Graph Retrieval-Augmented Generation: A Survey

BOCI PENG*, School of Intelligence Science and Technology, Peking University, China
YUN ZHU*, College of Computer Science and Technology, Zhejiang University, China
YONGCHAO LIU, Ant Group, China
XIAOHE BO, Gaoling School of Artificial Intelligence, Renmin University of China, China
HAIZHOU SHI, Rutgers University, US
CHUNTAO HONG, Ant Group, China
YAN ZHANG†, School of Intelligence Science and Technology, Peking University, China
SILIANG TANG, College of Computer Science and Technology, Zhejiang University, China

Recently, Retrieval-Augmented Generation (RAG) has achieved remarkable success in addressing the challenges of Large Language Models (LLMs) without necessitating retraining. By referencing an external knowledge base, RAG refines LLM outputs, effectively mitigating issues such as "hallucination", lack of domain-specific knowledge, and outdated information. However, the complex structure of relationships among different entities in databases presents challenges for RAG systems. In response, GraphRAG leverages structural information across entities to enable more precise and comprehensive retrieval, capturing relational knowledge and facilitating more accurate, context-aware responses. Given the novelty and potential of GraphRAG, a systematic review of current technologies is imperative. This paper provides the first comprehensive overview of GraphRAG methodologies. We formalize the GraphRAG workflow, encompassing Graph-Based Indexing, Graph-Guided Retrieval, and Graph-Enhanced Generation. We then outline the core technologies and training methods at each stage. Additionally, we examine downstream tasks, application domains, evaluation methodologies, and industrial use cases of GraphRAG. Finally, we explore future research directions to inspire further inquiries and advance progress in the field.

CCS Concepts: • **Computing methodologies** → **Knowledge representation and reasoning**; • **Information systems** → **Information retrieval**; *Data mining*.

Additional Key Words and Phrases: Large Language Models, Graph Retrieval-Augmented Generation, Knowledge Graphs, Graph Neural Networks

1 Introduction

The development of Large Language Models like GPT-4 [116], Qwen2 [170], and LLaMA [24] has sparked a revolution in the field of artificial intelligence, fundamentally altering the landscape of natural language processing. These models, built on Transformer [149] architectures and trained on diverse and extensive datasets, have demonstrated unprecedented capabilities in understanding, interpreting, and generating human language. The impact of these advancements is profound, stretching across various sectors including healthcare [93, 154, 188], finance [84, 114], and education [38, 157], where they facilitate more nuanced and efficient interactions between humans and machines.

*Both authors contributed equally to this research.
†Corresponding Author.

Authors' Contact Information: Boci Peng, School of Intelligence Science and Technology, Peking University, Beijing, China, bcpeng@stu.pku.edu.cn; Yun Zhu, College of Computer Science and Technology, Zhejiang University, Hangzhou, China, zhuyun_dcd@zju.edu.cn; Yongchao Liu, Ant Group, Hangzhou, China, yongchao.ly@antgroup.com; Xiaohe Bo, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China, bellebxh@gmail.com; Haizhou Shi, Rutgers University, New Brunswick, New Jersey, US, haizhou.shi@rutgers.edu; Chuntao Hong, Ant Group, Hangzhou, China, chuntao.hct@antgroup.com; Yan Zhang, School of Intelligence Science and Technology, Peking University, Beijing, China, zhyzhy001@pku.edu.cn; Siliang Tang, College of Computer Science and Technology, Zhejiang University, Hangzhou, China, siliang@zju.edu.cn.

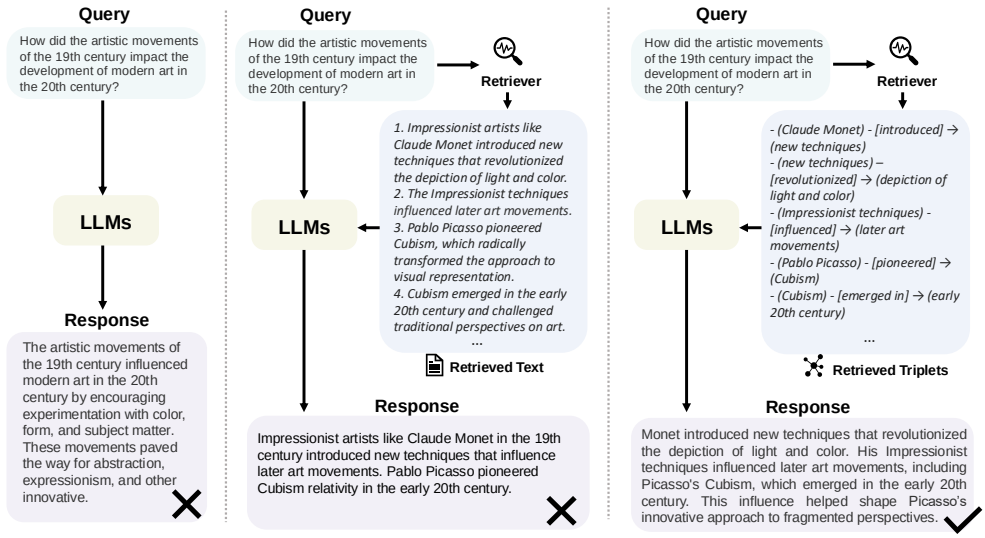

Fig. 1. Comparision between Direct LLM, RAG, and GraphRAG. Given a user query, direct answering by LLMs may suffer from shallow responses or lack of specificity. RAG addresses this by retrieving relevant textual information, somewhat alleviating the issue. However, due to the text's length and flexible natural language expressions of entity relationships, RAG struggles to emphasize "influence" relations, which is the core of the question. While, GraphRAG methods leverage explicit entity and relationship representations in graph data, enabling precise answers by retrieving relevant structured information.

Despite their remarkable language comprehension and text generation capabilities, LLMs may exhibit limitations due to a lack of domain-specific knowledge, real-time updated information, and proprietary knowledge, which are outside LLMs' pre-training corpus. These gaps can lead to a phenomenon known as "hallucination" [53] where the model generates inaccurate or even fabricated information. Consequently, it is imperative to supplement LLMs with external knowledge to mitigate this problem. Retrieval-Augmented Generation (RAG) [27, 37, 51, 54, 165, 180, 187] emerged as a significant evolution, which aims to enhance the quality and relevance of generated content by integrating a retrieval component within the generation process. The essence of RAG lies in its ability to dynamically query a large text corpus to incorporate relevant factual knowledge into the responses generated by the underlying language models. This integration not only enriches the contextual depth of the responses but also ensures a higher degree of factual accuracy and specificity. RAG has gained widespread attention due to its exceptional performance and broad applications, becoming a key focus within the field.

Although RAG has achieved impressive results and has been widely applied across various domains, it faces limitations in real-world scenarios: (1) *Neglecting Relationships*: In practice, textual content is not isolated but interconnected. Traditional RAG fails to capture significant structured relational knowledge that cannot be represented through semantic similarity alone. For instance, in a citation network where papers are linked by citation relationships, traditional RAG methods focus on finding the relevant papers based on the query but overlook important citation relationships between papers. (2) *Redundant Information*: RAG often recounts content in the form of textual snippets when concatenated as prompts. This makes context become excessively lengthy, leading to the "lost in the middle" dilemma [94]. (3) *Lacking Global Information*: RAG can only retrieve a

subset of documents and fails to grasp global information comprehensively, and hence struggles with tasks such as Query-Focused Summarization (QFS).

Graph Retrieval-Augmented Generation (GraphRAG) [25, 50, 108] emerges as an innovative solution to address these challenges. Unlike traditional RAG, GraphRAG retrieves graph elements containing relational knowledge pertinent to a given query from a pre-constructed graph database, as depicted in Figure 1. These elements may include nodes, triples, paths, or subgraphs, which are utilized to generate responses. GraphRAG considers the interconnections between texts, enabling a more accurate and comprehensive retrieval of relational information. Additionally, graph data, such as knowledge graphs, offer abstraction and summarization of textual data, thereby significantly shortening the length of the input text and mitigating concerns of verbosity. By retrieving subgraphs or graph communities, we can access comprehensive information to effectively address the QFS challenge by capturing the broader context and interconnections within the graph structure.

In this paper, we are the first to provide a systematic survey of GraphRAG. Specifically, we begin by introducing the GraphRAG workflow, along with the foundational background knowledge that underpins the field. Then, we categorize the literature according to the primary stages of the GraphRAG process: Graph-Based Indexing (G-Indexing), Graph-Guided Retrieval (G-Retrieval), and Graph-Enhanced Generation (G-Generation) in Section 5, Section 6 and Section 7 respectively, detailing the core technologies and training methods within each phase. Furthermore, we investigate downstream tasks, application domains, evaluation methodologies, and industrial use cases of GraphRAG. This exploration elucidates how GraphRAG is being utilized in practical settings and reflects its versatility and adaptability across various sectors. Finally, acknowledging that research in GraphRAG is still in its early stages, we delve into potential future research directions. This prognostic discussion aims to pave the way for forthcoming studies, inspire new lines of inquiry, and catalyze progress within the field, ultimately propelling GraphRAG toward more mature and innovative horizons.

Our contributions can be summarized as follows:

- We provide a comprehensive and systematic review of existing state-of-the-art GraphRAG methodologies. We offer a formal definition of GraphRAG, outlining its universal workflow which includes G-Indexing, G-Retrieval, and G-Generation.
- We discuss the core technologies underpinning existing GraphRAG systems, including G-Indexing, G-Retrieval, and G-Generation. For each component, we analyze the spectrum of model selection, methodological design, and enhancement strategies currently being explored. Additionally, we contrast the diverse training methodologies employed across these modules.
- We delineate the downstream tasks, benchmarks, application domains, evaluation metrics, current challenges, and future research directions pertinent to GraphRAG, discussing both the progress and prospects of this field. Furthermore, we compile an inventory of existing industry GraphRAG systems, providing insights into the translation of academic research into real-world industry solutions.

Organization. The rest of the survey is organized as follows: Section 2 compares related techniques, while Section 3 outlines the general process of GraphRAG. Sections 5 to 7 categorize the techniques associated with GraphRAG's three stages: G-Indexing, G-Retrieval, and G-Generation. Section 8 introduces the training strategies of retrievers and generators. Section 9 summarizes GraphRAG's downstream tasks, corresponding benchmarks, application domains, evaluation metrics, and industrial GraphRAG systems. Section 10 provides an outlook on future directions. Finally, Section 11 concludes the content of this survey.

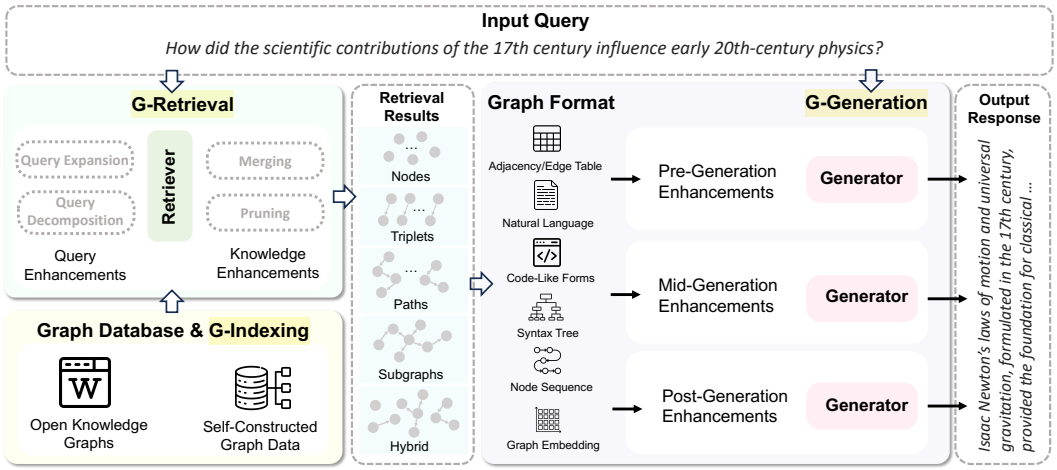

Fig. 2. The overview of the GraphRAG framework for question answering task. In this survey, we divide GraphRAG into three stages: G-Indexing, G-Retrieval, and G-Generation. We categorize the retrieval sources into open-source knowledge graphs and self-constructed graph data. Various enhancing techniques like query enhancement and knowledge enhancement may be adopted to boost the relevance of the results. Unlike RAG, which uses retrieved text directly for generation, GraphRAG requires converting the retrieved graph information into patterns acceptable to generators to enhance the task performance.

2 Comparison with Related Techniques and Surveys

In this section, we compare Graph Retrieval-Augmented Generation (GraphRAG) with related techniques and corresponding surveys, including RAG, LLMs on graphs, and Knowledge Base Question Answering (KBQA).

2.1 RAG

RAG combines external knowledge with LLMs for improved task performance, integrating domain-specific information to ensure factuality and credibility. In the past two years, researchers have written many comprehensive surveys about RAG [27, 37, 51, 54, 165, 180, 187]. For example, Fan et al. [27] and Gao et al. [37] categorize RAG methods from the perspectives of retrieval, generation, and augmentation. Zhao et al. [187] review RAG methods for databases with different modalities. Yu et al. [180] systematically summarize the evaluation of RAG methods. These works provide a structured synthesis of current RAG methodologies, fostering a deeper understanding and suggesting future directions of the area.

From a broad perspective, GraphRAG can be seen as a branch of RAG, which retrieves relevant relational knowledge from graph databases instead of text corpus. However, compared to text-based RAG, GraphRAG takes into account the relationships between texts and incorporates the structural information as additional knowledge beyond text. Furthermore, during the construction of graph data, raw text data may undergo filtering and summarization processes, enhancing the refinement of information within the graph data. Although previous surveys on RAG have touched upon GraphRAG, they predominantly center on textual data integration. This paper diverges by placing a primary emphasis on the indexing, retrieval, and utilization of structured graph data, which represents a substantial departure from handling purely textual information and spurs the emergence of many new techniques.

2.2 LLMs on Graphs

LLMs are revolutionizing natural language processing due to their excellent text understanding, reasoning, and generation capabilities, along with their generalization and zero-shot transfer abilities. Although LLMs are primarily designed to process pure text and struggle with non-Euclidean data containing complex structural information, such as graphs [41, 153], numerous studies [13, 28, 65, 83, 92, 105, 119, 120, 161, 189] have been conducted in these fields. These papers primarily integrate LLMs with GNNs to enhance modeling capabilities for graph data, thereby improving performance on downstream tasks such as node classification, edge prediction, graph classification, and others. For example, Zhu et al. [189] propose an efficient fine-tuning method named ENGINE, which combines LLMs and GNNs through a side structure for enhancing graph representation.

Different from these methods, GraphRAG focuses on retrieving relevant graph elements using queries from an external graph-structured database. In this paper, we provide a detailed introduction to the relevant technologies and applications of GraphRAG, which are not included in previous surveys of LLMs on Graphs.

2.3 KBQA

KBQA is a significant task in natural language processing, aiming to respond to user queries based on external knowledge bases [33, 76, 77, 174], thereby achieving goals such as fact verification, passage retrieval enhancement, and text understanding. Previous surveys typically categorize existing KBQA approaches into two main types: Information Retrieval (IR)-based methods and Semantic Parsing (SP)-based methods. Specifically, IR-based methods [60, 61, 102, 142, 155, 168, 181] retrieve information related to the query from the knowledge graph (KG) and use it to enhance the generation process. While SP-based methods [12, 15, 29, 40, 141, 177] generate a logical form (LF) for each query and execute it against knowledge bases to obtain the answer.

GraphRAG and KBQA are closely related, with IR-based KBQA methods representing a subset of GraphRAG approaches focused on downstream applications. In this work, we extend the discussion beyond KBQA to include GraphRAG's applications across various downstream tasks. Our survey provides a thorough and detailed exploration of GraphRAG technology, offering a comprehensive understanding of existing methods and potential improvements.

3 Preliminaries

In this section, we introduce background knowledge of GraphRAG for easier comprehension of our survey. First, we introduce Text-Attributed Graphs which is a universal and general format of graph data used in GraphRAG. Then, we provide formal definitions for two types of models that can be used in the retrieval and generation stages: Graph Neural Networks and Language Models.

