the LLM. By combining the structured knowledge with carefully designed prompts, this stage aims to enhance the LLM's reasoning capabilities, thereby improving both the accuracy and interpretability of the generated answers.

Overall, the knowledge graph construction stage transforms unstructured textual corpora into a structured graph, providing an explicit semantic foundation and supporting multi-hop reasoning. The knowledge graph retrieval stage enhances intent understanding by constructing explicit and implicit nodes, and combines triples with shortest paths to capture complementary local context and global connectivity information. The LLM-based QA stage integrates the structured knowledge into the prompt, enabling the LLM to leverage these different types of information, thereby improving both the accuracy and interpretability of QA. In practical QA scenarios, these components operate coherently and complementarily, jointly contributing to the overall performance.

In the following sections, we provide a detailed description of the aforementioned stages and discuss the design rationale.

3.2. LLM-Based Knowledge Graph Construction

3.2.1. Extraction and Post-Processing of Triples

Similarly to standard RAG pipelines, the corpora used for knowledge graph construction are segmented into chunks to prevent excessively long contexts from degrading the KG's completeness and accuracy. This also avoids prohibitively high costs associated with single-pass processing. In addition, a fixed-length overlap is introduced between adjacent chunks for improved contextual continuity, thus providing references for processes such as coreference resolution. The current DualGraphRAG framework is primarily designed for textual data and does not natively support non-textual modalities such as images, audios, or videos. To process multi-modal data with textual content, such as lists, tables, or diagrams, preprocessing is required to convert these elements into textual representations.

For knowledge graph construction, we employ the Qwen2.5-14B model [48] to extract structured information from the chunks. Differing from conventional LLM-based knowledge graph construction pipelines, our method implements one-pass automatic extraction of triples. In our implementation, the primary objective is to ensure that all relevant information is extracted without omission, while faithfully preserving the original textual expressions.

Specifically, for entity extraction, the categories of entities are strictly constrained and only the most essential content is retained. For example, only the core entity "Marie Curie" is retained, rather than the extended phrase "the famous scientist Marie Curie". Correspondingly, for relation extraction, we similarly enforce requirements of faithfulness and precision, avoiding vague or underspecified expressions such as "has" or "has been". In addition, for coreference resolution, all pronouns and abbreviations are resolved and converted into their complete and canonical forms, thus preventing ambiguous references within the KG. Furthermore, specialized handling is also applied to information enclosed in parentheses, which frequently appears in the corpora. Based on the inferred semantic type (e.g., synonymy, temporal information), such content is extracted as independent triples following predefined expression patterns, preventing the LLM from overlooking

these important complementary details when focusing on the main body of the text. Finally, we follow the output formats adopted in GraphRAG and LightRAG, using dedicated delimiters to standardize the triple extraction results. Although automated LLM-based triple extraction may inevitably introduce incomplete or noisy triples, carefully designed and task-specific few-shot prompting is additionally employed to reinforce the above constraints and ensure robust and high-quality extraction performance. In addition, the subsequent dual-view retrieval strategy mitigates the influence of individual missing links by jointly leveraging local context and global connectivity information. The corresponding prompt used for triple extraction is presented in Figures A1 and A2 in Appendix A, demonstrating the details and exact instructions adopted to ensure faithful generation.

After extraction, the resulting triples are matched according to the dedicated delimiters, and extraneous symbols are removed through post-processing. The next critical step is triple deduplication. Each triple is treated as an atomic unit, and textual similarity is computed to merge triples whose similarity exceeds a predefined threshold. Among similar candidates, the most complete triple is retained as the final representation. This process effectively eliminates redundant information and significantly improves the efficiency of subsequent matching and filtering operations.

3.2.2. Storage and Embedding of Knowledge Graph

For the post-processed triples, we perform a merge operation to form the final KG, which is then stored in a standard graph database through batch import. In this study, we employ Neo4j for graph-based operations and storage due to its native support for graph data modeling. Furthermore, in the storage implementation, we create constraints and indexes using the names of triple entities as key fields. This ensures the node uniqueness and enhances storage and retrieval efficiency, particularly for large-scale data.

After constructing and storing the KG, we perform text embeddings to obtain vector representations of KG nodes, providing a foundation for subsequent retrieval tasks. Specifically, all KG nodes are exported in batches and encoded as dense vectors using the BGE-M3 text embedding model. These vectors are uniformly stored in a vector database, ChromaDB. We perform embeddings at the node level rather than on the entire KG to align with node-level retrieval in later stages, ensuring consistent data representation and operations to maximize the accuracy of similarity-based matching.

Finally, the constructed KG serves as the knowledge base within the RAG framework, supporting retrieval across multiple stages of QA by supplying relevant information.

3.3. Knowledge Graph Retrieval

3.3.1. Query Enhancement

Traditional retrieval processes typically operate directly on the original query. However, in complex QA scenarios, particularly for compositional reasoning problems involving multi-hop questions, the original query often contains limited explicit and directly usable information. As a result, such queries pose significant challenges for both LLMs and conventional RAG methods in generating accurate and reliable answers.

To address this issue, the proposed framework performs query enhancement as the first step within the retrieval stage, extracting both explicit information and implicit referential cues embedded in the query. By explicitly handling the "hops" information involved in multi-hop reasoning, this process enables a more targeted analysis of complex queries, accurately capturing the underlying intent and providing clear guidance for subsequent processing.

Specifically, we maintain the granularity of query enhancement at the level of named entities, which correspond to the head and tail nodes of triples in the KG. This choice is mo-

tivated by the following considerations: (1) At the stage of knowledge graph construction, entity names serve as unique indexing keys. Operating at the named entity level therefore enables more precise identification of information required by the query. (2) At the stage of knowledge graph retrieval, starting from nodes facilitates more efficient acquisition of multi-hop knowledge. (3) Relations may sometimes mislead the problem-solving process, and empirical observations suggest that the number of predicates is generally smaller than that of subjects and objects, which can amplify the impact of errors to some extent. Node-centric retrieval helps mitigate this issue.

In practice, query enhancement consists of three steps, which take the original query as input and ultimately output explicit nodes and implicit nodes.

Named Entity Recognition. For the input query, the LLM is guided to perform NER through a few-shot prompting strategy. Unlike conventional approaches, in addition to explicitly identifiable named entities in the query, the LLM is also required to infer entities necessitated by the reasoning implied in the query. In this work, these two types of entities are defined as explicit entities and enhanced entities, respectively.

The extraction of explicit entities follows standard NER criteria and further imposes additional constraints: under predefined category restrictions, entities are required to preserve the original expression, adopt a minimal surface form, resolve coreferences, and ensure completeness.

Given the stringent requirements and inherent limitations of NER, enhanced entities primarily contribute to the completion of the query's semantic information and the reinforcement of its intent representation. Specifically, they are characterized as precise and complete nouns or noun phrases that can be aligned with KG nodes, excluding overly broad categorical terms. Notably, such entities may be derived from relational verbs when necessary.

Through the extraction of these two types of entities, most of the essential information within the query can be transformed into entity-level representations. The detailed NER prompt is provided in Figure A3 in Appendix A, which also illustrates concrete and representative enhanced entity instances generated by the proposed process.

Embedding Alignment. Subsequently, the entities derived from the query are mapped to corresponding KG nodes through embedding alignment. Concretely, the entities are encoded as dense vectors using the same BGE-M3 text embedding model with a dimensionality of 1024. These vectors are then compared with node embedding vectors stored in ChromaDB using cosine similarity. The KG nodes with the highest similarity scores exceeding a predefined threshold are returned. The threshold is selected according to commonly adopted empirical ranges for semantic textual similarity, and is further validated to balance precision and recall. Correspondingly, the resulting KG nodes are defined as explicit nodes and enhanced nodes. This general process of transforming textual entities into KG nodes, termed embedding alignment, maps unstructured text into rich and operable graph-structured knowledge.

