Enhancing Data Traceability: A Knowledge Graph Approach with Retrieval-Augmented Generation

1st Vincencius Christiano Tjokro
Information Systems Department
Faculty of Technology and Informatics
Universitas Multimedia Nusantara
Tangerang, Indonesia
vincencius@student.umn.ac.id

2nd Samuel Ady Sanjaya
Information Systems Department
Faculty of Engineering and Informatics
Universitas Multimedia Nusantara
Tangerang, Indonesia
samuel.ady@umn.ac.id

2024 7th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) | 979-8-3315-1964-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISRITI64779.2024.10963652

***Abstract*— This study explores the challenges of data traceability within Company ABC's Master Data Management (MDM) system, where handling data-related requests has become increasingly complex due to inefficient manual processes. Despite MDM's implementation, business growth has led to the accumulation of documents and difficulties in tracking request lineage, causing bottlenecks in daily operations. The study proposes a solution that integrates knowledge graphs with Retrieval-Augmented Generation (RAG), inspired by LinkedIn's question-answering system, to improve data management and traceability. The knowledge graph is constructed by manually parsing spreadsheet data related to requests, allowing for more efficient data integration and retrieval. The study compares three approaches—graph-based, vector-based, and hybrid—in terms of answer correctness and context retrieval. The graph-based approach yielded the best performance, improving answer correctness by 21.09% over the vector-based method and by 19.34% over the hybrid method. While the hybrid method achieved the highest context precision at 83.49%, it struggled with context recall, retrieving relevant context only 69.09% of the time. The study also found that the vector-based method's retrieval of irrelevant context caused the language model to produce misleading answers, even when accompanied by more relevant graph-based context. In conclusion, integrating RAG with knowledge graphs significantly enhances answer accuracy and context retrieval precision. Future research should focus on refining context-weighting techniques between the vector and graph-based methods and on addressing the issue of irrelevant context to further improve the overall effectiveness of the model in data traceability and information retrieval tasks.**

Keywords— *Large Language Model, Knowledge Graph, Retrieval-Augmented Generation, RAG, Data Traceability, Vector Retrieval*

I. INTRODUCTION

An organization's ability to make informed business decisions, foster innovation, and sustain a competitive advantage is deeply dependent on the quality of its data, particularly in terms of accuracy, consistency, and completeness [1]. Master Data Management (MDM) serves as a key methodology to tackle issues related to data quality and governance, enabling standardization and accessibility of data across different departments [2]. Company ABC has successfully adopted MDM as a centralized function, revolutionizing its data management and governance processes organization-wide. Data-related requests encompass all processes related to master or transactional data, such as article data creation or deletion, purchasing information, and extending data to other business units.

Despite the implementation of MDM, data-related requests faced challenges due to poor data integration and management. Currently, each individual in the MDM department manually logs changes made during the request processing using text documents. As business growth accelerates, the complexity and volume of these documents increase, making it difficult to trace the lineage of each request. This creates bottlenecks and results in non-value-added tasks, as team members must sift through documents during business operations. To overcome this issue, proper data integration and efficient information retrieval are essential. This study proposes the integration of knowledge graphs and retrieval-augmented generation as a solution.

Retrieval-Augmented Generation (RAG) is a method in natural language processing (NLP) that combines the strengths of retrieval-based systems and generative models such as the Large Language Model (LLM) [3]. The retrieval component of RAG identifies and retrieves the most pertinent information from a database based on an input query. This information subsequently serves as context for LLM to generate responses that are both contextually accurate and informative. RAG effectively addresses challenges related to hallucination, outdated information, and the incorporation of domain-specific knowledge [4]. However, the performance of RAG generation is heavily contingent on the effectiveness of the retrieval component; poor retrieval can lead to the retrieval of inaccurate and irrelevant information, resulting in hallucinations and misleading responses. [5]. Currently, most RAG frameworks heavily rely on vector retrieval as their retrieval component, which retrieves the top-k information by assessing similarity among vector embeddings. However, this approach has several limitations, including inadequate capture of conceptual correlations, insufficient aggregation of diverse facts, and an inability to effectively handle complex logical reasoning [6]. To address this limitation, this research will incorporate structured insight from knowledge graphs as a navigational tool.