3.1 Text-Attributed Graphs

The graph data used in Graph RAG can be represented uniformly as Text-Attributed Graphs (TAGs), where nodes and edges possess textual attributes. Formally, a text-attributed graph can be denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \{\mathbf{x}_v\}_{v\in\mathcal{V}}, \{\mathbf{e}_{i,j}\}_{i,j\in\mathcal{E}})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, $\mathcal{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the adjacent matrix. Additionally, $\{\mathbf{x}_v\}_{v\in\mathcal{V}}$ and $\{\mathbf{e}_{i,j}\}_{i,j\in\mathcal{E}}$ are textual attributes of nodes and edges, respectively. One typical kind of TAGs is Knowledge Graphs (KGs), where nodes are entities, edges are relations among entities, and text attributes are the names of entities and relations.

3.2 Graph Neural Networks

Graph Neural Networks (GNNs) are a kind of deep learning framework to model the graph data. Classical GNNs, e.g., GCN [74], GAT [150], GraphSAGE [44], adopt a message-passing manner to obtain node representations. Formally, each node representation $\mathbf{h}_i^{(l-1)}$ in the l-th layer is updated by aggregating the information from neighboring nodes and edges:

$$\mathbf{h}_i^{(l)} = \mathbf{UPD}(\mathbf{h}_i^{(l-1)}, \mathbf{AGG}_{j\in\mathcal{N}(i)}\mathbf{MSG}(\mathbf{h}_i^{(l-1)}, \mathbf{h}_j^{(l-1)}, \mathbf{e}_{i,j}^{(l-1)})), \tag{1}$$

where $\mathcal{N}_{(i)}$ represents the neighbors of node i. **MSG** denotes the message function, which computes the message based on the node, its neighbor, and the edge between them. **AGG** refers to the aggregation function that combines the received messages using a permutation-invariant method, such as mean, sum, or max. **UPD** represents the update function, which updates each node's attributes with the aggregated messages.

Subsequently, a readout function, e.g., mean, sum, or max pooling, can be applied to obtain the global-level representation:

$$\mathbf{h}_G = \mathbf{READOUT}_{i\in\mathcal{V}_G}(\mathbf{h}_i^{(L)}) \tag{2}$$

In GraphRAG, GNNs can be utilized to obtain representations of graph data for the retrieval phase, as well as to model the retrieved graph structures.

3.3 Language Models

Language models (LMs) excel in language understanding and are mainly classified into two types: discriminative and generative. Discriminative models, like BERT [22], RoBERTa [97] and Sentence-BERT [129], focus on estimating the conditional probability $P(\mathbf{y}|\mathbf{x})$ and are effective in tasks such as text classification and sentiment analysis. In contrast, generative models, including GPT-3 [10] and GPT-4 [116], aim to model the joint probability $P(\mathbf{x}, \mathbf{y})$ for tasks like machine translation and text generation. These generative pre-trained models have significantly advanced the field of natural language processing (NLP) by leveraging massive datasets and billions of parameters, contributing to the rise of Large Language Models (LLMs) with outstanding performance across various tasks.

In the early stages, RAG and GraphRAG focused on improving pre-training techniques for discriminative language models [22, 97, 129]. Recently, LLMs such as ChatGPT [117], LLaMA [24], and Qwen2 [170] have shown great potential in language understanding, demonstrating powerful in-context learning capabilities. Subsequently, research on RAG and GraphRAG shifted towards enhancing information retrieval for language models, addressing increasingly complex tasks and mitigating hallucinations, thereby driving rapid advancements in the field.

4 Overview of GraphRAG

GraphRAG is a framework that leverages external structured knowledge graphs to improve contextual understanding of LMs and generate more informed responses, as depicted in Figure 2. The goal of GraphRAG is to retrieve the most relevant knowledge from databases, thereby enhancing the answers of downstream tasks. The process can be defined as

$$a^* = \arg\max_{a\in A} p(a|q, \mathcal{G}), \tag{3}$$

TAG - text attributed graph

where a^* is the optimal answer of the query q given the TAG $\mathcal{G}$, and A is the set of possible responses. After that, we jointly model the target distribution $p(a|q, \mathcal{G})$ with a graph retriever $p_\theta(G|q, \mathcal{G})$ and an answer generator $p_\phi(a|q, G)$ where θ, ϕ are learnable parameters, and utilize the

decompose the target distribution

total probability formula to decompose $p(a|q, \mathcal{G})$, which can be formulated as

$$\begin{aligned} p(a|q, \mathcal{G}) &= \sum_{G \subseteq \mathcal{G}} p_\phi(a|q, G) p_\theta(G|q, \mathcal{G}) \\ &\approx p_\phi(a|q, G^*) p_\theta(G^*|q, \mathcal{G}), \end{aligned} \tag{4}$$

where G^* is the optimal subgraph. Because the number of candidate subgraphs can grow exponentially with the size of the graph, efficient approximation methods are necessary. The first line of Equation 4 is thus approximated by the second line. Specifically, a graph retriever is employed to extract the optimal subgraph G^*, after which the generator produces the answer based on the retrieved subgraph.

Therefore, in this survey, we decompose the entire process of GraphRAG into three main stages: Graph-Based Indexing, Graph-Guided Retrieval, and Graph-Enhanced Generation. The overall workflow of GraphRAG is illustrated in Figure 2 and detailed introductions of each stage are as follows.

Graph-Based Indexing (G-Indexing). Graph-Based Indexing constitutes the initial phase of GraphRAG, aimed at identifying or constructing a graph database $\mathcal{G}$ that aligns with downstream tasks and establishing indices on it. The graph database can originate from public knowledge graphs [2, 7, 91, 131, 138, 151], graph data [112], or be constructed based on proprietary data sources such as textual [25, 43, 80, 160] or other forms of data [169]. The indexing process typically includes mapping node and edge properties, establishing pointers between connected nodes, and organizing data to support fast traversal and retrieval operations. Indexing determines the granularity of the subsequent retrieval stage, playing a crucial role in enhancing query efficiency.

Graph-Guided Retrieval (G-Retrieval). Following graph-based indexing, the graph-guided retrieval phase focuses on extracting pertinent information from the graph database in response to user queries or input. Specifically, given a user query q which is expressed in natural language, the retrieval stage aims to extract the most relevant elements (e.g., entities, triplets, paths, subgraphs) from knowledge graphs, which can be formulated as

$$\begin{aligned} G^* &= \textbf{G-Retriever}(q, \mathcal{G}) \\ &= \arg\max_{G \subseteq \mathcal{R}(\mathcal{G})} p_\theta(G|q, \mathcal{G}) \\ &= \arg\max_{G \subseteq \mathcal{R}(\mathcal{G})} \textbf{Sim}(q, G), \end{aligned} \tag{5}$$

where G^* is the optimal retrieved graph elements and $\textbf{Sim}(\cdot, \cdot)$ is a function that measures the semantic similarity between user queries and the graph data. $\mathcal{R}(\cdot)$ represents a function to narrow down the search range of subgraphs, considering the efficiency.

Graph-Enhanced Generation (G-Generation). The graph-enhanced generation phase involves synthesizing meaningful outputs or responses based on the retrieved graph data. This could encompass answering user queries, generating reports, etc. In this stage, a generator takes the query, retrieved graph elements, and an optional prompt as input to generate a response, which can be denoted as

$$\begin{aligned} a^* &= \textbf{G-Generator}(q, G^*) \quad \text{G* comes from previous step} \\ &= \arg\max_{a \in A} p_\phi(a|q, G^*) \\ &= \arg\max_{a \in A} p_\phi(a|\mathcal{F}(q, G^*)), \end{aligned} \tag{6}$$

where $\mathcal{F}(\cdot, \cdot)$ is a function that converts graph data into a form the generator can process.

5 Graph-Based Indexing

The construction and indexing of graph databases form the foundation of GraphRAG, where the quality of the graph database directly impacts GraphRAG's performance. In this section, we categorize and summarize the selection or construction of graph data and various indexing methods that have been employed.

5.1 Graph Data

Various types of graph data are utilized in GraphRAG for retrieval and generation. Here, we categorize these data into two categories based on their sources, including Open Knowledge Graphs and Self-Constructed Graph Data.

5.1.1 Open Knowledge Graphs. Open knowledge graphs refer to graph data sourced from publicly available repositories or databases [2, 7, 138, 151]. Using these knowledge graphs could dramatically reduce the time and resources required to develop and maintain. In this survey, we further classify them into two categories according to their scopes, i.e., General Knowledge Graphs and Domain Knowledge Graphs.

(1) General Knowledge Graphs. General knowledge graphs primarily store general, structured knowledge, and typically rely on collective input and updates from a global community, ensuring a comprehensive and continually refreshed repository of information.

Encyclopedic knowledge graphs are a typical type of general knowledge graph, which contains large-scale real-world knowledge collected from human experts and encyclopedias. For example, Wikidata[1] [151] is a free and open knowledge base that stores structured data of its Wikimedia sister projects like Wikipedia, Wikivoyage, Wiktionary, and others. Freebase[2] [7] is an extensive, collaboratively edited knowledge base that compiles data from various sources, including individual contributions and structured data from databases like Wikipedia. DBpedia[3] [2] represents information about millions of entities, including people, places, and things, by leveraging the infoboxes and categories present in Wikipedia articles. YAGO[4] [138] collects knowledge from Wikipedia, WordNet, and GeoNames.

Commonsense knowledge graphs are another type of general knowledge graph. They include abstract commonsense knowledge, such as semantic associations between concepts and causal relationships between events. Typical Commonsense Knowledge Graphs include: ConceptNet[5] [91] is a semantic network built from nodes representing words or phrases connected by edges denoting semantic relationships. ATOMIC [56, 131] models the causal relationships between events.

(2) Domain Knowledge Graphs. As discussed in Section 1, domain-specific knowledge graphs are crucial for enhancing LLMs in addressing domain-specific questions. These KGs offer specialized knowledge in particular fields, aiding models in gaining deeper insights and a more comprehensive understanding of complex professional relationships. In the biomedical field, CMeKG[6] encompasses a wide range of data, including diseases, symptoms, treatments, medications, and relationships between medical concepts. CPubMed-KG[7] is a medical knowledge database in Chinese, building on the extensive repository of biomedical literature in PubMed. In the movie domain, Wiki-Movies [110]

[1] https://www.wikidata.org/
[2] http://www.freebase.be/
[3] https://www.dbpedia.org/
[4] https://yago-knowledge.org/
[5] https://conceptnet.io/
[6] https://cmekg.pcl.ac.cn/
[7] https://cpubmed.openi.org.cn/graph/wiki

extracts structured information from Wikipedia articles related to films, compiling data about movies, actors, directors, genres, and other relevant details into a structured format. Additionally, Jin et al. [66] construct a dataset named GR-Bench, which includes five domain knowledge graphs spanning academic, e-commerce, literature, healthcare, and legal fields. Furthermore, He et al. [47] convert triplet-format and JSON files from ExplaGraphs and SceneGraphs into a standard graph format and selects questions requiring 2-hop reasoning from WebQSP to create the universal graph-format dataset GraphQA for evaluating GraphRAG systems.

5.1.2 Self-Constructed Graph Data. Self-Constructed Graph Data facilitates the customization and integration of proprietary or domain-specific knowledge into the retrieval process. For downstream tasks that do not inherently involve graph data, researchers often propose constructing a graph from multiple sources (e.g., documents, tables, and other databases) and leveraging GraphRAG to enhance task performance. Generally, these self-constructed graphs are closely tied to the specific design of the method, distinguishing them from the open-domain graph data previously mentioned.

To model the structural relationships between the documents, Munikoti et al. [113] propose to construct a heterogeneous document graph capturing multiple document-level relations, including co-citation, co-topic, co-venue, etc. Li et al. [87] and Wang et al. [160] establish relationship between passages according to shared keywords. To capture the relations between entities in documents, Delile et al. [20], Edge et al. [25], Gutiérrez et al. [43] and Li et al. [80] utilize the named entity recognition tools to extract entities from documents and language models to further extract relations between entities, where the retrieved entities and relations then form a knowledge graph. There are also some mapping methods for downstream tasks that need to be designed based on the characteristics of the task itself. For example, to solve the patent phrase similarity inference task, Peng and Yang [122] convert the patent database into a patent-phrase graph. Connections between patent nodes and phrase nodes are established if the phrases appear in the patents, while connections between patent nodes are based on citation relations. Targeting customer service technical support scenarios, Xu et al. [169] propose to model historical issues into a KG, which transforms the issues into tree representations to maintain the intra-issue relations, and utilize semantic similarities and a threshold to preserve inter-issue relations.

5.2 Indexing

Graph-Based Indexing plays a crucial role in enhancing the efficiency and speed of query operations on graph databases, directly influencing subsequent retrieval methods and granularity. Common graph-based indexing methods include graph indexing, text indexing, and vector indexing.

5.2.1 Graph Indexing. Graph indexing represents the most commonly used approach, preserving the entire structure of the graph. This method ensures that for any given node, all its edges and neighboring nodes are easily accessible. During subsequent retrieval stages, classic graph search algorithms such as BFS and Shortest Path Algorithms can be employed to facilitate retrieval tasks [64, 66, 102, 142, 146, 175].

5.2.2 Text Indexing. Text indexing involves converting graph data into textual descriptions to optimize retrieval processes. These descriptions are stored in a text corpus, where various text-based retrieval techniques, such as sparse retrieval and dense retrieval, can be applied. Some approaches transform knowledge graphs into human-readable text using predefined rules or templates. For instance, Li et al. [81], Huang et al. [55] and Li et al. [86] use predefined templates to convert each triple in knowledge graphs into natural language, while Yu et al. [179] merge triplets with the same head entity into passages. Additionally, some methods convert subgraph-level information into

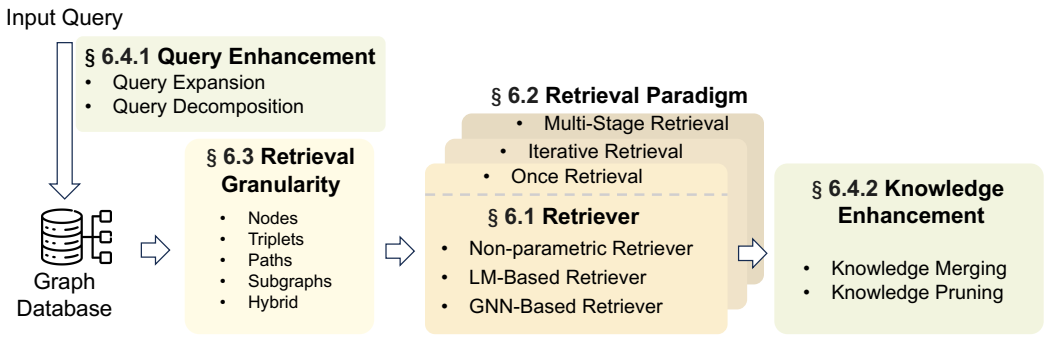

Fig. 3. The general architectures of graph-based retrieval.

textual descriptions. For example, Edge et al. [25] perform community detection on the graph and generate summaries for each community using LLMs.

5.2.3 Vector Indexing. Vector indexing transforms graph data into vector representations to enhance retrieval efficiency, facilitating rapid retrieval and effective query processing. For example, entity linking can be seamlessly applied through query embeddings, and efficient vector search algorithms such as Locality Sensitive Hashing (LSH) [57] can be utilized. G-Retriever [47] employs language models to encode textual information associated with each node and edge within the graph, while GRAG [50] uses language models to convert k-hop ego networks into graph embeddings, thereby better preserving structural information.

Remark. These three indexing methods each offer distinct advantages: graph indexing facilitates easy access to structural information, text indexing simplifies retrieval of textual content, and vector indexing enables quick and efficient searches. Therefore, in practical applications, a hybrid approach combining these indexing methods is often preferred over relying solely on one.

6 Graph-Guided Retrieval

In GraphRAG, the retrieval process is crucial for ensuring the quality and relevance of generated outputs by extracting pertinent and high-quality graph data from external graph databases. However, retrieving graph data presents two significant challenges: (1) *Explosive Candidate Subgraphs*: As the graph size increases, the number of candidate subgraphs grows exponentially, requiring heuristic search algorithms to efficiently explore and retrieve relevant subgraphs. (2) *Insufficient Similarity Measurement*: Accurately measuring similarity between textual queries and graph data necessitates the development of algorithms capable of understanding both textual and structural information.