Node Enhancement. Compared to single-hop questions, multi-hop questions typically require the composition of multiple intermediate facts to derive the final answer. While the explicit and enhanced nodes reflect relatively direct interpretations of the query, they may still suffer from semantic insufficiency or inaccurate matching. To address this issue, we design a tailored node enhancement strategy.

Node enhancement is used to explicitly identify and extract intermediate information that may be involved in the query and incorporate it into subsequent processing. This strategy aims to further clarify the direction of reasoning or problem-solving while reducing data processing complexity. In particular, it focuses on query elements exhibiting "hops" characteristics, which often manifest as referential or associative expressions. Taking "the

person who discovered radium" as an example, the semantically equivalent KG node "Marie Curie" is defined as an implicit node, in contrast to explicit nodes obtained through NER. The process of representing such information as its corresponding KG node is termed implicit node instantiation.

During this step, since explicit and enhanced nodes collectively capture the core semantic content of the query, we retrieve one-hop information centered on these nodes as reference for the instantiation. The original query, the two types of nodes, and their corresponding one-hop information are jointly fed into the LLM, which generates the final implicit nodes according to predefined instantiation rules. The primary objective of instantiation is to resolve "hops" information contained in the query. Accordingly, equivalence cases such as the aforementioned "Marie Curie" example are prioritized. Furthermore, since the identification of reasoning-path nodes is central to solving multi-hop questions, nodes with higher connectivity or frequency of occurrence are more likely to be selected as target implicit nodes. Typical examples include nodes that occur multiple times across different triples, where they function as head nodes in some triples and as tail nodes in others. Finally, the generated implicit nodes must originate from one-hop information and be distinct from the previously obtained nodes. To control subsequent computational and processing complexity, the number of implicit nodes is strictly limited to no more than three. These implicit nodes complement potentially missing reasoning anchors inferred from the query. Since most multi-hop questions involve only a small number of such intermediate concepts, this limit provides sufficient guidance while avoiding loosely related candidates. Moreover, this threshold is designed as a general-purpose setting, which can be adjusted when more extensive query-guided node coverage is needed. The implicit node instantiation prompt is provided in Figure A4 in Appendix A, where the practical implementation details can be observed, including how implicit nodes are derived from the input information through the designed strategy.

3.3.2. Node-Based Retrieval

After query enhancement, explicit nodes are obtained directly from the query, alongside implicit nodes that represent intermediate reasoning concepts implied by the query. Based on these two types of nodes, knowledge graph retrieval is performed to construct richer contextual information, thereby enabling targeted handling of multi-hop questions that require the composition of multiple intermediate facts.

In terms of the specific implementation, the explicit nodes and implicit nodes are treated as an input node set, from which two categories of KG information are retrieved. The first consists of one-hop information centered on each node, namely all non-duplicated triples that involve the given node in the KG. The second consists of shortest paths between all pairs of distinct nodes in the node set. The rationale of these two retrieval strategies is detailed in the following sections.

With regard to the first category, namely one-hop information, we treat it as local context that captures essential information associated with each node. As the fundamental representation units of KGs, triples provide fine-grained and structured evidence that preserves precise semantic information while reducing noise compared with loosely connected text chunks. This design not only ensures strong performance on relatively simple questions, but also maintains the completeness of retrieved KG information at the individual node level. In particular, considering both the characteristics of knowledge graph construction and the storage model of Neo4j database, the original directionality of triples is preserved, which allows the semantics of the original corpora to be effectively compressed and faithfully reconstructed.

The second category, namely the shortest paths between node pairs, is regarded as global connectivity information that represents the most relevant and concise relations between different nodes. On the one hand, since multi-hop questions require the combination of multiple intermediate facts, chain-structured information provides a natural and intuitive form of compositional reasoning evidence. Such paths not only efficiently encode the topological relationships between nodes, but also suggest potential causal or inferential relationships to some extent, thereby offering richer relation semantics at a finer granularity. By incorporating this information, we aim to enable the LLM to accurately identify connections among different pieces of knowledge, reduce hallucinations, and potentially enhance its reasoning capabilities during problem solving. On the other hand, given that KGs constitute large-scale structured data, conventional approaches such as k-hop subgraph extraction or random walk often incur substantial time and computational costs. In contrast, we innovatively adopt shortest-path retrieval, as it approximates minimal reasoning trajectories that better align with the intrinsic requirements of multi-hop questions, thereby yielding highly informative relation evidence while maintaining low computational overhead. Within the proposed framework, we set the maximum path length to four hops. This setting is based on the commonly adopted hop ranges in shortest path reasoning over KGs, which are typically limited to a small number of hops. Allowing up to four hops slightly extends this range to better support multi-hop reasoning while still controlling noise propagation and computational overhead. In addition, longer paths would substantially enlarge the search space of shortest path retrieval, leading to noticeable efficiency degradation. Therefore, this choice achieves a balanced trade-off between reasoning coverage and efficiency. Importantly, this setting is designed as a general-purpose default; for tasks that require deeper reasoning, the maximum path length can be increased, though with higher computational cost. Notably, similar to the representation of triples, relation directionality is preserved along shortest paths. As a result, bidirectional arrows may appear within a path. This pattern is interpreted as an alternative form of inferential relationship that differs from strict causality, where two nodes become associated by converging toward a common intermediate node along the path. Since the retrieval process is performed over all obtained nodes, shortest paths that exhibit containment relationships are merged, retaining the one with the maximum length among them. For node pairs without available paths, an empty result is returned.

The combination of triple-based local context and shortest path-based global connectivity information provides diverse and complementary views: triples ensure precise factual grounding for individual nodes, while shortest paths capture coherent multi-hop connections. This dual-view organization offers sufficiently informative and fine-grained relation evidence with controlled computational complexity, thereby enabling effective reasoning over complex questions, particularly those involving multi-hop QA.

3.4. LLM-Based QA

Consistent with standard RAG pipelines, in the final stage, the information retrieved in the previous stages is organized together with the original query using a carefully designed and structured prompt, which is then passed to the LLM for answer generation. Notably, the prompt preserves the structured representations of directed triples and bidirectional paths, and explicitly explains the granularity and semantic meaning associated with each form, rather than converting them into plain natural language. This design leverages structured information to activate the LLM's latent reasoning capabilities, thereby enabling more accurate and reliable answer generation.

https://doi.org/10.3390/app16052221

4. Experiments

4.1. Experimental Setup

Base Models. Qwen2.5-14B and BGE-M3 are selected as the base model and the text embedding model, respectively. The choice of the LLM is motivated by a balance among the application scenarios, the available computational resources, and the model's intrinsic capabilities. On the one hand, given the large scale of the experimental data and potential requirements for data confidentiality in future applications, relying on costly commercial APIs is neither economical nor feasible. In contrast, local deployment of an open-source LLM presents clear advantages of both efficiency and security. On the other hand, considering the unique characteristics of our algorithm and the need for reproducibility, the LLM is expected to demonstrate strong instruction-following capability, multilingual adaptability, and reasoning competence. Accordingly, the Qwen series aligns well with these considerations. Regarding the embedding model, BGE-M3 is selected primarily because it provides high-quality embedding vectors while maintaining computational efficiency. Moreover, its support for multi-functional and multi-granularity scenarios aligns closely with the demands of knowledge graph processing.

Evaluation Datasets. For model evaluation, two categories of publicly available QA benchmark datasets are selected, covering both simple single-hop and complex multi-hop questions. The first category is primarily used to evaluate the model's basic capabilities in retrieving relevant information and performing QA. For this purpose, the Natural Questions (NQ) [49] dataset is adopted. This dataset is constructed based on real Google search queries and corresponding Wikipedia corpora, covering a broad range of topics. Specifically, the version processed by Wang et al. [50] is used, which provides a standardized structure. The second category focuses on the evaluation of the model's advanced capabilities in integrating multiple pieces of related information and reasoning. For this purpose, the HotpotQA [51] and 2WikiMultihopQA [52] datasets are adopted. These datasets generate complex multi-hop questions through a carefully designed pipeline, requiring compositional reasoning over multiple paragraphs or documents to resolve. Moreover, both datasets also provide the supporting facts or evidence information required to solve each question, enabling the model to perform interpretable reasoning.