A knowledge graph is a structured representation of real-world information, where data is organized as a graph with nodes representing key entities and edges illustrating their relationships. This approach is particularly valuable for managing and processing complex, diverse data, enabling machines to interpret and analyze it more effectively. By modeling interconnected information, knowledge graphs provide a flexible and efficient solution for handling heterogeneous data [7]. Unlike NoSQL databases, which primarily support only traditional relational operators, knowledge graphs enable navigational operators, allowing for the traversal of connected entities within the graph, enhancing the ability to represent and explore intricate relationships [8]. Understanding these capabilities, knowledge graphs have been applied to various downstream tasks, including recommendation systems, question-answering, and information retrieval, as well as domain-specific applications such as news, healthcare, finance, and

education [9]. This approach enhances information collection and integration, ensuring factual consistency and fostering logical coherence. Integration with RAG enables Large Language Models (LLMs) to reason more like humans, allowing them to analyze relevant clues while continuously exploring topics. This synergy ultimately leads to more accurate and contextually rich answers [6].

The integration of RAG with knowledge graphs has demonstrated successful outcomes across various domains. In customer service, for example, this integration boosted retrieval recall@1 from 40% to 86% by maintaining information structure and preventing text segmentation. This improvement also enhanced generation performance by 32% on the BLEU metric and reduced issue resolution time by 28.6% [10]. In complex domains requiring high precision, such as biomedicine, this approach significantly enhances accuracy, improving it by 18.7% to reach 71.8% in multiple-choice questions and by 11.7% to achieve 80.3% in binary classification [11]. Knowledge graph and RAG outperform vector-based retrieval RAG on several evaluation metrics including: comprehensiveness, diversity, and empowerment [12]. Additionally, this approach reduces hallucinations by 50% [13]. However, rather than choosing between graph-based or vector-based retriever alone, a hybrid approach that combines both shows significant improvements in accuracy—up to 100%, with increases of 78.7% and 91.5%, and a 70.8% improvement in correctness [14]. Applying this hybrid approach to long-tail biomedical data results in a precision of 100% [15].

This study leverages RAG alongside a hybrid approach that integrates both graph-based and vector-based retrieval to address data traceability challenges. By combining the strengths of structured knowledge with the flexibility of vector-based retrieval, the goal is to improve the accuracy and reliability of information retrieval. Additionally, the study conducts a comparative assessment of each retrieval component, evaluating their individual contributions to performance and identifying the most effective configurations for solving traceability issues.

II. Methods

To enhance data traceability, this study proposes integrating a knowledge graph with Retrieval-Augmented Generation (RAG), drawing inspiration from the methodology used in LinkedIn's customer service question answering system [10]. As shown in Figure I, the approach consists of two phases: the construction of the knowledge graph and the integration of the RAG component. The construction of the knowledge graph involves manually parsing through all request-related spreadsheet data.

A. Data Collection

Maintaining the privacy and security of request-related data throughout the entire study remains the top priority. Equally important is ensuring the inclusivity of the data used in the research to support the generalization of the proposed solutions. To address these concerns, an interview with the Master Data Management (MDM) department was conducted. The discussions concluded that only two types of request data will be provided for this study: Purchasing Info Record (PIR) and Extend. Several considerations behind this decision are as follows:

1) Data Inclusivity
   Both PIR and Extend requests are representative of other request types in terms of information complexity and frequency. Additionally, these two request types are frequently queried for their data lineage, making them valuable for analysis in the study.
2) Data Privacy
   According to company data governance and compliance regulations, request data is permitted for use strictly within the project scope. However, it is not intended for public exposure, as this ensures privacy and reduces the risk of sensitive data leakage.

In this step, comprehensive request data from August 2024 were collected, with usage restricted to approved research projects only. The data is organized in a structured table, where each row represents a single request along with its corresponding details.

B. Knowledge Graph Construction

This study employs Neo4j, a highly scalable, Java-based graph database management system (GDMS), for the storage of knowledge graph data [16]. As a property graph database, Neo4j leverages Cypher, a declarative query language specifically designed for querying and modifying graph structures. Cypher facilitates the execution of complex queries, enabling the management of intricate state transitions and dependencies, thus supporting efficient and expressive interactions with highly connected datasets [17].