Considerable efforts have previously been dedicated to optimizing the retrieval process to address the above challenges. This survey focuses on examining various aspects of the retrieval process within GraphRAG, including the selection of the retriever, retrieval paradigm, retrieval granularity, and effective enhancement techniques. The general architectures of Graph-Guided Retrieval are depicted in Figure 3.

6.1 Retriever

In GraphRAG, various retrievers possess unique strengths for addressing different aspects of retrieval tasks. We categorize retrievers into three types based on their underlying models: Non-parametric Retriever, LM-based Retriever, and GNN-based Retriever. It is important to note that

models used in pre-processing steps, such as query encoding and entity linking, are not considered here, as these models vary across different methods and are not the primary focus of this paper.

6.1.1 Non-parametric Retriever. Non-parametric retrievers, based on heuristic rules or traditional graph search algorithms, do not rely on deep-learning models, thereby achieving high retrieval efficiency. For instance, Yasunaga et al. [175] and Taunk et al. [146] retrieve k-hop paths containing the topic entities of each question-choice pair. G-Retriever [47] enhances the conventional Prize-Collecting Steiner Tree (PCST) algorithm by incorporating edge prices and optimizing relevant subgraph extraction. Delile et al. [20] and Mavromatis and Karypis [108] first extract entities mentioned in the query and then retrieve the shortest path related to these entities. These methods often involve an entity linking pre-processing step to identify nodes in the graph before retrieval.

6.1.2 LM-based Retriever. LMs serve as effective retrievers in GraphRAG due to their strong natural language understanding capabilities. These models excel in processing and interpreting diverse natural language queries, making them versatile for a wide range of retrieval tasks within graph-based frameworks. We primarily categorized LMs into two types: discriminative and generative language models. Subgraph Retriever [181] trains RoBERTa [97] as the retriever, which expands from the topic entity and retrieves the relevant paths in a sequential decision process. KG-GPT [71] adopts LLMs to generate the set of top-K relevant relations of the specific entity. Wold et al. [164] utilize fine-tuned GPT-2 to generate reasoning paths. StructGPT [58] utilizes LLMs to automatically invoke several pre-defined functions, by which relevant information can be retrieved and combined to assist further reasoning.

6.1.3 GNN-based Retriever. GNNs are adept at understanding and leveraging complex graph structures. GNN-based retrievers typically encode graph data and subsequently score different retrieval granularities based on their similarity to the query. For example, GNN-RAG [108] first encodes the graph, assigns a score to each entity, and retrieves entities relevant to the query based on a threshold. EtD [90] iterates multiple times to retrieve relevant paths. During each iteration, it first uses LLaMA2 [148] to select edges connecting the current node, then employs GNNs to obtain embeddings of the new layer of nodes for the next round of LLM selection.

Remark. During the retrieval process, non-parametric retrievers exhibit good retrieval efficiency, but they may suffer from inaccurate retrieval due to a lack of training on downstream tasks. Meanwhile, although LM-based retrievers and GNN-based retrievers offer higher retrieval accuracy, they require significant computational overhead. Considering this complementarity, many methods propose hybrid retrieval approaches to improve both retrieval efficiency and accuracy. Many approaches adopt a multi-stage retrieval strategy, employing different models at each stage. For example, RoG [102] first utilizes LLMs to generate planning paths and then extracts paths satisfying the planning paths from knowledge graphs. GenTKGQA [36] infers crucial relations and constraints from the query using LLMs and extracts triplets according to these constraints.

6.2 Retrieval Paradigm

Within GraphRAG, different retrieval paradigms, including once retrieval, iterative retrieval, and multi-stage retrieval, play crucial roles in improving the relevance and depth of the retrieved information. Once retrieval aims to gather all pertinent information in a single operation. Iterative retrieval conducts further searches based on previously retrieved information, progressively narrowing down to the most relevant results. Here we further divide iterative retrieval into adaptive retrieval and non-adaptive retrieval, with the only difference lying in whether the stopping of the retrieval is determined by the model. Another retrieval paradigm is multi-stage retrieval, where

retrieval is divided into multiple stages. Different types of retrievers may be employed at each stage for more precise and diversified search results. Below, we will provide a detailed introduction to these types of retrieval paradigms.

6.2.1 Once Retrieval. Once retrieval aims to retrieve all the relevant information in a single query. One category of approaches [43, 50, 81] utilize embedding similarities to retrieve the most relevant pieces of information. Another category of methods design pre-defined rules or patterns to directly extract specific structured information such as triplets, paths or subgraphs from graph databases. For example, G-Retriever [47] utilizes an extended PCST algorithm to retrieve the most relevant subgraph. KagNet [88] extracts paths between all pairs of topic entities with lengths not exceeding k. Yasunaga et al. [175] and Taunk et al. [146] extract the subgraph that contains all topic entities along with their 2-hop neighbors.

Furthermore, in this subsection, we also include some multiple retrieval methods that involve decoupled and independent retrievals, allowing them to be computed in parallel and executed only once. For example, Luo et al. [102] and Cheng et al. [16] first instruct LLMs to generate multiple reasoning paths and then use a BFS retriever to sequentially search for subgraphs in the knowledge graphs that match each path. KG-GPT [71] decomposes the original query into several sub-queries, retrieving relevant information for each sub-query in a single retrieval process.

6.2.2 Iterative Retrieval. In iterative retrieval, multiple retrieval steps are employed, with subsequent searches depending on the results of prior retrievals. These methods aim to deepen the understanding or completeness of the retrieved information over successive iterations. In this survey, we further classify iterative retrieval into two categories: (1) non-adaptive and (2) adaptive retrieval. We provide a detailed summary of these two categories of methods below.

(1) Non-Adaptive Retrieval. Non-adaptive methods typically follow a fixed sequence of retrieval, and the termination of retrieval is determined by setting a maximum time or a threshold. For example, PullNet [139] retrieves problem-relevant subgraphs through T iterations. In each iteration, the paper designs a retrieval rule to select a subset of retrieved entities, and then expands these entities by searching relevant edges in the knowledge graph. In each iteration, KGP [160] first selects seed nodes based on the similarity between the context and the nodes in the graph. It then uses LLMs to summarize and update the context of the neighboring nodes of the seed nodes, which is utilized in the subsequent iteration.

(2) Adaptive Retrieval. One distinctive characteristic of adaptive retrieval is to let models autonomously determine the optimal moments to finish the retrieval activities. For instance, [42, 168] leverage an LM for hop prediction, which serves as an indicator to end the retrieval. There is also a group of researchers who utilize model-generated special tokens or texts as termination signals for the retrieval process. For example, ToG [142] prompts the LLM agent to explore the multiple possible reasoning paths until the LLM determines the question can be answered based on the current reasoning path. [181] trains a RoBERTa to expand a path from each topic entity. In the process, a virtual relation named as "[END]" is introduced to terminate the retrieval process.

Another common approach involves treating the large model as an agent, enabling it to directly generate answers to questions to signal the end of iteration. For instance, [58, 60, 66, 143, 158] propose LLM-based agents to reason on graphs. These agents could autonomously determine the information for retrieval, invoke the pre-defined retrieval tools, and cease the retrieval process based on the retrieved information.

6.2.3 Multi-Stage Retrieval. Multi-stage retrieval divides the retrieval process linearly into multiple stages, with additional steps such as retrieval enhancement, and even generation processes occurring between these stages. In multi-stage retrieval, different stages may employ various types of retrievers, which enables the system to incorporate various retrieval techniques tailored to different aspects of the query. For example, Wang et al. [159] first utilize a non-parametric retriever to extract n-hop paths of entities in the query's reasoning chain, then after a pruning stage, it further retrieves the one-hop neighbors of the entities in the pruned subgraph. OpenCSR [45] divides the retrieval process into two stages. In the first stage, it retrieves all 1-hop neighbors of the topic entity. In the second stage, it compares the similarity between these neighbor nodes and other nodes, selecting the top-k nodes with the highest similarity for retrieval. GNN-RAG [108] first employs GNNs to retrieve the top-k nodes most likely to be the answer. Subsequently, it retrieves all shortest paths between query entities and answer entities pairwise.

Remark. In GraphRAG, once retrieval typically exhibits lower complexity and shorter response times, making it suitable for scenarios requiring real-time responsiveness. In contrast, iterative retrieval often involves higher time complexity, especially when employing LLMs as retrievers, potentially leading to longer processing times. However, this approach can yield higher retrieval accuracy by iteratively refining retrieved information and generating responses. Therefore, the choice of retrieval paradigm should balance accuracy and time complexity based on specific use cases and requirements.

6.3 Retrieval Granularity

According to different task scenarios and indexing types, researchers design distinct retrieval granularities (i.e., the form of related knowledge retrieved from graph data), which can be divided into nodes, triplets, paths, and subgraphs. Each retrieval granularity has its own advantages, making it suitable for different practical scenarios. We will introduce the details of these granularities in the following sections.

6.3.1 Nodes. Nodes allow for precise retrieval focused on individual elements within the graph, which is ideal for targeted queries and specific information extraction. In general, for knowledge graphs, nodes refer to entities. For other types of text attribute graphs, nodes may include textual information that describes the node's attributes. By retrieving nodes within the graph, GraphRAG systems could provide detailed insights into their attributes, relationships, and contextual information. For example, Munikoti et al. [113], Li et al. [87] and Wang et al. [160] construct document graphs and retrieves relevant passage nodes. Liu et al. [90], Sun et al. [139] and Gutiérrez et al. [43] retrieve entities from constructed knowledge graphs.

6.3.2 Triplets. Generally, triplets consist of entities and their relationships in the form of subject-predicate-object tuples, providing a structured representation of relational data within a graph. The structured format of triplets allows for clear and organized data retrieval, making it advantageous in scenarios where understanding relationships and contextual relevance between entities is critical. Yang et al. [171] retrieve triplets containing topic entities as relevant information. Huang et al. [55], Li et al. [81] and Li et al. [86] first convert each triplet of graph data into textual sentences using predefined templates and subsequently adopt a text retriever to extract relevant triplets. However, directly retrieving triplets from graph data may still lack contextual breadth and depth, thus being unable to capture indirect relationships or reasoning chains. To address this challenge, Wang et al. [152] propose to generate the logical chains based on the original question, and retrieve the relevant triplets of each logical chain.

6.3.3 Paths. The retrieval of path-granularity data can be seen as capturing sequences of relationships between entities, enhancing contextual understanding and reasoning capabilities. In GraphRAG, retrieving paths offers distinct advantages due to their ability to capture complex relationships and contextual dependencies within a graph.

However, path retrieval can be challenging due to the exponential growth in possible paths as graph size increases, which escalates computational complexity. To address this, some methods retrieve relevant paths based on pre-defined rules. For example, Wang et al. [159] and Lo and Lim [98] first select entity pairs in the query and then traverse to find all the paths between them within n-hop. HyKGE [64] first defines three types of paths: path, co-ancestor chain, and co-occurrence chain, and then utilizes corresponding rules to retrieve each of these three types of paths. In addition, some methods utilize models to perform path searching on graphs. ToG [142] proposes to prompt the LLM agent to perform the beam search on KGs and find multiple possible reasoning paths that help answer the question. Luo et al. [102], Wu et al. [168] and Guo et al. [42] first utilizes the model to generate faithful reasoning plans and then retrieves relevant paths based on these plans. GNN-RAG [108] first identifies the entities in the question. Subsequently, all paths between entities that satisfy a certain length relationship are extracted.

6.3.4 Subgraphs. Retrieving subgraphs offers significant advantages due to its ability to capture comprehensive relational contexts within a graph. This granularity enables GraphRAG to extract and analyze complex patterns, sequences, and dependencies embedded within larger structures, facilitating deeper insights and a more nuanced understanding of semantic connections.

To ensure both information completeness and retrieval efficiency, some methods propose an initial rule-based approach to retrieve candidate subgraphs, which are subsequently refined or processed further. Peng and Yang [122] retrieve the ego graph of the patent phrase from the self-constructed patent-phrase graph. Yasunaga et al. [175], Feng et al. [32] and Taunk et al. [146] first select the topic entities and their two-hop neighbors as the node set, and then choose the edges with head and tail entities both in the node set to form the subgraph. Besides, there are also some embedding-based subgraph retrieval methods. For example, Hu et al. [50] first encode all the k-hop ego networks from the graph database, then retrieve subgraphs related to the query based on the similarities between embeddings. Wen et al. [163] and Li et al. [80] extract two types of graphs, including Path evidence subgraphs and Neighbor evidence subgraphs, based on pre-defined rules. OpenCSR [45] starts from a few initial seed nodes and gradually expands to new nodes, eventually forming a subgraph.

In addition to the aforementioned direct subgraph retrieval methods, some works propose first retrieving relevant paths and then constructing related subgraphs from them. For instance, Zhang et al. [181] train a RoBERTa model to identify multiple reasoning paths through a sequential decision process, subsequently merging identical entities from different paths to induce a final subgraph.

6.3.5 Hybrid Granularties. Considering the advantages and disadvantages of various retrieval granularities mentioned above, some researchers propose using hybrid granularities, that is, retrieving relevant information of multiple granularities from graph data. This type of granularity enhances the system's ability to capture both detailed relationships and broader contextual understanding, thus reducing noise while improving the relevance of the retrieved data. Various previous works propose to utilize LLM agents to retrieve complex hybrid information. Jin et al. [66], Jiang et al. [58], Jiang et al. [60], Wang et al. [158] and Sun et al. [143] propose to adopt LLM-based agents for adaptively selecting nodes, triplets, paths, and subgraphs.

Remark. (1) In real applications, there are no clear boundaries between these retrieval granularities, as subgraphs can be composed of multiple paths, and paths can be formed by several triplets. (2) Various granularities such as nodes, triplets, paths, and subgraphs offer distinct advantages in the GraphRAG process. Balancing between retrieval content and efficiency is crucial when selecting the granularity, depending on the specific context of the task. For straightforward queries or when efficiency is paramount, finer granularities such as entities or triplets may be preferred to optimize retrieval speed and relevance. In contrast, complex scenarios often benefit from a hybrid approach that combines multiple granularities. This approach ensures a more comprehensive understanding of the graph structure and relationships, enhancing the depth and accuracy of the generated responses. Thus, GraphRAG's flexibility in granularity selection allows it to adapt effectively to diverse information retrieval needs across various domains.

6.4 Retrieval Enhancement

To ensure high retrieval quality, researchers propose techniques to enhance both user queries and the knowledge retrieved. In this paper, we categorize query enhancement into query expansion and query decomposition, and knowledge enhancement into merging and pruning. These strategies collectively optimize the retrieval process. Although other techniques such as query rewriting [103, 106, 121, 126] are commonly used in RAG, they are less frequently applied in GraphRAG. We do not delve into these methods, despite their potential adaptation for GraphRAG.

6.4.1 Query Enhancement. Strategies applied to queries typically involve pre-processing techniques that enrich the information for better retrieval. This may include query expansion and query decomposition.

(1) Query Expansion. Due to the generally short length of queries and their limited information content, query expansion aims to improve search results by supplementing or refining the original query with additional relevant terms or concepts. Luo et al. [102] generate relation paths grounded by KGs with LLMs to enhance the retrieval query. Cheng et al. [16] adopt SPARQL to get all the aliases of the query entities from Wikidata to augment the retrieval queries, which capture lexical variations of the same entity. Huang et al. [55] propose a consensus-view knowledge retrieval method to improve retrieval accuracy, which first discover semantically relevant queries, and then re-weight the original query terms to enhance the retrieval performance. HyKGE [64] utilizes a large model to generate the hypothesis output of the question, concatenating the hypothesis output with the query as input to the retriever.

(2) Query Decomposition. Query decomposition techniques break down or decompose the original user query into smaller, more specific sub-queries. Each sub-query typically focuses on a particular aspect or component of the original query, which successfully alleviates the complexity and ambiguity of language queries. For instance, [18, 71] breaks down the primary question into sub-sentences, each representing a distinct relation, and sequentially retrieves the pertinent triplets for each sub-sentence.