The three datasets mentioned above all share a unified organizational format that includes queries, answers, and associated corpora. In the implementation, we retain the queries and answers solely for evaluation purposes, while concatenating all provided corpora as the raw input text. These corpora include not only documents that support the correct answers, but also distracting or irrelevant documents, which are intentionally preserved to evaluate the model's capability to filter noise and extract informative evidence. This operation is intended to simulate realistic QA scenarios as closely as possible. With respect to dataset scale, we randomly sample 1000 QA pairs from each dataset for evaluation, in order to mitigate the impact of stochasticity in model responses while maintaining a balance between experimental feasibility and statistical reliability. After concatenation, the total number of English characters in the input corpora for NQ, HotpotQA, and 2WikiMultihopQA is 6,077,270, 5,521,998, and 3,603,916, respectively.

Baselines. For the benchmark datasets described above, the following representative and competitive RAG frameworks are selected as baselines.

NaiveRAG: As the most fundamental RAG framework, NaiveRAG establishes the standard retrieval and generation pipeline. It first segments documents into text chunks and encodes them as dense vectors using a text embedding model. Given a specific query, the top-k most relevant text chunks are retrieved via vector similarity search. These retrieved chunks are then concatenated and provided to the LLM to generate the final answer.

GraphRAG: GraphRAG is a representative framework to systematically formalize knowledge graph construction and retrieval as an integrated workflow. It leverages LLMs to extract entities and relations from documents for knowledge graph construction. On top of the constructed graph, it performs entity-level community clustering and generates summary reports for different communities. To accommodate queries at varying levels of abstraction, GraphRAG supports multiple answering modes such as local search, global search, and drift search.

LightRAG: LightRAG is a lightweight framework that achieves an effective balance between LLM computational costs and QA performance. Similarly to GraphRAG, it employs LLMs to extract entities and relations for knowledge graph construction. In addition, it constructs specialized key–value pairs to link keywords with relevant textual content, thereby improving retrieval efficiency and enabling incremental updates of the KGs. This framework primarily employs a dual-level retrieval system based on keyword matching, including both low-level and high-level retrieval.

We include NaiveRAG as a baseline to isolate and evaluate the performance gains introduced by incorporating structured KGs. In contrast, GraphRAG and LightRAG are adopted as baselines to evaluate both performance and efficiency under settings that do not involve complex graph reasoning, particularly under resource-constrained scenarios.

4.2. Results and Analysis

4.2.1. Implementation Details

For all methods evaluated in our experiments, the temperature of the LLM is uniformly set to zero, with a context window size of 32,768 tokens. The text embedding model is configured with an embedding dimension of 1024 and a maximum input length of 8192 tokens. For document segmentation, we adopt a chunk size of 1024 tokens with an overlap of 20 tokens across all methods.

For GraphRAG and LightRAG, we retain the default hyperparameter settings reported to yield the best performance in their respective original studies. The only modification is that we configure the response type in their QA prompts to "Short Phrase" to better match the answer formats of the benchmark datasets and ensure a fair and accurate evaluation of their QA performance. Furthermore, considering the nature of the evaluated question types, both methods are operated in the local mode, which is consistent with precise and fact-oriented QA scenarios.

All baselines and the proposed framework run on a single NVIDIA RTX A6000 GPU (manufactured by NVIDIA, Santa Clara, CA, USA) with 48 GB of memory. All hyperparameter configurations and model selections are designed to ensure experimental reproducibility and practical feasibility.

4.2.2. Metrics

We focus primarily on the accuracy of the answers generated by the LLM. Accordingly, for all QA datasets considered in this work, we adopt Exact Match (EM) and F1 scores as evaluation metrics. In addition, the execution time of each model for both knowledge graph construction and retrieval-based QA on each dataset is used to evaluate the efficiency of these two stages.

4.2.3. Performance Analysis

We conduct experiments using Qwen2.5-14B on the NQ, HotpotQA, and 2WikiMultihopQA datasets. Table 1 reports the QA performance of different methods across these datasets, while Table 2 reports the corresponding time costs.

Table 1. Question answering (QA) performance of all methods on benchmark datasets. Bold numbers indicate the best performance.

	NQ		HotpotQA		2WikiMultihopQA		Average	
	EM	**F1 Scores**	**EM**	**F1 Scores**	**EM**	**F1 Scores**	**EM**	**F1 Scores**
NaiveRAG	27.40	41.22	32.60	53.57	36.20	43.57	32.10	46.12
GraphRAG	15.90	21.30	19.60	27.77	22.90	26.65	19.50	25.24
LightRAG	31.20	42.23	44.20	**60.68**	41.60	50.01	39.00	50.97
DualGraphRAG	**33.90**	**44.60**	**47.20**	59.56	**50.30**	**55.56**	**43.80**	**53.24**

Table 2. Execution time (min) of graph-based methods on different datasets for knowledge graph construction and retrieval-based QA. Bold numbers indicate the shortest execution time.

	NQ			HotpotQA			2WikiMultihopQA		
	Construction	**QA**	**Total**	**Construction**	**QA**	**Total**	**Construction**	**QA**	**Total**
GraphRAG	12,733	1119	13,852	9976	1281	11,257	9145	1192	10,337
LightRAG	3280	532	3812	1952	518	2470	934	576	1510
DualGraphRAG	**1799**	**151**	**1950**	**1402**	**152**	**1554**	**750**	**148**	**898**

As illustrated in Table 1, DualGraphRAG outperforms the baselines nearly in all settings. In particular, its Exact Match is consistently higher than those of the baselines, indicating precise alignment with the ground truth. Regarding F1 scores across datasets, DualGraphRAG only slightly underperforms LightRAG on HotpotQA, which contains a more diverse set of question types. Some prior studies have suggested that many questions in HotpotQA can be solved without performing true multi-hop reasoning [53]. Consequently, basic solutions such as LightRAG can perform equally well or slightly better than specialized multi-hop solutions like DualGraphRAG. In all other cases, DualGraphRAG achieves the best performance with notable improvements. Specifically, on the NQ dataset, DualGraphRAG improves the F1 scores by 2.37%, 23.3%, and 3.38% over LightRAG, GraphRAG, and NaiveRAG, respectively; on HotpotQA, by 31.79% and 5.99% over GraphRAG and NaiveRAG; and on 2WikiMultihopQA, by 5.55%, 28.91%, and 11.99% over LightRAG, GraphRAG, and NaiveRAG. These quantitative analyses and comparisons of QA performance demonstrate the effectiveness of DualGraphRAG in leveraging structured knowledge for reliable answer generation.

The performance gains can be attributed to the coordinated design of the query enhancement module and the dual-view retrieval strategy. Specifically, the query enhancement module first aligns the query with KG nodes via NER and embedding alignment, and further instantiates implicit nodes through node enhancement, thereby providing more complete reasoning anchors and explicitly guiding the direction of problem solving. Based on these nodes, the dual-view retrieval strategy gathers complementary information from two views: one-hop triples that capture local context and shortest paths that compress global connectivity information. Compared with NaiveRAG, this query-aware retrieval of triples reduces semantic noise introduced by loosely connected text chunks; compared with graph-based baselines, the incorporation of shortest paths offers more effective solution trajectories that better align with the intrinsic requirements of multi-hop reasoning. In practice, these differences also reflect distinct strategies for selecting and balancing KG information. NaiveRAG retrieves information directly from unstructured corpora, which may introduce loosely related content; GraphRAG retrieves more diverse and complex information at higher computational cost; and LightRAG links entities directly to documents for efficient but structurally limited context. In contrast, DualGraphRAG balances local and global KG information effectively, enabling sufficient coverage without excessive

noise or overhead. This collaboration between query enhancement and dual-view retrieval leads to more accurate and interpretable reasoning, particularly for complex multi-hop questions. Overall, DualGraphRAG achieves the highest average EM and F1 scores across multiple open-domain QA datasets, demonstrating strong adaptability and robustness across diverse QA scenarios.