1) Schema Construction
   A schema or ontology defines the properties of a specific domain and the relationships between them, making it essential for knowledge graph construction [7]. Additionally, schema creation is critical in practical implementations, as it imposes constraints on knowledge graph links, ensuring that relationships between entities are well-structured and clearly defined [16]. This study implements a property graph as the data model, allowing each node or edge to maintain property-value pairs. The mathematical representation of the property graph is illustrated in Equation (1).

$$G = (N, E, \rho, \lambda, \sigma) \quad (1)$$

   where,
   N is a set of nodes (vertices)
   E is a set of edges, separate from the nodes
   ρ connects each edge in **E** to two nodes in **N**
   λ assigns labels to each node or edge
   σ assigns properties to nodes or edges

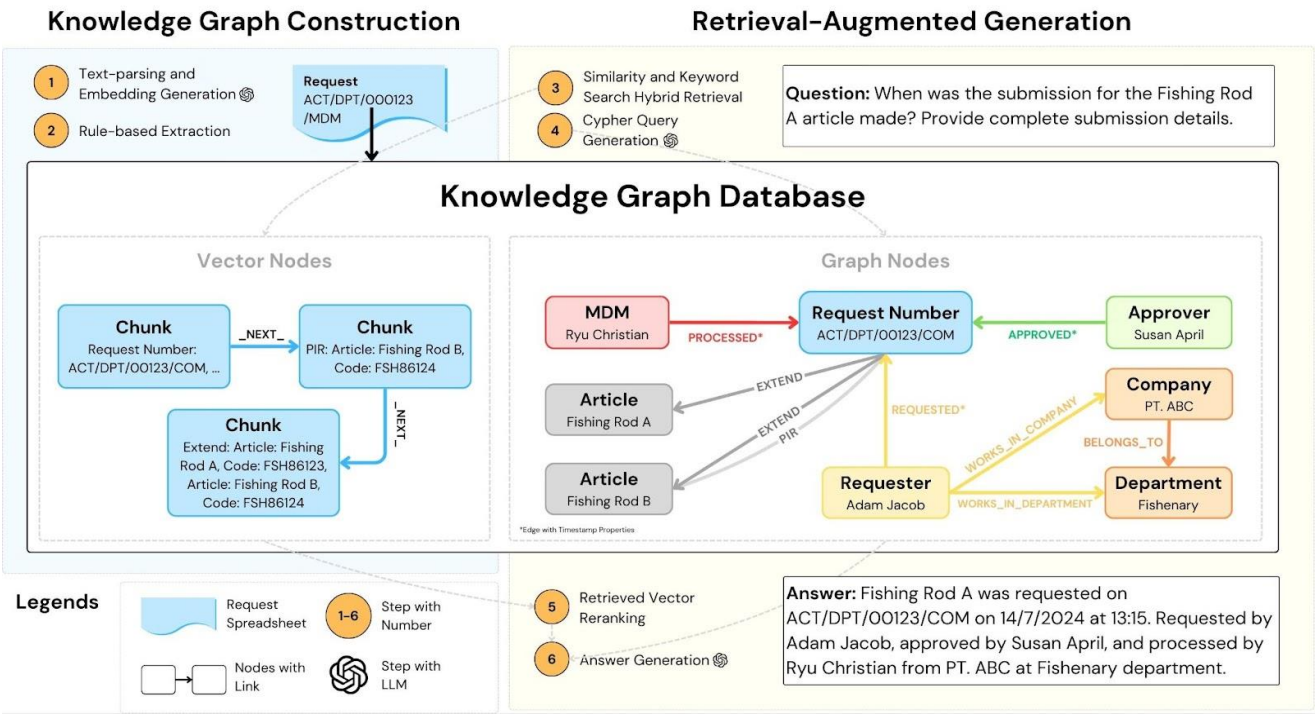

Fig 1. Overview of the Proposed Methodology for Integrating a Knowledge Graph with Retrieval-Augmented Generation (RAG). The left side illustrates the construction of the knowledge graph, while the right side depicts the RAG integration process for question-answering tasks