6.4.2 Knowledge Enhancement. After retrieving initial results, knowledge enhancement strategies are employed to refine and improve the retriever's results. This phase often involves knowledge merging and knowledge pruning processes to present the most pertinent information prominently. These techniques aim to ensure that the final set of retrieved results is not only comprehensive but also highly relevant to the user's information needs.

(1) Knowledge Merging. Knowledge merging retrieved information enables compression and aggregation of information, which assists in obtaining a more comprehensive view by consolidating

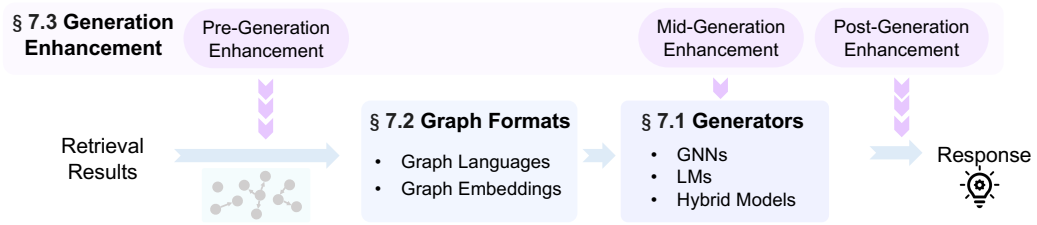

Fig. 4. The overview of graph-enhanced generation.

relevant details from multiple sources. This approach not only enhances the completeness and coherence of the information but also mitigates issues related to input length constraints in models. KnowledgeNavigator [42] merges nodes and condenses the retrieved sub-graph through triple aggregation to enhance the reasoning efficiency. In Subgraph Retrieval [181], after retrieving top-k paths from each topic entity to form a single subgraph, researchers propose to merge the same entities from different subgraphs to form the final subgraph. Wen et al. [163] and Li et al. [80] merge retrieved subgraphs based on relations, combining head entities and tail entities that satisfy the same relation into two distinct entity sets, ultimately forming a relation paths.

(2) Knowledge Pruning. Knowledge pruning involves filtering out less relevant or redundant retrieved information to refine the results. Previous approaches for pruning encompass two main categories: (re)-ranking-based approaches and LLM-based approaches. (Re)-ranking methods involve the reordering or prioritization of retrieved information using tailored metrics or criteria.

One line of methods introduces stronger models for reranking. For example, Li et al. [81] concatenate each retrieved triplet with the question-choice pair, and adopt a pre-trained cross-encoder [129] to re-rank the retrieved triplets. Jiang et al. [64] utilize the FlagEmbedding to encode the text to re-rank top-k documents returned by embedding model "bge_reranker_large".

Another category utilizes the similarity between queries and retrieved information for ranking. For instance, Cheng et al. [16] re-rank the candidate subgraphs based on the similarity for both relation and fine-grained concept between subgraphs and the query. Taunk et al. [146] first cluster the 2-hop neighbors and then delete the cluster with the lowest similarity score with the input query. Yasunaga et al. [175] prune the retrieved subgraph according to the relevance score between the question context and the KG entity nodes calculated by a pre-trained language model. Wang et al. [159], Jiang et al. [61], Gutiérrez et al. [43] and Luo et al. [100] adopt Personalized PageRank algorithm to rank the retrieved candidate information for further filtering. G-G-E [35] first divides the retrieved subgraph into several smaller subgraphs, then compares the similarity between each smaller subgraph and the query. Subgraphs with low similarity are removed, and the remaining smaller subgraphs are merged into a larger subgraph.

Additionally, a third category of methods proposes new metrics for reranking. For example, Munikoti et al. [113] propose a metric that measures both the impact and recency of the retrieved text chunks. KagNet [88] decomposes the retrieved paths into triplets and reranks the paths based on the confidence score measured by the knowledge graph embedding (KGE) techniques. LLM-based methods excel in capturing complex linguistic patterns and semantic nuances, which enhances their ability to rank search results or generate responses more accurately. To avoid introducing noisy information, Wang et al. [159] and Kim et al. [71] propose to prune the irrelevant graph data by calling LLMs to check.

7 Graph-Enhanced Generation

The generation stage is another crucial step in GraphRAG, aimed at integrating the retrieved graph data with the query to enhance response quality. In this stage, suitable generation models must be selected based on the downstream tasks. The retrieved graph data is then transformed into formats compatible with the generators. The generator takes both the query and the transformed graph data as inputs to produce the final response. Beyond these fundamental processes, generative enhancement techniques can further improve the output by intensifying the interaction between the query and the graph data and enriching the content generation itself. The organization of this section and the overview of graph-enhanced generation are depicted in Figure 4.

7.1 Generators

The selection of generators often depends on the type of downstream task at hand. For discriminative tasks (e.g., multi-choice question answering) or generative tasks that can be formulated as discriminative tasks (e.g., KBQA), one can utilize GNNs or discriminative language models to learn representations of the data. These representations can then be mapped to the logits associated with different answer options to provide responses. Alternatively, generative language models can be employed to directly generate answers. For generative tasks, however, the use of GNNs and discriminative language models alone is insufficient. These tasks require the generation of text, which necessitates the deployment of decoders.

7.1.1 GNNs. Due to the powerful representational capabilities of GNNs for graph data, they are particularly effective for discriminative tasks. GNNs can directly encode graph data, capturing complex relationships and node features inherent in the graph structure. This encoding is then processed through a Multi-Layer Perceptron (MLP) to generate predictive outcomes. These approaches primarily utilize classical GNN models (e.g., GCN [74], GAT [150], GraphSAGE [44], and Graph Transformers [135]), either in their original form or modified to better align with downstream tasks. For example, Sun et al. [140] compute PageRank scores for neighboring nodes and aggregates them weighted by these scores, during message-passing. This approach enhances the central node's ability to assimilate information from its most relevant neighboring nodes. Mavromatis and Karypis [107] decode the query into several vectors (instructions), and enhances instruction decoding and execution for effective reasoning by emulating breadth-first search (BFS) with GNNs to improve instruction execution and using adaptive reasoning to update the instructions with KG-aware information.

7.1.2 LMs. LMs possess strong capabilities in text understanding, which also allows them to function as generators. In the context of integrating LMs with graph data, it is necessary to first convert the retrieved graph data into specific graph formats. This conversion process ensures that the structured information is effectively understood and utilized by the LMs. These formats, which will be elaborated on in Section 7.2, are crucial for preserving the relational and hierarchical structure of the graph data, thereby enhancing the model's ability to interpret complex data types. Once the graph data is formatted, it is then combined with a query and fed into an LM.

For encoder-only models, such as BERT [22] and RoBERTa [97], their primary use is in discriminative tasks. Similar to GNNs, these models first encode the input text and then utilize MLPs to map it to the answer space [55, 61, 81]. On the other hand, encoder-decoder and decoder-only models, such as T5 [127], GPT-4 [116], and LLaMA [24], are adept at both discriminative and generative tasks. These models excel in text understanding, generation, and reasoning, allowing them to process textual inputs directly and generate textual responses [25, 64, 66, 102, 108, 142, 152, 159].

7.1.3 Hybrid Models. Considering the strengths of GNNs at representing the structure of graph data, and the robust understanding of text demonstrated by LMs, many studies are exploring the integration of these two technologies to generate coherent responses. This paper categorizes the hybrid generative approaches into two distinct types: cascaded paradigm and parallel paradigm.

(1) Cascaded Paradigm. In the cascaded approaches, the process involves a sequential interaction where the output from one model serves as the input for the next. Specifically, the GNN processes the graph data first, encapsulating its structural and relational information into a form that the LM can understand. Subsequently, this transformed data is fed into the LM, which then generates the final text-based response. These methods leverage the strengths of each model in a step-wise fashion, ensuring detailed attention to both structural and textual data.

In these methods, prompt tuning [79, 82, 95, 96] is a typical approach, where GNNs are commonly employed to encode the retrieved graph data. This encoded graph data is subsequently prepended as a prefix to the input text embeddings of an LM. The GNN is then optimized through downstream tasks to produce enhanced encodings of the graph data [36, 47, 50, 182].

(2) Parallel Paradigm. On the other hand, the parallel approach operates by concurrently utilizing the capabilities of both the GNN and the LLM. In this setup, both models receive the initial inputs simultaneously and work in tandem to process different facets of the same data. The outputs are then merged, often through another model or a set of rules, to produce a unified response that integrates insights from both the graphical structure and the textual content.

In the parallel paradigm, a typical approach involves separately encoding inputs using both GNNs and LMs, followed by integrating these two representations, or directly integrating their output responses. For instance, Jiang et al. [59] aggregate predictions from GNNs and LMs by weighted summation to obtain the final answer. Lin et al. [88] and Pahuja et al. [118] integrate the graph representations derived from GNNs and the text representations generated by LMs using attention mechanisms. Yasunaga et al. [175], Munikoti et al. [113] and Taunk et al. [146] directly concatenate graph representations with text representations.

Another approach involves designing dedicated modules that integrate GNNs with LMs, enabling the resulting representations to encapsulate both structural and textual information. For instance, Zhang et al. [184] introduce a module called the GreaseLM Layer, which incorporates both GNN and LM layers. At each layer, this module integrates textual and graph representations using a two-layer MLP before passing them to the next layer. Similarly, ENGINE [189] proposes G-Ladders, which combine LMs and GNNs through a side structure, enhancing node representations for downstream tasks.

Remark. Hybrid models that harness both the representation capabilities of GNNs for graph data and LMs for text data hold promising applications. However, effectively integrating information from these two modalities remains a significant challenge.

7.2 Graph Formats

When using GNNs as generators, the graph data can be directly encoded. However, when utilizing LMs as generators, the non-Euclidean nature of graph data poses a challenge, as it cannot be directly combined with textual data for input into the LMs. To address this, graph translators are employed to convert the graph data into a format compatible with LMs. This conversion enhances the generative capabilities of LMs by enabling them to effectively process and utilize structured graph information. In this survey, we summarize two distinct graph formats: graph languages and graph embeddings. We illustrate this process with an example in Figure 5, with detailed introductions provided below.

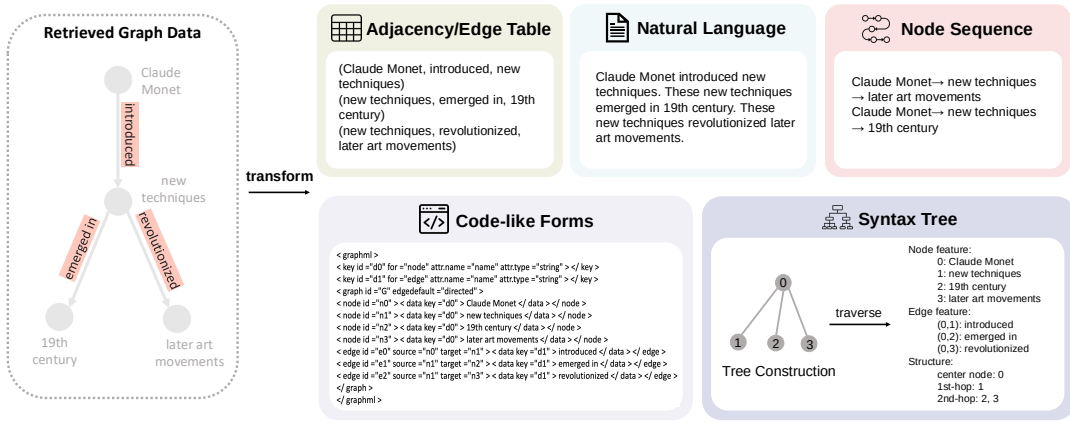

Fig. 5. Illustration of the graph languages. Given the retrieved subgraph on the left part, we show how to transform it into adjacency/edge table, natural language, node sequence, code-like forms and syntax trees to adapt the input form requirements of different generators.

7.2.1 Graph Languages. A graph description language is a formalized system of notation that is specifically crafted to characterize and represent graph data. It prescribes a uniform syntax and semantic framework that describes the components and interconnections within a graph. Through these languages, users can consistently generate, manipulate, and interpret graph data in a comprehensible format to machines. They enable the definition of graph architectures, the specification of attributes for nodes and edges, and the implementation of operations and queries on graph structures. Next, we will introduce five types of graph languages separately: Adjacency / Edge Table, Natural Language, Codes, Syntax Tree, and Node Sequence.

(1) Adjacency / Edge Table. The adjacency table and the edge table are widely used methods for describing graph structures [30, 41, 85, 153]. The adjacency table enumerates the immediate neighbors of each vertex, offering a compact way to represent connections in sparse graphs. For example, KG-GPT [71] linearizes the triples in the retrieved subgraph, which are then concatenated and fed into the LLMs. Conversely, the edge table details all the edges within the graph, providing a straightforward representation that is particularly useful for processing and analyzing graphs in a linear format. Both two methods are brief, easy to understand, and intuitive.

(2) Natural Language. Given that user queries are typically presented in natural language, and considering the outstanding natural language comprehension capabilities of LMs, it becomes a compelling approach to describe the retrieved graph data using natural language. By translating graph data into descriptive, easily comprehensible language, LMs can bridge the gap between raw data representation and user-friendly information, facilitating more effective interactions with data-driven applications. For example, some researchers [55, 81] propose defining a natural language template for each type of edge in advance and subsequently filling in the endpoints of each edge into the corresponding template based on its type. Ye et al. [176] employ natural language to describe the information of 1-hop and 2-hop neighboring nodes of the central node. Edge et al. [25] utilize LLMs to generate report-like summaries for each detected graph community. Wu et al. [168] and Guo et al. [42] adopt LMs to rewrite the edge table of retrieved subgraphs, generating a natural language description. Fatemi et al. [30] explore different representations of nodes (e.g., Integer encoding, alphabet letters, names, etc.) and edges (e.g., parenthesis, arrows, incident, etc.). Jin et al.

[66], Jiang et al. [58], Jiang et al. [60], Wang et al. [158] and Sun et al. [143] integrate information from different granularities within the graph into prompts through natural language in the form of dialogue.

(3) Code-Like Forms. Considering that natural language descriptions and other 1-D sequences are inherently inadequate for directly representing the 2-D structure of graph data, and given the robust code comprehension capabilities of LMs, many researchers [41] explore using code-like formats to represent graph structures. For example, Guo et al. [41] examine the use of Graph Modeling Language (GML) [48] and Graph Markup Language (GraphML) [130] for representing graphs. These standardized languages are specifically designed for graph data, providing comprehensive descriptions that encompass nodes, edges, and their interrelationships.

(4) Syntax Tree. Compared to direct flattening of graphs, some research [186] propose transforming graphs into structures akin to syntax trees. Syntax trees possess a hierarchical structure and, being topological graphs, also maintain a topological order. This method retains more structural information, enhancing the understanding and analysis of the graph's intrinsic properties. Such a transformation not only preserves the relational dynamics between different graph elements but also facilitates more sophisticated algorithms for graph analysis and processing. GRAPHTEXT [186] proposes transforming the ego network of a central node into a graph-syntax tree format. This format not only encapsulates structural information but also integrates the features of the nodes. By traversing this syntax tree, it is possible to obtain a node sequence that maintains both topological order and hierarchical structure.

(5) Node Sequence. Some studies [14, 108] propose representing graphs through sequences of nodes, which are often generated using predefined rules. Compared to natural language descriptions, these node sequences are more concise and incorporate prior knowledge, specifically the structural information emphasized by the rules. Luo et al. [102] and Sun et al. [142] transform the retrieved paths into node sequences and input them into an LLM to enhance the task performance. LLaGA [14] proposes two templates that can transform graphs into node sequences. The first template, known as the Neighborhood Detail Template, offers a detailed examination of the central node along with its immediate surroundings. The second, termed the Hop-Field Overview Template, provides a summarized perspective of a node's neighborhood, which can be expanded to encompass broader areas. GNN-RAG [108] inputs the retrieved reasoning paths into LMs in the form of node sequences as prompts.