Notably, compared with NaiveRAG, DualGraphRAG exhibits substantially larger performance gains on multi-hop QA datasets relative to single-hop QA datasets. This finding suggests that incorporating KGs and constructing specific structured information can enhance the reasoning capabilities of LLMs, enabling more targeted handling of multi-hop questions. In contrast, the relatively weak performance of GraphRAG can be attributed to its overly complex and redundant structural design, which leads to significant performance degradation when processing large volumes of corpora, even falling below the performance of NaiveRAG. Therefore, when integrating KGs into RAG frameworks, it is crucial to balance empirical effectiveness with the structural rationality of the overall design. In this regard, DualGraphRAG performs particularly well, demonstrating that its concise and efficient architecture maintains high performance while avoiding unnecessary complexity.

In addition, Table 2 and Figure 2 both present and compare the execution time of different graph-based methods, excluding NaiveRAG, across datasets and stages to evaluate the efficiency of graph-enhanced frameworks. Specifically, the knowledge graph construction stage includes the time from corpus to KGs, and the retrieval-based QA stage covers the entire process from query to answer generation. For DualGraphRAG in particular, the retrieval-based QA stage fully accounts for all query processing steps, including NER, embedding alignment, node enhancement, and subsequent node-based retrieval leading to the final answer. This ensures that the execution time reflects the complete end-to-end pipeline and allows for a fair comparison with baseline methods.

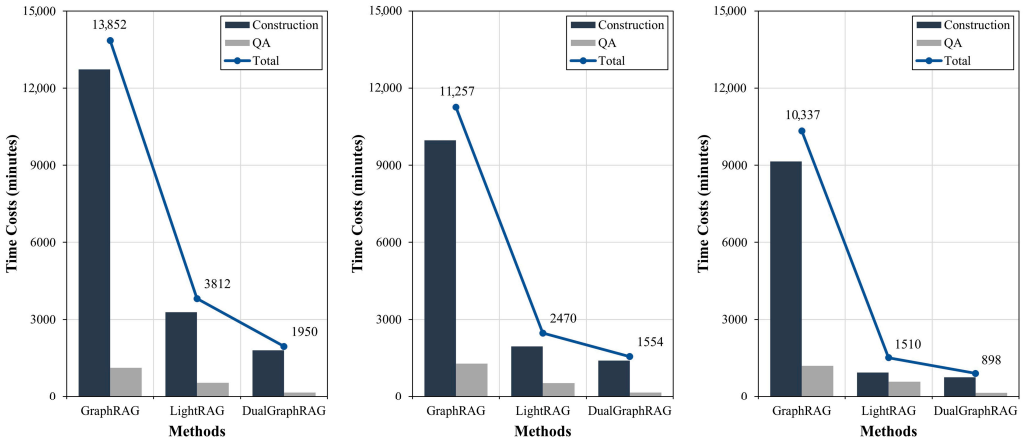


Figure 2. Execution time comparison of graph-based methods across datasets and different stages. (**a**) NQ; (**b**) HotpotQA; (**c**) 2WikiMultihopQA.

The results show that DualGraphRAG achieves up to 1.8× speedup in knowledge graph construction and 3.9× in retrieval-based QA over LightRAG, yielding an overall 2× efficiency improvement. Compared with GraphRAG, the corresponding speedup reaches 12.2× and 8.4×, yielding up to 11.5× overall improvement. It can be observed that DualGraphRAG consistently achieves the fastest execution time across both individual modules and the entire process. The consistent improvements across both stages suggest that the efficiency gains are not derived from a single module but from multiple coordinated designs. Specifically, these efficiency gains can be attributed to three primary factors.

First, during knowledge graph construction, the framework preserves original textual expressions and performs exhaustive extraction in a single pass, which reduces repeated LLM filtering and decision-making. This design minimizes redundant reasoning calls and leads to a lower construction time complexity. Second, in the retrieval stage, DualGraphRAG performs text embedding and similarity computation exclusively on node information rather than full textual contexts. By reducing the processed information granularity from text chunks to compact KG nodes, the computational volume is substantially decreased, which directly contributes to the observed speedup. Third, the retrieval process explicitly constrains both the number of returned entities and the maximum hop length of shortest paths. These constraints and the choice of shortest paths effectively bound the search space and can be interpreted as a structured pruning strategy, preventing exponential growth in multi-hop reasoning while maintaining sufficient information coverage.

Notably, the total time costs of all frameworks generally exhibit a positive correlation with the volume of processed corpora. However, DualGraphRAG maintains a stable time cost during retrieval-based QA. In addition to the factors discussed above, this stability can be attributed to query enhancement which clarifies the reasoning direction through implicit nodes, as well as to the dynamic balance achieved between the number of retrieved triples and shortest paths under typical KG scales. Although DualGraphRAG achieves slightly lower QA performance than LightRAG on HotpotQA, its time cost is nearly half that of LightRAG, indicating a favorable trade-off between efficiency and performance. These benefits arise from the coordinated query enhancement and dual-view retrieval strategy, which provides both precise reasoning anchors and complementary reasoning information via triples and shortest paths without excessive computational overhead, differentiating DualGraphRAG from existing graph-based methods.

In summary, DualGraphRAG demonstrates an excellent balance between QA performance and computational efficiency, highlighting its effectiveness and efficiency on publicly available datasets. Although the framework is originally developed as a general-purpose solution, its coordinated design of automated knowledge graph construction, query enhancement, and dual-view retrieval enable robust and interpretable reasoning across diverse scenarios. This suggests that when applied in real-world QA systems or domain-specific applications, only minor adaptations, such as implementing re-ranking strategies or incorporating domain ontologies, are needed to maintain competitive performance. Together, these features make DualGraphRAG a practical and adaptable framework for both general and specialized knowledge-intensive tasks.

4.2.4. Ablation Study

To quantify the contribution of individual components in DualGraphRAG, we conduct an ablation study on the 2WikiMultihopQA dataset, which is specifically designed for multi-hop reasoning. The full model is compared with three variants by removing triple retrieval, shortest path retrieval, and implicit node instantiation, respectively. The detailed results of QA performance are reported in Table 3.

Table 3. Ablation study results: QA performance on 2WikiMultihopQA dataset. Bold numbers indicate the best performance.

	2WikiMultihopQA	
	EM	**F1 Scores**
DualGraphRAG	**50.30**	**55.56**
w/o triple retrieval	27.10	31.20
w/o shortest path retrieval	41.90	48.33
w/o implicit node instantiation	30.60	33.98

As shown in Table 3, removing any individual component leads to a noticeable performance degradation, indicating that all components are essential and that the framework benefits from their coordinated interaction rather than a simple additive combination. In particular, removing triple retrieval causes the largest degradation, demonstrating that triples serve as the most fundamental structural units and provide the most directly relevant local evidence. This observation aligns with the theoretical role of KGs as carriers of fine-grained relational facts. When shortest path retrieval is removed, the model still maintains relatively strong performance, suggesting that triples can partially capture multi-hop relations in some cases. However, the inclusion of shortest paths effectively models global connectivity, which helps form explicit and coherent reasoning chains and further improves reasoning performance and stability. In addition, removing implicit node instantiation results in a substantial performance degradation, highlighting its critical role in multi-hop reasoning. By complementing intermediate concepts inferred from the query, implicit nodes provide more precise reasoning anchors and help bridge gaps between scattered pieces of knowledge, which is vital for generating the complete reasoning chains.

Overall, these results strongly support that the performance gains of DualGraphRAG arise from the coordinated integration of complementary structured KG information and query-oriented semantic completion, rather than relying on an individual enhancement component. They further indicate that effective multi-hop reasoning depends not only on graph structures but also on the ability to supplement missing intermediate concepts.

4.2.5. Case Study

We present a representative case in Figure 3, which comprehensively illustrates the process from the initial query to the identification of different types of nodes, the retrieval of relevant triples and shortest paths, and the generation of the final answer, thereby demonstrating the effectiveness of DualGraphRAG. The highlighted portions indicate information directly related to answering the query.

Several key findings can be drawn from this case study: (1) The final explicit and implicit nodes are more precise and concise compared to the original query, with the implicit nodes providing clearer guidance toward the reasoning direction. Together, these nodes facilitate a better understanding of the query, enhance the expression of user intent, and reduce redundant or irrelevant information. For instance, the explicit nodes “Garth Richardson” and “The Iveys” identify the primary focus of the query, while the implicit nodes “Jack Richardson” and “Badfinger” further clarify equivalent multi-hop information and the exploration direction. (2) The retrieved triples and shortest paths provide complementary information from two distinct views for answer generation. In this case, the triples primarily offer foundational support, whereas the final answer relies mainly on the shortest paths. It can be noted that the information contained in the shortest paths aligns closely with the logical flow of the query, enabling the LLM to reliably produce the correct answer. We conclude that such path information, which consists of the complete chain of nodes and relations, can potentially enhance the reasoning capabilities of LLMs by supporting stepwise resolution of multi-hop questions with clear guidance and structured evidence. (3) Adopting the shortest path strategy achieves the maximization of effective information with minimal resource costs, contributing significantly to the balance between QA performance and computational efficiency. In this case, the shortest paths are precise and concise, containing exactly the information necessary to address the query.

Query:
Garth Richardson is the son of the producer who worked for which group that evolved from The Iveys?

Explicit entities: ['Garth Richardson', 'The Iveys']
Enhanced entities: ['producer', 'group', 'parent']

Explicit nodes: ['Garth Richardson', 'The Iveys']
Enhanced nodes: ['producer']

Query-enhanced triples:
[['Loaded Gun', 'was recorded by', 'Garth Richardson'], **['Badfinger', 'renamed themselves from', 'The Iveys'],** ['The Iveys', 'was formed in', '1961 and Swansea and Wales'], ['Garth Richardson', 'engineered recordings for', 'Mötley Crüe and Nickelback and The Red Hot Chili Peppers'], ['Garth Richardson', 'produced for', 'Kittie and Mudvayne and Rage Against the Machine and Shihad and The Melvins'], ['The Iveys', 'was formed by', "David 'Dai' Jenkins and Ham and Ron Griffiths"], ['Garth Richardson', 'is', 'a Canadian music engineer and a Canadian music producer'], ['Garth Richardson', 'is also known as', 'Rage Against the Machine'], ['Loaded Gun and Only Revolutions and Siren Song of the Counter Culture and The Fiction We Live', 'produced by', 'Garth Richardson'], ['Garth Richardson', 'co-founded', 'the Nimbus School of Recording Arts'], ['The Iveys', 'were signed by', "the Beatles' Apple label"], ['Griffiths', 'left', 'The Iveys'], ['I Am Destroyer and Loaded Gun', 'was produced by', 'Garth Richardson'], ['The Iveys', 'were signed in', '1968'], ['Nimbus School of Recording & Media', 'was founded by', 'Garth Richardson'], ['Peter Richardson', 'acted as', 'producer'], ['Garth Richardson', 'also known as', 'GGGarth and Mudvayne, Rage Against The Machine'], ['Garth Richardson', 'has the nickname', 'GGGarth'], **['Garth Richardson', 'is the son of', 'Jack Richardson']**]

Implicit nodes: ['Jack Richardson', 'Badfinger']
Final nodes: ['Garth Richardson', 'The Iveys', 'Jack Richardson', 'Badfinger']

Final triples:
[['Jack Richardson', 'was a pioneer of the music recording industry in', 'the 1960s and 1970s'], ['Jack Richardson', 'was a pioneer of', 'the music recording industry'], ['Loaded Gun', 'was recorded by', 'Garth Richardson'], ['Badfinger', 'recorded', 'five albums'], ['Badfinger', 'became embroiled in', "the chaos of Apple Records' dissolution"], ['Badfinger', 'recorded for', 'Apple'], **['Badfinger', 'renamed themselves from', 'The Iveys'],** ['The Iveys', 'was formed in', '1961 and Swansea and Wales'], ['Garth Richardson', 'engineered recordings for', 'Mötley Crüe and Nickelback and The Red Hot Chili Peppers'], ['Garth Richardson', 'produced for', 'Kittie and Mudvayne and Rage Against the Machine and Shihad and The Melvins'], ['The Iveys', 'was formed by', "David 'Dai' Jenkins and Ham and Ron Griffiths"], ['Jack Richardson', 'is', 'a Canadian music producer'], ['Garth Richardson', 'is', 'a Canadian music engineer and a Canadian music producer'], ['Badfinger', 'consisted of', 'Joey Molland and Mike Gibbins and Pete Ham and Tom Evans'], **['Jack Richardson', 'credits include', 'Alice Cooper and Badfinger and Poco and The Guess Who'],** ['Garth Richardson', 'is also known as', 'Rage Against the Machine'], ['Loaded Gun and Only Revolutions and Siren Song of the Counter Culture and The Fiction We Live', 'produced by', 'Garth Richardson'], ['Garth Richardson', 'co-founded', 'the Nimbus School of Recording Arts'], ['The Iveys', 'were signed by', "the Beatles' Apple label"], ['Griffiths', 'left', 'The Iveys'], ['I Am Destroyer and Loaded Gun', 'was produced by', 'Garth Richardson'], ['Badfinger', 'engaged', 'Stan Polley'], ['The Iveys', 'were signed in', '1968'], ['Nimbus School of Recording & Media', 'was founded by', 'Garth Richardson'], ['Garth Richardson', 'also known as', 'GGGarth and Mudvayne, Rage Against The Machine'], ['Garth Richardson', 'has the nickname', 'GGGarth'], ['Badfinger', 'was', 'a British rock band'], **['Garth Richardson', 'is the son of', 'Jack Richardson']**]

Final shortest paths:
['Garth Richardson -[is the son of]-> Jack Richardson -[credits include]-> Badfinger -[renamed themselves from]-> The Iveys']

Answer: Badfinger
Ground truth: ['Badfinger']

Figure 3. Case study of a QA example using DualGraphRAG.

Additionally, it can be observed that providing triples and shortest paths through knowledge graph retrieval enables straightforward traceability of the LLM's outputs. DualGraphRAG thus offers transparent reference information, substantially improving interpretability in QA tasks. Beyond interpretability, this feature also carries practical value in real-world applications. In cases where potential errors may exist, tracing back through the retrieved triples and shortest paths allows correction of the KG, thereby preventing the propagation of errors and improving both the quality and accuracy of the KG. From a coverage perspective, the constructed knowledge graph is intended to

capture the essential entities and relations required for multi-hop reasoning over typical queries. As illustrated by the preceding performance analysis, the current framework demonstrates strong performance across multiple benchmark datasets, indicating that this KG quality is sufficient to achieve the framework's intended goals for general-purpose applications. Although full domain completeness may not be guaranteed in the current implementation, the framework is flexible and can be extended with domain-specific ontologies for specialized KG coverage.

5. Conclusions

In this work, we propose DualGraphRAG, a novel graph-enhanced RAG framework that constructs knowledge graphs by automatically extracting triples from unstructured corpora and organizes KG information for LLM-based question answering through a dual-view strategy. DualGraphRAG enhances QA performance while substantially improving computational efficiency.

Experimental results demonstrate that DualGraphRAG performs exceptionally well on complex multi-hop reasoning tasks, achieving the best or highly competitive performance across benchmarks. Notably, even when performance gains over closely related graph-based baselines are marginal, DualGraphRAG requires only about half of the total time. This demonstrates that the proposed framework, which leverages the coordinated query enhancement and dual-view retrieval strategy, provides an efficient and sufficiently expressive alternative to existing graph-based methods.

Nevertheless, the current framework still exhibits certain limitations in selecting and balancing KG information. While the proposed design already organizes and exploits KG information effectively, its performance could be enhanced by more refined control. Future research may focus on designing re-ranking algorithms that assign specific weights to different KG nodes, enabling more targeted filtering of retrieved information with varying importance and further improving performance. In addition, domain-specific adaptations could be explored by incorporating domain ontologies to guide KG construction and retrieval, which would strengthen the specialization and adaptability of the DualGraphRAG framework across diverse application scenarios.