Remark. Good graph languages should be complete, concise, and comprehensible. Completeness entails capturing all essential information within the graph structure, ensuring no critical details are omitted. Conciseness refers to the necessity of keeping textual descriptions brief to avoid the "lost in the middle" phenomenon [94] or exceeding the length limitations of LMs. Lengthy inputs can hinder LMs' processing capabilities, potentially causing loss of context or truncated data interpretation. Comprehensibility ensures that the language used is easily understood by LLMs, facilitating accurate representation of the graph's structure. Due to the characteristics of different graph languages, their choice can significantly impact the performance of downstream tasks [30].

7.2.2 Graph Embeddings. The above graph language methods transform graph data into text sequences, which may result in overly lengthy contexts, incurring high computational costs and potentially exceeding the processing limits of LLMs. Additionally, LLMs currently struggle to fully comprehend graph structures even with graph languages [41]. Thus, using GNNs to represent graphs as embeddings presents a promising alternative. The core challenge lies in integrating graph embeddings with textual representations into a unified semantic space. Current research focuses on

utilizing prompt tuning methodologies, as discussed earlier. Notably, feeding graph representations into LMs is feasible primarily with open-source LMs, not closed-source models like GPT-4 [116]. While graph embedding methods avoid handling long text inputs, they face other challenges, such as difficulty in preserving precise information like specific entity names and poor generalization.

7.3 Generation Enhancement

In the generation phase, besides converting the retrieved graph data into formats acceptable by the generator and inputting it together with the query to generate the final response, many researchers explore various methods of generation enhancement techniques to improve the quality of output responses. These methods can be classified into three categories based on their application stages: pre-generation enhancement, mid-generation enhancement, and post-generation enhancement.

7.3.1 Pre-Generation Enhancement. Pre-generation enhancement techniques focus on improving the quality of input data or representations before feeding them into the generator. In fact, there is no clear boundary between Pre-Generation Enhancement and Retrieval. In this survey, we categorize the retrieval stage as the process of retrieving knowledge from the original graph, and merging and pruning retrieved knowledge. Subsequent operations are considered Pre-Generation Enhancements.

Commonly used pre-generation enhancement approaches primarily involve semantically enriching the retrieved graph data to achieve tighter integration between the graph data and textual query. Wu et al. [168] employ LLMs to rewrite retrieved graph data, enhancing the naturalness and semantic richness of the transformed natural language output. This method not only ensures that graph data is converted into more fluent and natural language but also enriches its semantic content. Conversely, DALK [80] utilizes the retrieved graph data to rewrite the query. Cheng et al. [16] first leverage LLMs to generate a reasoning plan and answer queries according to the plan. Taunk et al. [146] and Yasunaga et al. [175] aim to enhance GNNs by enabling them to learn graph representations relevant to queries. They achieve this by extracting all nouns from the QA pairs (or the QA pairs themselves) and inserting them as nodes into the retrieved subgraph. Mavromatis and Karypis [107] propose a method where, prior to generation, the representation of the query is decomposed into multiple vectors termed "instructions", each representing different features of the query. These instructions are used as conditions during message passing when applying GNNs to learn from retrieved subgraphs. In addition, there are methods that incorporate additional information beyond graph data. For example, PullNet [139] incorporates documents relevant to entities and MVP-Tuning [55] retrieves other related questions.

7.3.2 Mid-Generation Enhancement. Mid-generation enhancement involves techniques applied during the generation process. These methods typically adjust the generation strategies based on intermediate results or contextual cues. TIARA [136] introduces constrained decoding to control the output space and reduce generation errors. When generating logical forms, if the constrained decoder detects that it is currently generating a pattern item, it restricts the next generated token to options that exist in tries containing KB classes and relations. Compared with the Beam Search, this approach ensures that pattern items generated are guaranteed to exist in the knowledge graph, thereby reducing generation errors. There are other methods adjusting the prompts of LLMs to achieve multi-step reasoning. For example, MindMap [163] not only produces answers but also generates the reasoning process.

7.3.3 Post-Generation Enhancement. Post-generation enhancement occurs after the initial response is generated. Post-generation enhancement methods primarily involve integrating multiple generated responses to obtain the final response. Some methods focus on integrating outputs from

the same generator under different conditions or inputs. For example, Edge et al. [25] generate a summary for each graph community, followed by generating responses to queries based on the summary, and then scoring these responses using an LLM. Ultimately, the responses are sorted in descending order according to their scores and sequentially incorporated into the prompt until the token limit is reached. Subsequently, the LLM generates the final response. Wang et al. [152] and Kim et al. [71] first decompose the query into several sub-questions, then generate answers for each sub-question, and finally merge the answers of all sub-questions to obtain the final answer.

Alternatively, other methods combine or select responses generated by different models. Lin et al. [88] and Jiang et al. [59] combine the outputs generated by both GNNs and LLMs to reach a synergistic effect. UniOQA [86] explores two methods for generating answers: one involves generating queries in Cypher Query Language (CQL) to execute and obtain results, while the other method directly generates answers based on retrieved triplets. The final answer is determined through a dynamic selection mechanism. In EmbedKGQA [133], besides the learned scoring function, researchers additionally design a rule-based score based on the graph structures. These two scores are combined to find the answer entity. Li et al. [85] combine answers based on retrieved graph data with responses generated according to the LLM's own knowledge.

8 Training

In this section, we summarize the individual training of retrievers, generators, and their joint training. We categorize previous works into Training-Free and Training-Based approaches based on whether explicit training is required. Training-Free methods are commonly employed when using closed-source LLMs such as GPT-4 [116] as retrievers or generators. These methods primarily rely on carefully crafted prompts to control the retrieval and generation capabilities of LLMs. Despite LLMs' strong abilities in text comprehension and reasoning, a challenge of Training-Free methods lies in the potential sub-optimality of results due to the lack of specific optimization for downstream tasks. Conversely, Training-Based methods involve training or fine-tuning models using supervised signals. These approaches enhance the model performance by adapting them to specific task objectives, thereby potentially improving the quality and relevance of retrieved or generated content. Joint training of retrievers and generators aims to enhance their synergy, thereby boosting performance on downstream tasks. This collaborative approach leverages the complementary strengths of both components to achieve more robust and effective results in information retrieval and content generation applications.

8.1 Training Strategies of Retriever

8.1.1 Training-Free. There are two primary types of Training-Free Retrievers currently in use. The first type consists of non-parametric retrievers. These retrievers rely on pre-defined rules or traditional graph search algorithms rather than specific models [146, 175]. The second type utilizes pre-trained LMs as retrievers. Specifically, one group of works utilizes pre-trained embedding models to encode the queries and perform retrieval directly based on the similarity between the query and graph elements [81]. Another group of works adopts generative language models for training-free retrieval. Candidate graph elements such as entities, triples, paths, or subgraphs are included as part of the prompt input to the LLMs. The LLMs then leverage semantic associations to select appropriate graph elements based on the provided prompt [25, 66, 71, 108, 142, 152, 159]. These methods harness the powerful semantic understanding capabilities of LMs to retrieve relevant graph elements without the need for explicit training.

8.1.2 Training-Based. Training retrievers often adopt an autoregressive approach, where the previous relationship path is concatenated to the end of the query. The model then predicts the next relation based on this concatenated input [42, 168].

However, the lack of ground truth for retrieval content in the majority of datasets poses a significant challenge. To address this issue, many methods attempt to construct reasoning paths based on distant supervision to guide retriever training. For example, Zhang et al. [181], Feng et al. [31] and Luo et al. [102] extract all paths (or shortest paths) between entities in the queries and entities in the answers, using them as training data for the retriever. In addition, Zhang et al. [181] also employ a relationship extraction dataset for distant supervision in unsupervised settings. There is another category of methods that utilize implicit intermediate supervision signals to train Retrievers. For instance, KnowGPT [183] starts searching for the optimal path from the head entity, using the discovery of the tail entity as a reward, and is trained using Policy Gradient. NSM [46] employs a bidirectional search strategy, where two retrievers start searching from the head entity and tail entity, respectively. The supervised objective is to ensure that the paths searched by the two retrievers converge as closely as possible.

Some methods argue that distant supervision signals or implicit intermediate supervision signals may contain considerable noise, making it challenging to train effective retrievers. Therefore, they consider employing self-supervised methods for pre-training retrievers. SKP [23] pretrains the DPR (Dense Passage Retrieval) model [69]. Initially, it conducts random sampling on subgraphs and transforms the sampled subgraphs into passages. Subsequently, it randomly masks passages, trains the model using a Masked Language Model (MLM), and employs contrastive learning by treating the masked passages and original passages as positive pairs for comparison.

8.2 Training of Generator

8.2.1 Training-Free. Training-Free Generators primarily cater to closed-source LLMs or scenarios where avoiding high training costs is essential. In these methods, the retrieved graph data is fed into the LLM alongside the query. The LLMs then generate responses based on the task description provided in the prompt, heavily relying on their inherent ability to understand both the query and the graph data.

8.2.2 Training-Based. Training the generator can directly receive supervised signals from downstream tasks. For generative LLMs, fine-tuning can be achieved using supervised fine-tuning (SFT), where task descriptions, queries, and graph data are inputted, and the output is compared against the ground truth for the downstream task [47, 50, 102]. On the other hand, for GNNs or discriminative models functioning as generators, specialized loss functions tailored to the downstream tasks are employed to train the models effectively [59, 81, 146, 175, 184].

8.3 Joint Training

Jointly training retrievers and generators simultaneously enhances performance on downstream tasks by leveraging their complementary strengths. Some approaches unify retrievers and generators into a single model, typically LLMs, and train them with both retrieval and generation objectives simultaneously [102]. This method capitalizes on the cohesive capabilities of a unified architecture, enabling the model to seamlessly retrieve relevant information and generate coherent responses within a single framework.

Other methodologies involve initially training retrievers and generators separately, followed by joint training techniques to fine-tune both components. For instance, Subgraph Retriever [181] adopts an alternating training paradigm, where the retriever's parameters are fixed to use the graph data for training the generator. Subsequently, the generator's parameters are fixed, and

feedback from the generator is used to guide the retriever's training. This iterative process helps both components refine their performance in a coordinated manner.

9 Applications and Evaluation

In this section, we will summarize the downstream tasks, application domains, benchmarks and metrics, and industrial applications related to GraphRAG. Table 1 collects existing GraphRAG techniques, categorizing them by downstream tasks, benchmarks, methods, and evaluation metrics. This table serves as a comprehensive overview, highlighting the various aspects and applications of GraphRAG technologies across different domains.

Table 1. The tasks, benchmarks, methods, and metrics of GraphRAG.

Tasks		Benchmarks	Methods	Metrics
QA	KBQA	WebQSP [178]	[102], [142], [181], [168], [42], [155], [58], [60], [101], [152], [3], [136], [90], [108], [139], [179], [23], [35], [100], [4], [133], [46], [61], [17], [140], [62]	Accuracy, Hits@1, EM, Recall, F1, BERTScore, GPT-4 Average Ranking
		WebQ [5]	[159], [142], [52], [168], [107], [62]	
		CWQ [144]	[102], [142], [52], [181], [155], [60], [101], [107], [78], [90], [108], [139], [179], [35], [100], [61], [85], [46], [17]	
		GrailQA [39]	[142], [60], [136]	
		QALD10-en [123]	[142], [85], [143]	
		SimpleQuestions [9]	[142], [3]	
		CMCQA[8]	[159]	
		MetaQA [185]	[102], [168], [42], [71], [152], [107], [139], [133], [58], [90], [46], [61], [17]	
		Natural Question [75]	[52]	
		TriviaQA [68]	[52], [61]	
		HotpotQA [173]	[52], [43]	
		FACTKG [73]	[71]	
		Mintaka [134]	[3], [85], [4]	
		FreebaseQA [63]	[179], [100]	
	CSQA	CSQA [145]	[146], [175], [55], [81], [88], [31]	
		OBQA [109]	[146], [175], [55], [81], [31], [45]	
		MedQA [67]	[146], [31], [80]	
		SocialIQA [132]	[55]	
		PIQA [6]	[55]	
		RiddleSenseQA [89]	[55]	
IE	Entity Linking	ZESHEL [99]	[167]	Recall@K
		CoNLL [49]	[167]	
	Relation Extraction	T-Rex [26]	[143], [142]	Hits@1
		ZsRE [124]	[85], [143], [142]	
Others	Fact Verification	Creak [115]	[85], [143], [142]	Hits@1
	Link Prediction	FB15K-237 [147]	[18], [118]	MRR, Hits@K
		FB15k [8]	[18]	
		WN18RR [21]	[118]	
		NELL995 [11]	[18]	
	Dialogue Systems	OpenDialKG [111]	[3]	MRR, Hits@K
	Recommender Systems	Yelp[9]	[156]	NDCG@K, Recall@K

9.1 Downstream Tasks

GraphRAG is applied in various downstream tasks (especially NLP tasks), including Question Answering, Information Extraction, and others.

9.1.1 Question Answering. The QA tasks specifically include Knowledge Base Question Answering (KBQA) and CommonSense Question Answering (CSQA).

(1) KBQA. KBQA serves as a cornerstone downstream task for GraphRAG. In KBQA, questions typically pertain to specific knowledge graphs, and answers often involve entities, relationships, or operations between sets of entities within the knowledge graph. The task tests the systems' ability to retrieve and reason over structured knowledge bases, which is crucial in facilitating complex query responses.

(2) CSQA. Distinguished from KBQA, CSQA primarily takes the form of multiple-choice questions. Commonsense reasoning typically presents a commonsense question along with several answer options, each potentially representing either the name of an entity or a statement. The objective is for machines to utilize external commonsense knowledge graphs, such as ConceptNet, to find relevant knowledge pertaining to the question and options, and to engage in appropriate reasoning and derive the correct answer.

9.1.2 Information Retrieval. Information Retrieval tasks consist of two categories: Entity Linking (EL) and Relation Extraction (RE).

(1) Entity Linking. Entity Linking (EL) is a critical task in the field of natural language processing that involves identifying entities mentioned in text segments and linking them to their corresponding entities in a knowledge graph. By leveraging a system such as Graph RAG, it is possible to retrieve relevant information from the knowledge graph, which facilitates the accurate inference of the specific entities that match the mentions in the text [167].

(2) Relation Extraction. Relation Extraction (RE) aims at identifying and classifying semantic relationships between entities within a text. GraphRAG can significantly enhance this task by using graph-based structures to encode and exploit the interdependencies among entities, thus facilitating more accurate and contextually nuanced extraction of relational data from diverse text sources [85, 142, 143].

9.1.3 Others. In addition to the aforementioned downstream tasks, GraphRAG can be applied to various other tasks in the realm of natural language processing such as fact verification, link prediction, dialogue system, and recommender systems.

(1) Fact Verification. The fact verification task typically involves assessing the truthfulness of a factual statement using knowledge graphs. Models are tasked with determining the validity of a given factual assertion by leveraging structured knowledge repositories. GraphRAG techniques can be utilized to extract evidential connections between entities to enhance the system's efficiency and accuracy [85, 125, 142, 143].

(2) Link Prediction. Link prediction involves predicting missing relationships or potential connections between entities in a graph. GraphRAG is applied to this task [18, 118] by leveraging its ability to retrieve and analyze structured information from graphs, enhancing prediction accuracy by uncovering latent relationships and patterns within the graph data.

(3) Dialogue Systems. Dialogue Systems is designed to converse with humans using natural language, handling various tasks such as answering questions, providing information, or facilitating user interactions. By structuring conversation histories and contextual relationships in a graph-based framework, GraphRAG systems [3] can improve the model's ability to generate coherent and contextually relevant responses.

(4) Recommender Systems. In the context of e-commerce platforms, the purchase relationships between users and products naturally form a network graph. The primary objective of recommender systems within these platforms is to predict the future purchasing intentions of users, effectively forecasting the potential connections within this graph [156].

9.2 Application Domains

GraphRAG is widely applied in e-commerce and biomedical, academic, literature, legal, and other application scenarios for its outstanding ability to integrate structured knowledge graphs with natural language processing, which will be introduced below.

9.2.1 E-Commerce. The primary goal in the e-commerce area involves improving customer shopping experiences and increasing sales through personalized recommendations and intelligent customer services. In this area, historical interactions between users and products can naturally form a graph, which implicitly encapsulates users' behavioral patterns and preference information. However, due to the increasing number of e-commerce platforms and the growing volume of user interaction data, using GraphRAG technology to extract key subgraphs is crucial. Wang et al. [156] ensemble multiple retrievers under different types or with different parameters to extract relevant subgraphs, which are then encoded for temporal user action prediction. To improve the model performance of customer service question answering systems, Xu et al. [169] construct a past-issue graph with intra-issue and inter-issue relations. For each given query, subgraphs of similar past issues are retrieved to enhance the system's response quality.

9.2.2 Biomedical. Recently, GraphRAG techniques are increasingly applied in biomedical question answering systems, achieving advanced medical decision-making performance. In this area, each disease is associated with specific symptoms, and every medication contains certain active ingredients that target and treat particular diseases. Some researchers [20, 80] construct KGs for specific task scenarios, while others [64, 163, 171] utilize open-source knowledge graphs such as CMeKG and CPubMed-KG as retrieval sources. Existing methods generally begin with non-parametric retrievers for initial search, followed by designing methods to filter retrieved content through reranking [20, 64, 80, 163, 171]. Additionally, some approaches propose rewriting model inputs using retrieved information to enhance generation effectiveness [80].

9.2.3 Academic. In the academic research domain, each paper is authored by one or more researchers and is associated with a field of study. Authors are affiliated with institutions, and there exist relationships among authors, such as collaboration or shared institutional affiliations. These elements can be structured into a graph format. Utilizing GraphRAG on this graph can facilitate academic exploration, including predicting potential collaborators for an author, identifying trends within a specific field, etc.

9.2.4 Literature. Similar to academic research, a knowledge graph can be constructed in the realm of literature, with nodes representing books, authors, publishers, and series, and edges labeled "written-by", "published-in", and "book-series". GraphRAG can be utilized to enhance realistic applications like smart libraries.

9.2.5 Legal. In legal contexts, extensive citation connections exist between cases and judicial opinions, as judges frequently reference previous opinions when making new decisions. This naturally creates a structured graph where nodes represent opinions, opinion clusters, dockets, and courts, and edges encompass relationships such as "opinion-citation", "opinion-cluster", "cluster-docket", and "docket-court". The application of GraphRAG in legal scenario could aid lawyers and legal researchers in various tasks such as case analysis and legal consultation.

9.2.6 Others. In addition to the above applications, GraphRAG is also applied to other real-world scenarios such as intelligence report generation [128] and patent phrase similarity detection [122]. Ranade and Joshi [128] first construct an Event Plot Graph (EPG) and retrieve the critical aspects of the events to aid the generation of intelligence reports. Peng and Yang [122] create a patent-phrase graph and retrieve the ego-network of the given patent phrase to assist the judgment of phrase similarity.

9.3 Benchmarks and Metrics

9.3.1 Benchmarks. The benchmarks used to evaluate the performance of the GraphRAG system can be divided into two categories. The first category is the corresponding datasets of downstream tasks. We summarize the benchmarks and papers tested with them according to the classification in Section 9.1, details of which are shown in Table 1. The second category consists of benchmarks specifically designed for the GraphRAG systems. These benchmarks usually cover multiple task domains to provide a comprehensive test result. For example, STARK [166] benchmarks LLM Retrieval on semi-structured knowledge bases covering three domains, including product search, academic paper search, and queries in precision medicine to access the capacity of current GraphRAG systems. He et al. [47] propose a flexible question-answering benchmark targeting real-world textual graphs, named GraphQA, which is applicable to multiple applications including scene graph understanding, commonsense reasoning, and knowledge graph reasoning. Graph Reasoning Benchmark (GRBENCH) [66] is constructed to facilitate the research of augmenting LLMs with graphs, which contains 1,740 questions that can be answered with the knowledge from 10 domain graphs. CRAG [172] provides a structured query dataset, with additional mock APIs to access information from underlying mock KGs to achieve fair comparison.

9.3.2 Metrics. The evaluation metrics for GraphRAG can be broadly categorized into two main types: downstream task evaluation (generation quality) and retrieval quality.

(1) Downstream Task Evaluation (Generation Quality). In the majority of research studies, downstream task evaluation metrics serve as the primary method for assessing GraphRAG's performance. For example, in KBQA, Exact Match (EM) and F1 score are commonly used to measure the accuracy of answering entities. In addition, many researchers utilize BERT4Score and GPT4Score to mitigate instances where LLMs generate entities that are synonymous with the ground truth but not exact matches. In CSQA, Accuracy is the most commonly used evaluation metric. For generative tasks such as QA systems, metrics like BLEU, ROUGE-L, METEOR, and others are commonly employed to assess the quality of the text generated by the model.

(2) Retrieval Quality Evaluation. While evaluating GraphRAG based on downstream task performance is feasible, directly measuring the accuracy of retrieved content poses challenges. Therefore, many studies employ specific metrics to gauge the precision of retrieved content. For instance, when ground truth entities are available, retrieval systems face a balance between the quantity of retrieved information and the coverage of answers. Hence, some studies utilize the ratio between answer coverage and the size of the retrieval subgraph to evaluate the performance of the retrieval system. In addition, several studies have explored metrics such as query relevance, diversity, and faithfulness score to respectively assess the similarity between retrieved content and queries, the diversity of retrieved content, and the faithfulness of the information retrieved.

9.4 GraphRAG in Industry

In this section, we mainly focus on industrial GraphRAG systems. These systems are characterized by their reliance on industrial graph database systems or their focus on large-scale graph data, details of which are as follows.

• GraphRAG (by Microsoft)[10]: The system uses LLMs to construct entity-based knowledge graphs and pre-generate community summaries of related entity groups, which enables the capture of both local and global relationships within a document collection, thereby enhancing Query-Focused

[10]https://github.com/microsoft/graphrag

Summarization (QFS) task [25]. The project can also utilize open-source RAG toolkits for rapid implementation, such as LlamaIndex[11], LangChain[12], etc.

• GraphRAG (by NebulaGraph)[13]: The project is the first industrial GraphRAG system, which is developed by NebulaGraph Corporation. The project integrates LLMs into the NebulaGraph database, which aims to deliver more intelligent and precise search results.

• GraphRAG (by Antgroup)[14]: The framework is developed on the foundation of several AI engineering frameworks such as DB-GPT, knowledge graph engine OpenSPG, and graph database TuGraph. Specifically, the system begins by extracting triples from documents using LLMs, which are then stored in the graph database. During the retrieval phase, it identifies keywords from the query, locates corresponding nodes in the graph database, and traverses the subgraph using BFS or DFS. In the generation phase, the retrieved subgraph data is formatted into text and submitted along with the context and query for processing by LLMs.

• NallM (by Neo4j)[15]: The NaLLM (Neo4j and Large Language Models) framework integrates Neo4j graph database technology with LLMs. It aims to explore and demonstrate the synergy between Neo4j and LLMs, focusing on three primary use cases: Natural Language Interface to a Knowledge Graph, Creating a Knowledge Graph from Unstructured Data, and Generate Reports Using Both Static Data and LLM Data.

• LLM Graph Builder (by Neo4j)[16]: It is a project developed by Neo4j for automatically constructing knowledge graphs, suitable for the GraphRAG's Graph Database Construction and Indexing phase. The project primarily utilizes LLMs to extract nodes, relationships, and their properties from unstructured data, and utilizes the LangChain framework to create structured knowledge graphs.

10 Future Prospects

While GraphRAG technology has made substantial strides, it continues to face enduring challenges that demand comprehensive exploration. This section will delve into the prevalent obstacles and outline prospective avenues for future research in the field of GraphRAG.

10.1 Dynamic and Adaptive Graphs

Most GraphRAG methods [25, 33, 76, 77, 101, 174] are built upon static databases; however, as time progresses, new entities and relationships inevitably emerge. Rapidly updating these changes is both promising and challenging. Incorporating updated information is crucial for achieving better results and addressing emerging trends that require current data. Developing efficient methods for dynamic updates and real-time integration of new data will significantly enhance the effectiveness and relevance of GraphRAG systems.

10.2 Multi-Modality Information Integration

Most knowledge graphs primarily encompass textual information, thereby lacking the inclusion of other modalities such as images, audio, and videos, which hold the potential to significantly enhance the overall quality and richness of the database [162]. The incorporation of these diverse modalities could provide a more comprehensive and nuanced understanding of the stored knowledge. However, the integration of such multi-modal data presents considerable challenges. As the volume of information increases, the graph's complexity and size grow exponentially, rendering it increasingly

[11]https://docs.llamaindex.ai/en/stable/ examples/index structs/knowledge graph/KnowledgeGraphDemo.html
[12]https://python.langchain.com/docs/use_cases/graph
[13]https://www.nebula-graph.io/posts/graph-RAG
[14]https://github.com/eosphoros-ai/DB-GPT
[15]https://github.com/neo4j/NaLLM
[16]https://github.com/neo4j-labs/llm-graph-builder

difficult to manage and maintain. This escalation in scale necessitates the development of advanced methodologies and sophisticated tools to efficiently handle and seamlessly integrate the diverse data types into the existing graph structure, ensuring both the accuracy and accessibility of the enriched knowledge graph.

10.3 Scalable and Efficient Retrieval Mechanisms

Knowledge graphs in the industrial setting may encompass millions or even billions of entities, representing a vast and intricate scale. However, most contemporary methods are tailored for small-scale knowledge graphs [25], which may only comprise thousands of entities. Efficiently and effectively retrieving pertinent entities within large-scale knowledge graphs remains a practical and significant challenge. Developing advanced retrieval algorithms and scalable infrastructure is essential to address this issue, ensuring that the system can manage the extensive data volume while maintaining high performance and accuracy in entity retrieval.

10.4 Combination with Graph Foundation Model

Recently, graph foundation models [34, 104], which can effectively address a wide range of graph tasks, have achieved significant success. Deploying these models to enhance the current GraphRAG pipeline is an essential problem. The input data for graph foundation models is inherently graph-structured, enabling them to handle such data more efficiently than LLM models. Integrating these advanced models into the GraphRAG framework could greatly improve the system's ability to process and utilize graph-structured information, thereby enhancing overall performance and capability.

10.5 Lossless Compression of Retrieved Context

In GraphRAG, the retrieved information is organized into a graph structure containing entities and their interrelations. This information is then transformed into a sequence that can be understood by LLMs, resulting in a very long context. There are two issues with inputting such long contexts: LLMs cannot handle very long sequences, and extensive computation during inference can be a hindrance for individuals. To address these problems, lossless compression of long contexts is crucial. This approach removes redundant information and compresses lengthy sentences into shorter, yet meaningful ones. It helps LLMs capture the essential parts of the context and accelerates inference. However, designing a lossless compression technique is challenging. Current works [33, 77] make a trade-off between compression and preserving information. Developing an effective lossless compression technique is crucial but challenging for GraphRAG.

10.6 Standard Benchmarks

GraphRAG is a relatively new field that lacks unified and standard benchmarks for evaluating different methods. Establishing a standard benchmark is crucial for this area as it can provide a consistent framework for comparison, facilitate objective assessments of various approaches, and drive progress by identifying strengths and weaknesses. This benchmark should encompass diverse and representative datasets, well-defined evaluation metrics, and comprehensive test scenarios to ensure robust and meaningful evaluations of GraphRAG methods.

10.7 Broader Applications

Current GraphRAG applications primarily focus on common tasks such as customer service systems [169], recommendation systems [19], and KBQA [33]. Extending GraphRAG to broader applications such as healthcare [70], financial services [1], legal and compliance [72], smart cities and IoT [137], and more, involves incorporating more complex techniques. For instance, in healthcare,

GraphRAG can support medical diagnosis, patient record analysis, and personalized treatment plans by integrating medical literature, patient histories, and real-time health data. In financial services, GraphRAG can be utilized for fraud detection, risk assessment, and personalized financial advice by analyzing transactional data, market trends, and customer profiles. Legal and compliance applications can benefit from GraphRAG by enabling comprehensive legal research, contract analysis, and regulatory compliance monitoring through the integration of legal documents, case law, and regulatory updates. Expanding GraphRAG to these diverse and complex domains will enhance its utility and impact, providing more sophisticated and targeted solutions across various fields.

11 Conclusion

In summary, this survey offers a comprehensive retrospective of GraphRAG technology, systematically categorizing and organizing its fundamental techniques, training methodologies, and application scenarios. GraphRAG significantly enhances the relevance, accuracy, and comprehensiveness of information retrieval by leveraging pivotal relational knowledge derived from graph datasets, thereby addressing critical limitations associated with traditional Retrieval-Augmented Generation approaches. Furthermore, as GraphRAG represents a relatively nascent field of study, we delineate the benchmarks, analyze prevailing challenges, and illuminate prospective future research directions within this domain.

Acknowledgments

This work is supported by Ant Group through Ant Research Intern Program.

References

[1] Muhammad Arslan and Christophe Cruz. 2024. Business-RAG: Information Extraction for Business Insights. *ICSBT 2024* (2024), 88.

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007 (Lecture Notes in Computer Science, Vol. 4825)*. 722–735.

[3] Jinheon Baek, Alham Fikri Aji, Jens Lehmann, and Sung Ju Hwang. 2023. Direct Fact Retrieval from Knowledge Graphs without Entity Linking. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. 10038–10055.

[4] Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering. arXiv:2306.04136 [cs.CL] https://arxiv.org/abs/2306.04136

[5] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1533–1544.

[6] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: Reasoning about Physical Commonsense in Natural Language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. 7432–7439.

[7] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[8] Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*. 1247–1250.

[9] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. arXiv:1506.02075 [cs.LG] https://arxiv.org/abs/1506.02075

[10] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[11] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. 1306–1313.

[12] Abir Chakraborty. 2024. Multi-hop Question Answering over Knowledge Graphs using Large Language Models. arXiv:2404.19234 [cs.AI] https://arxiv.org/abs/2404.19234

[13] Huajun Chen. 2024. Large Knowledge Model: Perspectives and Challenges. arXiv:2312.02706 [cs.AI] https://arxiv.org/abs/2312.02706

[14] Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. 2024. LLaGA: Large Language and Graph Assistant. arXiv:2402.08170 [cs.LG] https://arxiv.org/abs/2402.08170

[15] Shuang Chen, Qian Liu, Zhiwei Yu, Chin-Yew Lin, Jian-Guang Lou, and Feng Jiang. 2021. ReTraCk: A flexible and efficient framework for knowledge base question answering. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing: system demonstrations*. 325–336.

[16] Keyuan Cheng, Gang Lin, Haoyang Fei, Yuxuan zhai, Lu Yu, Muhammad Asif Ali, Lijie Hu, and Di Wang. 2024. Multi-hop Question Answering under Temporal Knowledge Editing. arXiv:2404.00492 [cs.CL] https://arxiv.org/abs/2404.00492

[17] Hyeong Kyu Choi, Seunghun Lee, Jaewon Chu, and Hyunwoo J. Kim. 2023. NuTrea: Neural Tree Search for Context-guided Multi-hop KGQA. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

[18] Nurendra Choudhary and Chandan K. Reddy. 2024. Complex Logical Reasoning over Knowledge Graphs using Large Language Models. arXiv:2305.01157 [cs.LO] https://arxiv.org/abs/2305.01157

[19] Yashar Deldjoo, Zhankui He, Julian McAuley, Anton Korikov, Scott Sanner, Arnau Ramisa, René Vidal, Maheswaran Sathiamoorthy, Atoosa Kasirzadeh, and Silvia Milano. 2024. A Review of Modern Recommender Systems Using Generative Models (Gen-RecSys). arXiv:2404.00579 [cs.IR] https://arxiv.org/abs/2404.00579

[20] Julien Delile, Srayanta Mukherjee, Anton Van Pamel, and Leonid Zhukov. 2024. Graph-Based Retriever Captures the Long Tail of Biomedical Knowledge. arXiv:2402.12352 [cs.CL] https://arxiv.org/abs/2402.12352

[21] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 1811–1818.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[23] Guanting Dong, Rumei Li, Sirui Wang, Yupeng Zhang, Yunsen Xian, and Weiran Xu. 2023. Bridging the KB-Text Gap: Leveraging Structured Knowledge-aware Pre-training for KBQA. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*. 3854–3859.