Author Contributions: Conceptualization, R.Q. and M.L.; methodology, R.Q. and M.L.; software, M.L.; validation, R.Q. and M.L.; formal analysis, M.L.; investigation, M.L.; resources, R.Q.; data curation, M.L.; writing—original draft preparation, M.L.; writing—review and editing, R.Q. and M.L.; visualization, M.L.; supervision, R.Q.; project administration, R.Q.; funding acquisition, R.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China (2021YFC2800500).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The NQ dataset utilized in this work can be obtained from https://huggingface.co/datasets/yhao-wang/rear-eval (accessed on 22 January 2026); the HotpotQA dataset utilized in this work can be obtained from http://curtis.ml.cmu.edu/datasets/hotpot (accessed on 22 January 2026); the 2WikiMultihopQA dataset utilized in this work can be obtained from https://www.dropbox.com/scl/fi/heid2pkiswhfaqr5g0piw/data.zip?rlkey=ira57daau8lxfj022xvk1irju&e=1 (accessed on 22 January 2026).

Acknowledgments: The authors are grateful to researchers who provided the public datasets utilized in this work. We also thank the reviewers and editors for their suggestions to improve the quality of this paper.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

The appendix provides representative prompts used in the DualGraphRAG framework. Specifically, three types of prompts are illustrated: triple extraction prompt, named entity recognition prompt, and implicit node instantiation prompt. For clarity, the triple extraction prompt is illustrated across two figures, showing the complete structure and stepwise design.

Triple Extraction Prompt

-----Role-----

You are an expert in knowledge extraction.
Your task is to extract all (subject, predicate, object) triples from the following text.

-----Extract Rules-----

Follow these rules strictly:

1. **Entity type awareness**:
   Consider only entities belonging to the following categories:
   [person, organization, location, facility, product, event, time, award, scientific_concept, number].
   When forming subjects and objects, ensure they fit one of these categories.

2. **Concise and accurate extraction**:
   Extract triples in their most compact and semantically accurate form.
   - For entities (subjects and objects): keep only the essential names; remove redundant attributives (descriptors, articles, or modifiers) unless they are part of the entity's core name.
     Example: use "Marie Curie" instead of "the famous scientist Marie Curie".
   - For relations (predicates):
     use the verb phrase that captures the factual relation without redundancy; avoid vague predicates such as "has", "has been" unless truly required by context.
   The goal is to make triples precise and consistent.

3. **Keep original expressions**:
   Preserve the exact wording from the original text for subjects, predicates, and objects.
   Do not paraphrase, translate, or summarize names.
   Split subjects and objects into minimal semantic units while preserving meaning.
   Extract each meaning as an independent triple.

4. **Parentheses content extraction**
   Only when entity includes parentheses:
   - Bracketed content must be extracted into one or more additional triples depending on semantics:
     - Synonyms / abbreviations / aliases → predicate: "also known as"
     - Time info → predicate: "born in", "released in", "established in"
     - Location → predicate: "located in" or suitable geo relation
   Extract each meaning as an independent triple.

5. **Coreference resolution**:
   Resolve all pronouns, vague references ("the director", "the film"), and shortened names ("Rohmer") to the full, explicit entity they refer to using the context across sentences.
   After this step, all references must be normalized to their complete entity names already mentioned in the text, with no pronouns or partial names remaining.

6. **Multiple facts**:
   Extract all factual relations mentioned, including cross-sentence ones.

7. **Output format (use structure markers)**:
   To avoid invalid JSON formatting, use the following delimiters:
   - {tuple_delimiter} between triple elements
   - {record_delimiter} between different triples
   - {completion_delimiter} to mark the end of the output
   Each triple must follow this format:
("triple"{tuple_delimiter}<subject>{tuple_delimiter}<predicate>{tuple_delimiter}<object>){record_delimiter}
   At the end, output {completion_delimiter} to mark completion.

8. **Output only structured triples**:
   Do not include any explanations, extra text, or comments.
   Do not output JSON, code blocks, or human-readable sentences.

Figure A1. Triple Extraction Prompt (Part 1).

-----Examples-----
Text: "The film The Lord of the Rings: The Return of the King (2003) was directed by Peter Jackson. It won eleven Academy Awards, including Best Picture. "
Output:
("triple"{tuple_delimiter}"The Lord of the Rings: The Return of the King"{tuple_delimiter}"released in"{tuple_delimiter}"2003"){record_delimiter}
("triple"{tuple_delimiter}"The Lord of the Rings: The Return of the King"{tuple_delimiter}"was directed by"{tuple_delimiter}"Peter Jackson"){record_delimiter}
("triple"{tuple_delimiter}"The Lord of the Rings: The Return of the King"{tuple_delimiter}"won"{tuple_delimiter}"eleven Academy Awards"){record_delimiter}
("triple"{tuple_delimiter}"The Lord of the Rings: The Return of the King"{tuple_delimiter}"won"{tuple_delimiter}"Best Picture"){record_delimiter}
{completion_delimiter}

Text: "Apple Inc.(AAPL) was founded by Steve Jobs in 1976. He later launched the iPhone, which became one of the most successful products in the world. Steve Jobs had been the CEO of Apple Inc. since 1997."
Output:
("triple"{tuple_delimiter}"Apple Inc."{tuple_delimiter}"also known as"{tuple_delimiter}"AAPL"){record_delimiter}
("triple"{tuple_delimiter}"Apple Inc."{tuple_delimiter}"was founded by"{tuple_delimiter}"Steve Jobs"){record_delimiter}
("triple"{tuple_delimiter}"Apple Inc."{tuple_delimiter}"was founded in year"{tuple_delimiter}"1976"){record_delimiter}
("triple"{tuple_delimiter}"Steve Jobs"{tuple_delimiter}"launched"{tuple_delimiter}"iPhone"){record_delimiter}
("triple"{tuple_delimiter}"iPhone"{tuple_delimiter}"became"{tuple_delimiter}"one of the most successful products in the world"){record_delimiter}
("triple"{tuple_delimiter}"Steve Jobs"{tuple_delimiter}"had been the CEO of"{tuple_delimiter}"Apple Inc."){record_delimiter}
("triple"{tuple_delimiter}"Steve Jobs"{tuple_delimiter}"had been the CEO of Apple Inc. since"{tuple_delimiter}"1997"){record_delimiter}
{completion_delimiter}

Text: "Marie Curie discovered radium and polonium, which are radioactive elements. She won two Nobel Prizes for her work in physics and chemistry."
Output:
("triple"{tuple_delimiter}"Marie Curie"{tuple_delimiter}"discovered"{tuple_delimiter}"radium"){record_delimiter}
("triple"{tuple_delimiter}"Marie Curie"{tuple_delimiter}"discovered"{tuple_delimiter}"polonium"){record_delimiter}
("triple"{tuple_delimiter}"radium"{tuple_delimiter}"is"{tuple_delimiter}"a radioactive element"){record_delimiter}
("triple"{tuple_delimiter}"polonium"{tuple_delimiter}"is"{tuple_delimiter}"a radioactive element"){record_delimiter}
("triple"{tuple_delimiter}"Marie Curie"{tuple_delimiter}"won"{tuple_delimiter}"two Nobel Prizes"){record_delimiter}
("triple"{tuple_delimiter}"two Nobel Prizes"{tuple_delimiter}"awarded for"{tuple_delimiter}"Marie Curie's work in physics"){record_delimiter}
("triple"{tuple_delimiter}"two Nobel Prizes"{tuple_delimiter}"awarded for"{tuple_delimiter}"Marie Curie's work in chemistry"){record_delimiter}
{completion_delimiter}

-----Input-----
Now, read the following text carefully and output the result using the specified format only.
Text:
{text}

Figure A2. Triple Extraction Prompt (Part 2).