[24] Abhimanyu Dubey, Abhinav Jauhri, and et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[25] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. arXiv:2404.16130 [cs.CL] https://arxiv.org/abs/2404.16130

[26] Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

[27] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models. arXiv:2405.06211 [cs.CL] https://arxiv.org/abs/2405.06211

[28] Wenqi Fan, Shijie Wang, Jiani Huang, Zhikai Chen, Yu Song, Wenzhuo Tang, Haitao Mao, Hui Liu, Xiaorui Liu, Dawei Yin, and Qing Li. 2024. Graph Machine Learning in the Era of Large Language Models (LLMs). arXiv:2404.14928 [cs.LG] https://arxiv.org/abs/2404.14928

[29] Haishuo Fang, Xiaodan Zhu, and Iryna Gurevych. 2024. DARA: Decomposition-Alignment-Reasoning Autonomous Language Agent for Question Answering over Knowledge Graphs. arXiv:2406.07080 [cs.CL] https://arxiv.org/abs/2406.07080

[30] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2023. Talk like a Graph: Encoding Graphs for Large Language Models. arXiv:2310.04560 [cs.LG] https://arxiv.org/abs/2310.04560

[31] Chao Feng, Xinyu Zhang, and Zichu Fei. 2023. Knowledge Solver: Teaching LLMs to Search for Domain Knowledge from Knowledge Graphs. arXiv:2309.03118 [cs.CL] https://arxiv.org/abs/2309.03118

[32] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable Multi-Hop Relational Reasoning for Knowledge-Aware Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. 1295–1309.

[33] Bin Fu, Yunqi Qiu, Chengguang Tang, Yang Li, Haiyang Yu, and Jian Sun. 2020. A Survey on Complex Question Answering over Knowledge Base: Recent Advances and Challenges. arXiv:2007.13069 [cs.CL] https://arxiv.org/abs/2007.13069

[34] Mikhail Galkin, Xinyu Yuan, Hesham Mostafa, Jian Tang, and Zhaocheng Zhu. 2023. Towards Foundation Models for Knowledge Graph Reasoning. In *The Twelfth International Conference on Learning Representations*.

[35] Hanning Gao, Lingfei Wu, Po Hu, Zhihua Wei, Fangli Xu, and Bo Long. 2022. Graph-augmented Learning to Rank for Querying Large-scale Knowledge Graph. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - Volume 1: Long Papers, Online Only, November 20-23, 2022*. 82–92.

[36] Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. 2024. Two-stage Generative Question Answering on Temporal Knowledge Graph Using Large Language Models. arXiv:2402.16568 [cs.CL] https://arxiv.org/abs/2402.16568

[37] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997 [cs.CL] https://arxiv.org/abs/2312.10997

[38] Aashish Ghimire, James Prather, and John Edwards. 2024. Generative AI in Education: A Study of Educators' Awareness, Sentiments, and Influencing Factors. arXiv:2403.15586 [cs.AI] https://arxiv.org/abs/2403.15586

[39] Yu Gu, Sue Kase, Michelle Vanni, Brian M. Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond I.I.D.: Three Levels of Generalization for Question Answering on Knowledge Bases. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*. 3477–3488.

[40] Yu Gu and Yu Su. 2022. ArcaneQA: Dynamic Program Induction and Contextualized Encoding for Knowledge Base Question Answering. In *Proceedings of the 29th International Conference on Computational Linguistics*. 1718–1731.

[41] Jiayan Guo, Lun Du, Hengyu Liu, Mengyu Zhou, Xinyi He, and Shi Han. 2023. GPT4Graph: Can Large Language Models Understand Graph Structured Data ? An Empirical Evaluation and Benchmarking. arXiv:2305.15066 [cs.AI] https://arxiv.org/abs/2305.15066

[42] Tiezheng Guo, Qingwen Yang, Chen Wang, Yanyi Liu, Pan Li, Jiawei Tang, Dapeng Li, and Yingyou Wen. 2024. KnowledgeNavigator: Leveraging Large Language Models for Enhanced Reasoning over Knowledge Graph. arXiv:2312.15880 [cs.CL] https://arxiv.org/abs/2312.15880

[43] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models. arXiv:2405.14831 [cs.CL] https://arxiv.org/abs/2405.14831

[44] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 1024–1034.

[45] Zhen Han, Yue Feng, and Mingming Sun. 2023. A Graph-Guided Reasoning Approach for Open-ended Commonsense Question Answering. arXiv:2303.10395 [cs.CL] https://arxiv.org/abs/2303.10395

[46] Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving Multi-hop Knowledge Base Question Answering by Learning Intermediate Supervision Signals. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*. 553–561.

[47] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering. arXiv:2402.07630 [cs.LG] https://arxiv.org/abs/2402.07630

[48] Michael Himsolt. 1996. GML: Graph Modelling Language. *University of Passau* (1996).

[49] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. 782–792.

[50] Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2024. GRAG: Graph Retrieval-Augmented Generation. arXiv:2405.16506 [cs.LG] https://arxiv.org/abs/2405.16506

[51] Yucheng Hu and Yuxing Lu. 2024. RAG and RAU: A Survey on Retrieval-Augmented Language Model in Natural Language Processing. arXiv:2404.19543 [cs.CL] https://arxiv.org/abs/2404.19543

[52] Ziniu Hu, Yichong Xu, Wenhao Yu, Shuohang Wang, Ziyi Yang, Chenguang Zhu, Kai-Wei Chang, and Yizhou Sun. 2022. Empowering Language Models with Knowledge Graph Reasoning for Open-Domain Question Answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. 9562–9581.

[53] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. arXiv:2311.05232 [cs.CL] https://arxiv.org/abs/2311.05232

[54] Yizheng Huang and Jimmy Huang. 2024. A Survey on Retrieval-Augmented Text Generation for Large Language Models. arXiv:2404.10981 [cs.IR] https://arxiv.org/abs/2404.10981

[55] Yongfeng Huang, Yanyang Li, Yichong Xu, Lin Zhang, Ruyi Gan, Jiaxing Zhang, and Liwei Wang. 2023. MVP-Tuning: Multi-View Knowledge Retrieval with Prompt Tuning for Commonsense Reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. 13417–13432.

[56] Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (Comet-) Atomic 2020: On Symbolic and Neural Commonsense Knowledge Graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. 6384–6392.

[57] Omid Jafari, Preeti Maurya, Parth Nagarkar, Khandker Mushfiqul Islam, and Chidambaram Crushev. 2021. A Survey on Locality Sensitive Hashing Algorithms and their Applications. arXiv:2102.08942 [cs.DB] https://arxiv.org/abs/2102.08942

[58] Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023. StructGPT: A General Framework for Large Language Model to Reason over Structured Data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. 9237–9251.

[59] Jinhao Jiang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. $Great Truths are Always Simple: $ A Rather Simple Knowledge Encoder for Enhancing the Commonsense Reasoning Capacity of Pre-Trained Models. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*. 1730–1741.

[60] Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. 2024. KG-Agent: An Efficient Autonomous Agent Framework for Complex Reasoning over Knowledge Graph. arXiv:2402.11163 [cs.CL] https://arxiv.org/abs/2402.11163

[61] Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. 2023. UniKGQA: Unified Retrieval and Reasoning for Solving Multi-hop Question Answering Over Knowledge Graph. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

[62] Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. 2023. UniKGQA: Unified Retrieval and Reasoning for Solving Multi-hop Question Answering Over Knowledge Graph. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

[63] Kelvin Jiang, Dekun Wu, and Hui Jiang. 2019. FreebaseQA: A New Factoid QA Data Set Matching Trivia-Style Question-Answer Pairs with Freebase. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. 318–323.

[64] Xinke Jiang, Ruizhe Zhang, Yongxin Xu, Rihong Qiu, Yue Fang, Zhiyuan Wang, Jinyi Tang, Hongxin Ding, Xu Chu, Junfeng Zhao, and Yasha Wang. 2024. HyKGE: A Hypothesis Knowledge Graph Enhanced Framework for Accurate and Reliable Medical LLMs Responses. arXiv:2312.15883 [cs.CL] https://arxiv.org/abs/2312.15883

[65] Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. 2024. Large Language Models on Graphs: A Comprehensive Survey. arXiv:2312.02783 [cs.CL] https://arxiv.org/abs/2312.02783

[66] Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, and Jiawei Han. 2024. Graph Chain-of-Thought: Augmenting Large Language Models by Reasoning on Graphs. arXiv:2404.07103 [cs.CL] https://arxiv.org/abs/2404.07103

[67] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What Disease does this Patient Have? A Large-scale Open Domain Question Answering Dataset from Medical Exams. arXiv:2009.13081 [cs.CL] https://arxiv.org/abs/2009.13081

[68] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. 1601–1611.

[69] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. 6769–6781.

[70] Sohum Kashyap et al. 2024. Knowledge Graph Assisted Large Language Models. (2024).

[71] Jiho Kim, Yeonsu Kwon, Yohan Jo, and Edward Choi. 2023. KG-GPT: A General Framework for Reasoning on Knowledge Graphs Using Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*. 9410–9421.

[72] Jaewoong Kim and Moohong Min. 2024. From RAG to QA-RAG: Integrating Generative AI for Pharmaceutical Regulatory Compliance Process. arXiv:2402.01717 [cs.CL] https://arxiv.org/abs/2402.01717

[73] Jiho Kim, Sungjin Park, Yeonsu Kwon, Yohan Jo, James Thorne, and Edward Choi. 2023. FactKG: Fact Verification via Reasoning on Knowledge Graphs. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. 16190–16206.

[74] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[75] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: a Benchmark for Question Answering Research. *Trans. Assoc. Comput. Linguistics* 7 (2019), 452–466.

[76] Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A Survey on Complex Knowledge Base Question Answering: Methods, Challenges and Solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*. 4483–4491.

[77] Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Complex Knowledge Base Question Answering: A Survey. *IEEE Trans. Knowl. Data Eng.* 35, 11 (2023), 11196–11215.

[78] Yunshi Lan and Jing Jiang. 2020. Query Graph Generation for Answering Multi-hop Complex Questions from Knowledge Bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. 969–974.

[79] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. 3045–3059.

[80] Dawei Li, Shu Yang, Zhen Tan, Jae Young Baik, Sukwon Yun, Joseph Lee, Aaron Chacko, Bojian Hou, Duy Duong-Tran, Ying Ding, Huan Liu, Li Shen, and Tianlong Chen. 2024. DALK: Dynamic Co-Augmentation of LLMs and KG to answer Alzheimer's Disease Questions with Scientific Literature. arXiv:2405.04819 [cs.CL] https://arxiv.org/abs/2405.04819

[81] Shiyang Li, Yifan Gao, Haoming Jiang, Qingyu Yin, Zheng Li, Xifeng Yan, Chao Zhang, and Bing Yin. 2023. Graph Reasoning for Question Answering with Triplet Retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. 3366–3375.

[82] Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. 4582–4597.

[83] Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. 2024. A Survey of Graph Meets Large Language Model: Progress and Future Directions. arXiv:2311.12399 [cs.LG] https://arxiv.org/abs/2311.12399

[84] Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. 2024. Large Language Models in Finance: A Survey. arXiv:2311.10723 [q-fin.GN] https://arxiv.org/abs/2311.10723

[85] Yihao Li, Ru Zhang, and Jianyi Liu. 2024. An Enhanced Prompt-Based LLM Reasoning Scheme via Knowledge Graph-Integrated Collaboration. arXiv:2402.04978 [cs.CL] https://arxiv.org/abs/2402.04978

[86] Zhuoyang Li, Liran Deng, Hui Liu, Qiaoqiao Liu, and Junzhao Du. 2024. UniOQA: A Unified Framework for Knowledge Graph Question Answering with Large Language Models. arXiv:2406.02110 [cs.CL] https://arxiv.org/abs/2406.02110

[87] Zijian Li, Qingyan Guo, Jiawei Shao, Lei Song, Jiang Bian, Jun Zhang, and Rui Wang. 2024. Graph Neural Network Enhanced Retrieval for Question Answering of LLMs. arXiv:2406.06572 [cs.CL] https://arxiv.org/abs/2406.06572

[88] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-Aware Graph Networks for Commonsense Reasoning. 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 2019, Hong Kong, China, November 3-7, 2019*. 2829–2839.

[89] Bill Yuchen Lin, Ziyi Wu, Yichi Yang, Dong-Ho Lee, and Xiang Ren. 2021. RiddleSense: Reasoning about Riddle Questions Featuring Linguistic Creativity and Commonsense Knowledge. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021 (Findings of ACL, Vol. ACL/IJCNLP 2021)*. 1504–1515.

[90] Guangyi Liu, Yongqi Zhang, Yong Li, and Quanming Yao. 2024. Explore then Determine: A GNN-LLM Synergy Framework for Reasoning over Knowledge Graph. arXiv:2406.01145 [cs.CL] https://arxiv.org/abs/2406.01145

[91] H Liu and P Singh. 2004. ConceptNet—a practical commonsense reasoning tool-kit. *BT technology journal* 22, 4 (2004), 211–226.

[92] Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S. Yu, and Chuan Shi. 2024. Towards Graph Foundation Models: A Survey and Beyond. arXiv:2310.11829 [cs.LG] https://arxiv.org/abs/2310.11829

[93] Lei Liu, Xiaoyan Yang, Junchi Lei, Xiaoyang Liu, Yue Shen, Zhiqiang Zhang, Peng Wei, Jinjie Gu, Zhixuan Chu, Zhan Qin, and Kui Ren. 2024. A Survey on Medical Large Language Models: Technology, Application, Trustworthiness, and Future Directions. arXiv:2406.03712 [cs.CL] https://arxiv.org/abs/2406.03712

[94] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Trans. Assoc. Comput. Linguistics* 12 (2024), 157–173.

[95] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 61–68.

[96] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. GPT Understands, Too. arXiv:2103.10385 [cs.CL] https://arxiv.org/abs/2103.10385

[97] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs.CL] https://arxiv.org/abs/1907.11692

[98] Pei-Chi Lo and Ee-Peng Lim. 2023. Contextual Path Retrieval: A Contextual Entity Relation Embedding-based Approach. *ACM Trans. Inf. Syst.* 41, 1 (2023), 1:1–1:38.

[99] Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-Shot Entity Linking by Reading Entity Descriptions. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. 3449–3460.

[100] Dan Luo, Jiawei Sheng, Hongbo Xu, Lihong Wang, and Bin Wang. 2023. Improving Complex Knowledge Base Question Answering with Relation-Aware Subgraph Retrieval and Reasoning Network. In *International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, June 18-23, 2023*. 1–8.

[101] Haoran Luo, Haihong E, Zichen Tang, Shiyao Peng, Yikai Guo, Wentai Zhang, Chenghao Ma, Guanting Dong, Meina Song, Wei Lin, Yifan Zhu, and Luu Anh Tuan. 2024. ChatKBQA: A Generate-then-Retrieve Framework for Knowledge Base Question Answering with Fine-tuned Large Language Models. arXiv:2310.08975 [cs.CL] https://arxiv.org/abs/2310.08975

[102] Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2024. Reasoning on Graphs: Faithful and Interpretable Large Language Model Reasoning. arXiv:2310.01061 [cs.CL] https://arxiv.org/abs/2310.01061

[103] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query Rewriting for Retrieval-Augmented Large Language Models. arXiv:2305.14283 [cs.CL] https://arxiv.org/abs/2305.14283

[104] Haitao Mao, Zhikai Chen, Wenzhuo Tang, Jianan Zhao, Yao Ma, Tong Zhao, Neil Shah, Mikhail Galkin, and Jiliang Tang. 2024. Position: Graph Foundation Models Are Already Here. In *Forty-first International Conference on Machine Learning*.

[105] Qiheng Mao, Zemin Liu, Chenghao Liu, Zhuo Li, and Jianling Sun. 2024. Advancing Graph Representation Learning with Large Language Models: A Comprehensive Survey of Techniques. arXiv:2402.05952 [cs.LG] https://arxiv.org/abs/2402.05952

[106] Shengyu Mao, Yong Jiang, Boli Chen, Xiao Li, Peng Wang, Xinyu Wang, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. 2024. RaFe: Ranking Feedback Improves Query Rewriting for RAG. arXiv:2405.14431 [cs.CL] https://arxiv.org/abs/2405.14431

[107] Costas Mavromatis and George Karypis. 2022. ReaRev: Adaptive Reasoning for Question Answering over Knowledge Graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. 2447–2458.

[108] Costas Mavromatis and George Karypis. 2024. GNN-RAG: Graph Neural Retrieval for Large Language Model Reasoning. arXiv:2405.20139 [cs.CL] https://arxiv.org/abs/2405.20139

[109] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. 2381–2391.

[110] Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-Value Memory Networks for Directly Reading Documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. 1400–1409.

[111] Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable Conversational Reasoning with Attention-based Walks over Knowledge Graphs. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. 845–854.

[112] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. 2020. TU-Dataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*.

[113] Sai Munikoti, Anurag Acharya, Sridevi Wagle, and Sameera Horawalavithana. 2023. ATLANTIC: Structure-Aware Retrieval-Augmented Language Model for Interdisciplinary Science. arXiv:2311.12289 [cs.CL] https://arxiv.org/abs/2311.12289

[114] Yuqi Nie, Yaxuan Kong, Xiaowen Dong, John M. Mulvey, H. Vincent Poor, Qingsong Wen, and Stefan Zohren. 2024. A Survey of Large Language Models for Financial Applications: Progress, Prospects and Challenges. arXiv:2406.11903 [q-fin.GN] https://arxiv.org/abs/2406.11903

[115] Yasumasa Onoe, Michael J. Q. Zhang, Eunsol Choi, and Greg Durrett. 2021. CREAK: A Dataset for Commonsense Reasoning over Entity Knowledge. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

[116] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] https://arxiv.org/abs/2303.08774

[117] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[118] Vardaan Pahuja, Boshi Wang, Hugo Latapie, Jayanth Srinivasa, and Yu Su. 2023. A Retrieve-and-Read Framework for Knowledge Graph Link Prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*. 1992–2002.

[119] Jeff Z. Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, Russa Biswas, Gerard de Melo, Angela Bonifati, Edlira Vakaj, Mauro Dragoni, and Damien Graux. 2023. Large Language Models and Knowledge Graphs: Opportunities and Challenges. *TGDK* 1, 1 (2023), 2:1–2:38.

[120] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE Trans. Knowl. Data Eng.* 36, 7 (2024), 3580–3599.

[121] Wenjun Peng, Guiyang Li, Yue Jiang, Zilong Wang, Dan Ou, Xiaoyi Zeng, Derong Xu, Tong Xu, and Enhong Chen. 2024. Large Language Model based Long-tail Query Rewriting in Taobao Search. In *Companion Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, Singapore, May 13-17, 2024*. 20–28.

[122] Zhuoyi Peng and Yi Yang. 2024. Connecting the Dots: Inferring Patent Phrase Similarity with Retrieved Phrase Graphs. arXiv:2403.16265 [cs.CL] https://arxiv.org/abs/2403.16265

[123] Aleksandr Perevalov, Dennis Diefenbach, Ricardo Usbeck, and Andreas Both. 2022. QALD-9-plus: A Multilingual Dataset for Question Answering over DBpedia and Wikidata Translated by Native Speakers. In *16th IEEE International Conference on Semantic Computing, ICSC 2022, Laguna Hills, CA, USA, January 26-28, 2022*. 229–234.

[124] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick S. H. Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a Benchmark for Knowledge Intensive Language Tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*. 2523–2544.

[125] Zhixiao Qi, Yijiong Yu, Meiqi Tu, Junyi Tan, and Yongfeng Huang. 2023. FoodGPT: A Large Language Model in Food Testing Domain with Incremental Pre-training and Knowledge Graph Prompt. arXiv:2308.10173 [cs.CL] https://arxiv.org/abs/2308.10173

[126] Zile Qiao, Wei Ye, Yong Jiang, Tong Mo, Pengjun Xie, Weiping Li, Fei Huang, and Shikun Zhang. 2024. Supportiveness-based Knowledge Rewriting for Retrieval-augmented Language Modeling. arXiv:2406.08116 [cs.CL] https://arxiv.org/abs/2406.08116

[127] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67.

[128] Priyanka Ranade and Anupam Joshi. 2023. FABULA: Intelligence Report Generation Using Retrieval-Augmented Narrative Construction. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2023, Kusadasi, Turkey, November 6-9, 2023*. 603–610.

[129] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. 3980–3990.

[130] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

[131] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: An Atlas of Machine Commonsense for If-Then Reasoning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. 3027–3035.

[132] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. SocialIQA: Commonsense Reasoning about Social Interactions. arXiv:1904.09728 [cs.CL] https://arxiv.org/abs/1904.09728

[133] Apoorv Saxena, Aditay Tripathi, and Partha P. Talukdar. 2020. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. 4498–4507.

[134] Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. Mintaka: A Complex, Natural, and Multilingual Dataset for End-to-End Question Answering. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*. 1604–1619.

[135] Ahsan Shehzad, Feng Xia, Shagufta Abid, Ciyuan Peng, Shuo Yu, Dongyu Zhang, and Karin Verspoor. 2024. Graph Transformers: A Survey. arXiv:2407.09777 [cs.LG] https://arxiv.org/abs/2407.09777

[136] Yiheng Shu, Zhiwei Yu, Yuhan Li, Börje F. Karlsson, Tingting Ma, Yuzhong Qu, and Chin-Yew Lin. 2022. TIARA: Multi-grained Retrieval for Robust Question Answering over Large Knowledge Bases. arXiv:2210.12925 [cs.CL] https://arxiv.org/abs/2210.12925

[137] Saurabh Srivastava, Milind D Jain, Harshita Jain, Kritik Jaroli, VJ Mayank Patel, and L Khan. 2020. IOT monitoring bin for smart cities. In *3rd Smart Cities Symposium (SCS 2020)*, Vol. 2020. IET, 533–536.

[138] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*. 697–706.

[139] Haitian Sun, Tania Bedrax-Weiss, and William W. Cohen. 2019. 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 2019, Hong Kong, China, November 3-7, 2019*. 2380–2390.

[140] Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W. Cohen. 2018. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. 4231–4242.

[141] Hao Sun, Yang Li, Liwei Deng, Bowen Li, Binyuan Hui, Binhua Li, Yunshi Lan, Yan Zhang, and Yongbin Li. 2023. History Semantic Graph Enhanced Conversational KBQA with Temporal Information Modeling. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. 3521–3533.

[142] Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-Graph: Deep and Responsible Reasoning of Large Language Model on Knowledge Graph. arXiv:2307.07697 [cs.CL] https://arxiv.org/abs/2307.07697

[143] Lei Sun, Zhengwei Tao, Youdi Li, and Hiroshi Arakawa. 2024. ODA: Observation-Driven Agent for integrating LLMs and Knowledge Graphs. arXiv:2404.07677 [cs.CL] https://arxiv.org/abs/2404.07677

[144] Alon Talmor and Jonathan Berant. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. 641–651.

[145] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. 4149–4158.

[146] Dhaval Taunk, Lakshya Khanna, Siri Venkata Pavan Kumar Kandru, Vasudeva Varma, Charu Sharma, and Makarand Tapaswi. 2023. GrapeQA: GRaph Augmentation and Pruning to Enhance Question-Answering. In *Companion Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*. 1138–1144.

[147] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing Text for Joint Embedding of Text and Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. 1499–1509.

[148] Hugo Touvron, Louis Martin, and et al. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL] https://arxiv.org/abs/2307.09288

[149] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual*

Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. 5998–6008.

[150] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. arXiv:1710.10903 [stat.ML] https://arxiv.org/abs/1710.10903

[151] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[152] Chaojie Wang, Yishi Xu, Zhong Peng, Chenxi Zhang, Bo Chen, Xinrun Wang, Lei Feng, and Bo An. 2023. keqing: knowledge-based question answering is a nature chain-of-thought mentor of LLM. arXiv:2401.00426 [cs.CL] https://arxiv.org/abs/2401.00426

[153] Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023. Can Language Models Solve Graph Problems in Natural Language?. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

[154] Jinqiang Wang, Huansheng Ning, Yi Peng, Qikai Wei, Daniel Tesfai, Wenwei Mao, Tao Zhu, and Runhe Huang. 2024. A Survey on Large Language Models from General Purpose to Medical Applications: Datasets, Methodologies, and Evaluations. arXiv:2406.10303 [cs.CL] https://arxiv.org/abs/2406.10303

[155] Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023. Knowledge-Driven CoT: Exploring Faithful Reasoning in LLMs for Knowledge-intensive Question Answering. arXiv:2308.13259 [cs.CL] https://arxiv.org/abs/2308.13259

[156] Ruijie Wang, Zheng Li, Danqing Zhang, Qingyu Yin, Tong Zhao, Bing Yin, and Tarek F. Abdelzaher. 2022. RETE: Retrieval-Enhanced Temporal Event Forecasting on Unified Query Product Evolutionary Graph. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. 462–472.

[157] Shen Wang, Tianlong Xu, Hang Li, Chaoli Zhang, Joleen Liang, Jiliang Tang, Philip S. Yu, and Qingsong Wen. 2024. Large Language Models for Education: A Survey and Outlook. arXiv:2403.18105 [cs.CL] https://arxiv.org/abs/2403.18105

[158] Xintao Wang, Qianwen Yang, Yongting Qiu, Jiaqing Liang, Qianyu He, Zhouhong Gu, Yanghua Xiao, and Wei Wang. 2023. KnowledGPT: Enhancing Large Language Models with Retrieval and Storage Access on Knowledge Bases. arXiv:2308.11761 [cs.CL] https://arxiv.org/abs/2308.11761

[159] Yuqi Wang, Boran Jiang, Yi Luo, Dawei He, Peng Cheng, and Liangcai Gao. 2024. Reasoning on Efficient Knowledge Paths:Knowledge Graph Guides Large Language Model for Domain Question Answering. arXiv:2404.10384 [cs.CL] https://arxiv.org/abs/2404.10384

[160] Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa F. Siu, Ruiyi Zhang, and Tyler Derr. 2024. Knowledge Graph Prompting for Multi-Document Question Answering. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*. 19206–19214.

[161] Yaoke Wang, Yun Zhu, Wenqiao Zhang, Yueting Zhuang, Yunfei Li, and Siliang Tang. 2024. Bridging Local Details and Global Context in Text-Attributed Graphs. arXiv:2406.12608 [cs.CL] https://arxiv.org/abs/2406.12608

[162] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM international conference on multimedia*. 1437–1445.

[163] Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. MindMap: Knowledge Graph Prompting Sparks Graph of Thoughts in Large Language Models. arXiv:2308.09729 [cs.AI] https://arxiv.org/abs/2308.09729

[164] Sondre Wold, Lilja Øvrelid, and Erik Velldal. 2023. Text-To-KG Alignment: Comparing Current Methods on Classification Tasks. arXiv:2306.02871 [cs.CL] https://arxiv.org/abs/2306.02871

[165] Shangyu Wu, Ying Xiong, Yufei Cui, Haolun Wu, Can Chen, Ye Yuan, Lianming Huang, Xue Liu, Tei-Wei Kuo, Nan Guan, and Chun Jason Xue. 2024. Retrieval-Augmented Generation for Natural Language Processing: A Survey. arXiv:2407.13193 [cs.CL] https://arxiv.org/abs/2407.13193

[166] Shirley Wu, Shiyu Zhao, Michihiro Yasunaga, Kexin Huang, Kaidi Cao, Qian Huang, Vassilis N. Ioannidis, Karthik Subbian, James Zou, and Jure Leskovec. 2024. STaRK: Benchmarking LLM Retrieval on Textual and Relational Knowledge Bases. arXiv:2404.13207 [cs.IR] https://arxiv.org/abs/2404.13207

[167] Taiqiang Wu, Xingyu Bai, Weigang Guo, Weijie Liu, Siheng Li, and Yujiu Yang. 2023. Modeling Fine-grained Information via Knowledge-aware Hierarchical Graph for Zero-shot Entity Retrieval. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 - 3 March 2023*. 1021–1029.

[168] Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023. Retrieve-Rewrite-Answer: A KG-to-Text Enhanced LLMs Framework for Knowledge Graph Question Answering. arXiv:2309.11206 [cs.CL] https://arxiv.org/abs/2309.11206

[169] Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. 2024. Retrieval-Augmented Generation with Knowledge Graphs for Customer Service Question Answering. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*. 2905–2909.

[170] An Yang, Baosong Yang, and et al. 2024. Qwen2 Technical Report. arXiv:2407.10671 [cs.CL] https://arxiv.org/abs/2407.10671

[171] Rui Yang, Haoran Liu, Edison Marrese-Taylor, Qingcheng Zeng, Yu He Ke, Wanxin Li, Lechao Cheng, Qingyu Chen, James Caverlee, Yutaka Matsuo, and Irene Li. 2024. KG-Rank: Enhancing Large Language Models for Medical QA with Knowledge Graphs and Ranking Techniques. arXiv:2403.05881 [cs.CL] https://arxiv.org/abs/2403.05881

[172] Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen tau Yih, and Xin Luna Dong. 2024. CRAG – Comprehensive RAG Benchmark. arXiv:2406.04744 [cs.CL] https://arxiv.org/abs/2406.04744

[173] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. 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, October 31 - November 4, 2018*. 2369–2380.

[174] Mohammad Yani and Adila Alfa Krisnadhi. 2021. Challenges, Techniques, and Trends of Simple Knowledge Graph Question Answering: A Survey. *Inf.* 12, 7 (2021), 271.

[175] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*. 535–546.

[176] Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2024. Language is All a Graph Needs. arXiv:2308.07134 [cs.CL] https://arxiv.org/abs/2308.07134

[177] Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2021. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. *arXiv preprint arXiv:2109.08678* (2021).

[178] Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The Value of Semantic Parse Labeling for Knowledge Base Question Answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*.

[179] Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Yang Wang, Zhiguo Wang, and Bing Xiang. 2023. DecAF: Joint Decoding of Answers and Logical Forms for Question Answering over Knowledge Bases. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

[180] Hao Yu, Aoran Gan, Kai Zhang, Shiwei Tong, Qi Liu, and Zhaofeng Liu. 2024. Evaluation of Retrieval-Augmented Generation: A Survey. arXiv:2405.07437 [cs.CL] https://arxiv.org/abs/2405.07437

[181] Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph Retrieval Enhanced Model for Multi-hop Knowledge Base Question Answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. 5773–5784.

[182] Mengmei Zhang, Mingwei Sun, Peng Wang, Shen Fan, Yanhu Mo, Xiaoxiao Xu, Hong Liu, Cheng Yang, and Chuan Shi. 2024. GraphTranslator: Aligning Graph Model to Large Language Model for Open-ended Tasks. In *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*. 1003–1014.

[183] Qinggang Zhang, Junnan Dong, Hao Chen, Daochen Zha, Zailiang Yu, and Xiao Huang. 2024. KnowGPT: Knowledge Graph based Prompting for Large Language Models. arXiv:2312.06185 [cs.CL] https://arxiv.org/abs/2312.06185

[184] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. 2022. GreaseLM: Graph REASoning Enhanced Language Models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*.

[185] Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2018. Variational Reasoning for Question Answering With Knowledge Graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 6069–6076.

[186] Jianan Zhao, Le Zhuo, Yikang Shen, Meng Qu, Kai Liu, Michael Bronstein, Zhaocheng Zhu, and Jian Tang. 2023. GraphText: Graph Reasoning in Text Space. arXiv:2310.01089 [cs.CL] https://arxiv.org/abs/2310.01089

[187] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. 2024. Retrieval-Augmented Generation for AI-Generated Content: A Survey. arXiv:2402.19473 [cs.CV] https://arxiv.org/abs/2402.19473

[188] Yanxin Zheng, Wensheng Gan, Zefeng Chen, Zhenlian Qi, Qian Liang, and Philip S. Yu. 2024. Large Language Models for Medicine: A Survey. arXiv:2405.13055 [cs.CL] https://arxiv.org/abs/2405.13055

[189] Yun Zhu, Yaoke Wang, Haizhou Shi, and Siliang Tang. 2024. Efficient Tuning and Inference for Large Language Models on Textual Graphs. arXiv:2401.15569 [cs.CL] https://arxiv.org/abs/2401.15569