Named Entity Recognition Prompt

-----Role-----
You are an expert in knowledge extraction and knowledge graph reasoning.
Your task is to extract two categories of entities from the input text:
1. Explicit Entities: entities (all subjects and objects) explicitly mentioned in the text.
2. Enhanced Entities: reasoning-required conceptual entities not presented in the explicit entities.
You must output the two categories separately, each as one continuous sequence.

-----Rules for Explicit Entities-----
- Extract explicit entities belonging only to:
  [person, organization, location, facility, product, event, time, award, scientific_concept, number, role].
- Preserve the accurate wording in the original text with no paraphrasing.
- Resolve pronouns and descriptive references into full explicit entities.
- Keep entities minimal yet complete.
- Extract all distinct explicit entities.
- Output format: each item in quotes, separated by {record_delimiter} and terminated with {completion_del imiter}.

-----Rules for Enhanced Entities-----
- Extract up to 3 enhanced conceptual keywords needed to reason about the question.
- Must be nouns or noun phrases, concise, complete, aligned with graph entities.
- Do not use overly broad categories such as "organization", "person", "event", "location".
- Do not repeat any explicit entity.
- Convert relational verbs into conceptual forms when necessary (e.g., "directed" → "director").
- Output format: each item in quotes, separated by {record_delimiter} and terminated with {completion_del imiter}.

-----Output Format (Strict)-----
Explicit Entities:
"entity1"{record_delimiter}"entity2"{record_delimiter}...{completion_delimiter}
Enhanced Entities:
"concept1"{record_delimiter}"concept2"{record_delimiter}...{completion_delimiter}
No JSON. No explanations. No additional text.

-----Examples-----
Text: "Who directed the film The Lord of the Rings: The Return of the King? (2003) that won eleven Acad emy Awards?"
Explicit Entities:
"The Lord of the Rings: The Return of the King"{record_delimiter}"2003"{record_delimiter}"eleven Acad emy Awards"{record_delimiter}{completion_delimiter}
Enhanced Entities:
"director"{record_delimiter}{completion_delimiter}

Text: "Which institution awarded Marie Curie the Nobel Prizes for her discovery of the radioactive element s radium and polonium?"
Explicit Entities:
"Marie Curie"{record_delimiter}"Nobel Prizes"{record_delimiter}radioactive elements"{record_delimiter}"radium"{record_delimiter}"polonium"{record_delimiter}{completion_delimiter}
Enhanced Entities:
"awarding institution"{record_delimiter}"discovery"{record_delimiter}{completion_delimiter}

Text: "Which American band from Knoxville Tennessee released the album From Birth to Burial?"
Explicit Entities:
"Knoxville Tennessee"{record_delimiter}"From Birth to Burial"{record_delimiter}{completion_delimiter}
Enhanced Entities:
"American band"{record_delimiter}"album release"{record_delimiter}{completion_delimiter}

-----Input Text-----
{text}

Figure A3. Named Entity Recognition Prompt.

Implicit Node Instantiation Prompt

---Role---
You are an expert in knowledge graph reasoning.
Your task is to analyze the question together with the provided entities and retrieved triples, and identify up to three additional entities that are essential for answering the question.

---Goal---
Find at most three entities that:
1. Must come directly and exactly from the subjects or objects in the provided triples. Only subjects and objects could be considered.
2. Do not appear in the question or provided entity list.
3. Are highly relevant to the reasoning chain required for answering the question.

---Priority Rules (in strict order)---
When deciding which entities are essential, apply the following importance criteria:

1. **Multi-hop equivalence priority**
   If the question implicitly refers to an entity via a description or pronoun (e.g., "the person who discovered radium"),
   and a triple provides the concrete entity (e.g., ['Marie Curie', 'discovered', 'radium']),
   then that concrete entity (e.g., "Marie Curie") is considered an essential reasoning anchor and should be selected first.
   In this case, you must always select the concrete entity that fulfills the role implied in the question, not the object they relate to.

2. **Key-path nodes**
   Entities that lie on the essential reasoning chain required to obtain the answer.

3. **Higher connectivity**
   Entities appearing in multiple triples or both subject and object roles.

4. **Higher frequency**
   Entities appearing more frequently across triples.

---Input Data---
Question:
{user_query}

Provided Entities:
{entities}

Retrieved Triples:
{triples}

---Output Rules---
1. Output no more than three entities.
2. If multiple candidate entities could be chosen, prefer the most concise valid entity.
3. Entities must be taken exactly as they appear in the triples:
   - Do not shorten, paraphrase, or partially extract any entity.
   - Output only complete entity strings exactly as they occur.
4. Do not output entities already in the provided entities list.
5. Do not explain your reasoning.

---Output Format---
"<entity1>"{record_delimiter}"<entity2>"{record_delimiter}
Followed by:
{completion_delimiter}

Figure A4. Implicit Node Instantiation Prompt.

References

1. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
2. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.

3. Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A. Training language models to follow instructions with human feedback. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 27730–27744.
4. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z. A survey of large language models. *arXiv* **2023**, arXiv:2303.18223.
5. Sun, H.; Bedrax-Weiss, T.; Cohen, W. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 2380–2390.
6. Rasool, Z.; Kurniawan, S.; Balugo, S.; Barnett, S.; Vasa, R.; Chesser, C.; Hampstead, B.M.; Belleville, S.; Mouzakis, K.; Bahar-Fuchs, A. Evaluating llms on document-based qa: Exact answer selection and numerical extraction using cogtale dataset. *Nat. Lang. Process. J.* **2024**, *8*, 100083. [CrossRef]
7. Lu, Y.; Liu, Q.; Dai, D.; Xiao, X.; Lin, H.; Han, X.; Sun, L.; Wu, H. Unified structure generation for universal information extraction. *arXiv* **2022**, arXiv:2203.12277. [CrossRef]
8. Keloth, V.K.; Hu, Y.; Xie, Q.; Peng, X.; Wang, Y.; Zheng, A.; Selek, M.; Raja, K.; Wei, C.H.; Jin, Q. Advancing entity recognition in biomedicine via instruction tuning of large language models. *Bioinformatics* **2024**, *40*, btae163. [CrossRef] [PubMed]
9. Nakshatri, N.; Liu, S.; Chen, S.; Roth, D.; Goldwasser, D.; Hopkins, D. Using LLM for improving key event discovery: Temporal-guided news stream clustering with event summaries. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023; pp. 4162–4173.
10. Zhang, H.; Yu, P.S.; Zhang, J. A systematic survey of text summarization: From statistical methods to large language models. *ACM Comput. Surv.* **2025**, *57*, 1–41. [CrossRef]
11. Li, J.; Zhou, H.; Huang, S.; Cheng, S.; Chen, J. Eliciting the translation ability of large language models via multilingual finetuning with translation instructions. *Trans. Assoc. Comput. Linguist.* **2024**, *12*, 576–592. [CrossRef]
12. Zhu, W.; Liu, H.; Dong, Q.; Xu, J.; Huang, S.; Kong, L.; Chen, J.; Li, L. Multilingual machine translation with large language models: Empirical results and analysis. In Proceedings of the Findings of the association for computational linguistics: NAACL 2024, Mexico City, Mexico, 16–21 June 2024; pp. 2765–2781.
13. Kojima, T.; Gu, S.S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large language models are zero-shot reasoners. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 22199–22213.
14. Budnikov, M.; Bykova, A.; Yamshchikov, I.P. Generalization potential of large language models. *Neural Comput. Appl.* **2025**, *37*, 1973–1997. [CrossRef]
15. Shuster, K.; Poff, S.; Chen, M.; Kiela, D.; Weston, J. Retrieval augmentation reduces hallucination in conversation. *arXiv* **2021**, arXiv:2104.07567. [CrossRef]
16. Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.J.; Madotto, A.; Fung, P. Survey of hallucination in natural language generation. *ACM Comput. Surv.* **2023**, *55*, 1–38. [CrossRef]
17. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9459–9474.
18. Izacard, G.; Grave, E. Leveraging passage retrieval with generative models for open domain question answering. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, Kyiv, Ukraine, 19–23 April 2021; pp. 874–880.
19. James, A.; Trovati, M.; Bolton, S. Retrieval-Augmented Generation to Generate Knowledge Assets and Creation of Action Drivers. *Appl. Sci.* **2025**, *15*, 6247. [CrossRef]
20. Zhang, Q.; Chen, S.; Bei, Y.; Yuan, Z.; Zhou, H.; Hong, Z.; Chen, H.; Xiao, Y.; Zhou, C.; Dong, J. A survey of graph retrieval-augmented generation for customized large language models. *arXiv* **2025**, arXiv:2501.13958.
21. Pan, S.; Luo, L.; Wang, Y.; Chen, C.; Wang, J.; Wu, X. Unifying large language models and knowledge graphs: A roadmap. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 3580–3599. [CrossRef]
22. Hogan, A.; Blomqvist, E.; Cochez, M.; d'Amato, C.; Melo, G.D.; Gutierrez, C.; Kirrane, S.; Gayo, J.E.L.; Navigli, R.; Neumaier, S. Knowledge graphs. *ACM Comput. Surv. Csur* **2021**, *54*, 1–37.
23. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Yu, P.S. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 494–514. [CrossRef] [PubMed]
24. Li, Z.; Guo, Q.; Shao, J.; Song, L.; Bian, J.; Zhang, J.; Wang, R. Graph neural network enhanced retrieval for question answering of llms. *arXiv* **2024**, arXiv:2406.06572.
25. Yang, W.; Some, L.; Bain, M.; Kang, B. A comprehensive survey on integrating large language models with knowledge-based methods. *Knowl.-Based Syst.* **2025**, *318*, 113503. [CrossRef]
26. Peng, B.; Zhu, Y.; Liu, Y.; Bo, X.; Shi, H.; Hong, C.; Zhang, Y.; Tang, S. Graph retrieval-augmented generation: A survey. *ACM Trans. Inf. Syst.* **2025**, *44*, 1–52. [CrossRef]

27. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H.; Wang, H. Retrieval-augmented generation for large language models: A survey. *arXiv* **2023**, arXiv:2312.10997.
28. Menick, J.; Trebacz, M.; Mikulik, V.; Aslanides, J.; Song, F.; Chadwick, M.; Glaese, M.; Young, S.; Campbell-Gillingham, L.; Irving, G. Teaching language models to support answers with verified quotes. *arXiv* **2022**, arXiv:2203.11147. [CrossRef]
29. Izacard, G.; Lewis, P.; Lomeli, M.; Hosseini, L.; Petroni, F.; Schick, T.; Dwivedi-Yu, J.; Joulin, A.; Riedel, S.; Grave, E. Few-shot learning with retrieval augmented language models. *arXiv* **2022**, arXiv:2208.03299. [CrossRef]
30. Fan, W.; Ding, Y.; Ning, L.; Wang, S.; Li, H.; Yin, D.; Chua, T.-S.; Li, Q. A survey on rag meeting llms: Towards retrieval-augmented large language models. In Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Barcelona, Spain, 25–29 August 2024; pp. 6491–6501.
31. Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.S.; Wu, L.; Edunov, S.; Chen, D.; Yih, W.-t. Dense Passage Retrieval for Open-Domain Question Answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Punta Cana, Dominican Republic, 16–20 November 2020; pp. 6769–6781.
32. Robertson, S.; Zaragoza, H. *The Probabilistic Relevance Framework: BM25 and Beyond*; Now Publishers Inc.: Norwell, MA, USA, 2009; Volume 4.
33. Santhanam, K.; Khattab, O.; Saad-Falcon, J.; Potts, C.; Zaharia, M. Colbertv2: Effective and efficient retrieval via lightweight late interaction. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, WA, USA, 10–15 July 2022; pp. 3715–3734.
34. Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M. Retrieval augmented language model pre-training. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 12–18 July 2020; pp. 3929–3938.
35. Shi, W.; Min, S.; Yasunaga, M.; Seo, M.; James, R.; Lewis, M.; Zettlemoyer, L.; Yih, W.-t. Replug: Retrieval-augmented black-box language models. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), Mexico City, Mexico, 16–21 June 2024; pp. 8371–8384.
36. Guo, T.; Yang, Q.; Wang, C.; Liu, Y.; Li, P.; Tang, J.; Li, D.; Wen, Y. Knowledgenavigator: Leveraging large language models for enhanced reasoning over knowledge graph. *Complex Intell. Syst.* **2024**, *10*, 7063–7076. [CrossRef]
37. Linders, J.; Tomczak, J.M. Knowledge graph-extended retrieval augmented generation for question answering. *Appl. Intell.* **2025**, *55*, 1102. [CrossRef]
38. Soman, K.; Rose, P.W.; Morris, J.H.; E Akbas, R.; Smith, B.; Peetoom, B.; Villouta-Reyes, C.; Cerono, G.; Shi, Y.; Rizk-Jackson, A. Biomedical knowledge graph-enhanced prompt generation for large language models. *arXiv* **2023**, arXiv:2311.17330. [CrossRef]
39. Li, M.; Miao, S.; Li, P. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. *arXiv* **2024**, arXiv:2410.20724.
40. Gao, F.; Xu, S.; Hao, W.; Lu, T. KA-RAG: Integrating Knowledge Graphs and Agentic Retrieval-Augmented Generation for an Intelligent Educational Question-Answering Model. *Appl. Sci.* **2025**, *15*, 12547. [CrossRef]
41. Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; Metropolitansky, D.; Ness, R.O.; Larson, J. From local to global: A graph rag approach to query-focused summarization. *arXiv* **2024**, arXiv:2404.16130. [CrossRef]
42. Guo, Z.; Xia, L.; Yu, Y.; Ao, T.; Huang, C. Lightrag: Simple and fast retrieval-augmented generation. *arXiv* **2024**, arXiv:2410.05779.
43. Gutiérrez, B.J.; Shu, Y.; Qi, W.; Zhou, S.; Su, Y. From rag to memory: Non-parametric continual learning for large language models. *arXiv* **2025**, arXiv:2502.14802. [CrossRef]
44. Jimenez Gutierrez, B.; Shu, Y.; Gu, Y.; Yasunaga, M.; Su, Y. Hipporag: Neurobiologically inspired long-term memory for large language models. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 59532–59569.
45. Jiang, X.; Zhang, R.; Xu, Y.; Qiu, R.; Fang, Y.; Wang, Z.; Tang, J.; Ding, H.; Chu, X.; Zhao, J. Hykge: A hypothesis knowledge graph enhanced rag framework for accurate and reliable medical llms responses. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vienna, Austria, 27 July–1 August 2025; pp. 11836–11856.
46. Sun, J.; Xu, C.; Tang, L.; Wang, S.; Lin, C.; Gong, Y.; Ni, L.M.; Shum, H.-Y.; Guo, J. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. *arXiv* **2023**, arXiv:2307.07697.
47. Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; Liu, Z. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv* **2024**, arXiv:2402.03216.
48. Hui, B.; Yang, J.; Cui, Z.; Yang, J.; Liu, D.; Zhang, L.; Liu, T.; Zhang, J.; Yu, B.; Lu, K. Qwen2. 5-coder technical report. *arXiv* **2024**, arXiv:2409.12186.
49. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K. Natural questions: A benchmark for question answering research. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 453–466. [CrossRef]
50. Wang, Y.; Ren, R.; Li, J.; Zhao, W.X.; Liu, J.; Wen, J.-R. Rear: A relevance-aware retrieval-augmented framework for open-domain question answering. *arXiv* **2024**, arXiv:2402.17497.
51. Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.; Salakhutdinov, R.; Manning, C.D. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2369–2380.

52. Ho, X.; Nguyen, A.-K.D.; Sugawara, S.; Aizawa, A. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv* **2020**, arXiv:2011.01060. [CrossRef]
53. Trivedi, H.; Balasubramanian, N.; Khot, T.; Sabharwal, A. ♫ MuSiQue: Multihop Questions via Single-hop Question Composition. *Trans. Assoc. Comput. Linguist.* **2022**, *10*, 539–554. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.