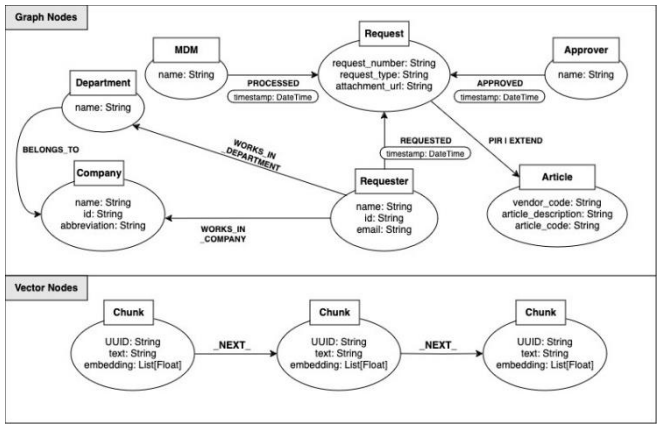

Fig 2. Knowledge Graph Schema. The upper-half side graph-nodes represent request workflow within company ABC. The lower-half side represent details of request as vector.

Figure 2 illustrates the constructed property graph schema. In this schema, two types of nodes are defined to optimize the use of Neo4j for storing vector data. The graph nodes represent data in the knowledge graph, while the vector nodes contain vector embedding data, which are subsequently indexed for efficient retrieval.

2) Vector Nodes Construction

Large Language Models (LLMs) are constrained by a finite context length, which limits their ability to process extensive information in a single input. To overcome this limitation, a structured chunking strategy is employed [17]. In the construction of vector nodes, each chunk is assigned an auto-generated universally unique identifier (UUID), along with corresponding text and embedding properties. The text for each chunk is generated by concatenating key-value pairs, separated by commas, extracted from tabular data. This method ensures that the text retains a structured format, with each key-value pair representing a relevant piece of information. The first chunk node provides an overview of the request, maintaining essential context. Subsequent chunk nodes focus on detailed information related to the request, such as articles and associated activities (like PIR or Extend). Each chunk is limited to 15 articles to ensure manageability. Once the text is generated, it is converted into embedding vectors using OpenAI's embedding model, text-embedding-small-3, chosen for its cost-efficiency and high-quality output.

3) Graph Nodes Construction

The construction of graph nodes is performed through rule-based extraction from the request spreadsheets, guided by a predefined schema. Predefined Cypher queries are created, populated with relevant data, and executed to generate the knowledge graph.

C. RAG Integration

LangChain is utilized in this study as an open-source framework that simplifies the development of applications based on large language models (LLMs). It provides abstract modular components and pre-built chains, which serve as ideal starting points for building these applications [18].

1) Vector-based retrieval

Vector-based retrieval components are built using a hybrid search and re-ranking architecture. This approach improves retrieval accuracy by combining vector similarity with keyword search. After the initial retrieval, a re-ranking phase is applied to further enhance relevance, ensuring the most pertinent documents appear at the top [19]. In a graph database, vector data are stored as nodes, and hybrid search is implemented using Cypher queries

through pre-defined search templates in Neo4jVector. However, based on the constructed schema, only the first Chunk node contains the overview information related to the request. Consequently, it is necessary to construct a custom retrieval query to integrate the matched Chunk nodes with the initial Chunk node, as illustrated in Figure 3.

```
WITH node, score
MATCH path = (first_node)-[:_NEXT_*0..]->(node)
WHERE NOT ()-[:_NEXT_]->(first_node)
WITH first_node, node, score
RETURN
 {concatenatedText: CASE
   WHEN first_node = node THEN first_node.text
   ELSE first_node.text + node.text
 END} AS text,
 score,
 {foo: ""} AS metadata
```

Fig 3. Custom Retrieval Query

Vector Similarity Search

The retrieval of vector nodes utilizes cosine similarity to conduct vector similarity searches. Cosine similarity measures the cosine of the angle between two embedding vectors, providing an indication of their semantic similarity [20]. Cosine similarity is calculated based on the formula on equation (2) [21].

$$CosSim(A,B) = \cos(\theta) = \frac{A \cdot B}{|A|\,|B|} \quad (2)$$

where,
A and **B**, each represent a vector data point
θ (theta), represent angle between two data point

Keyword Search

In information retrieval, keyword search refers to a method that identifies documents containing one or more keywords specified by the user [22]. Keyword search in Neo4j is implemented through the creation of a full-text index, which allows for content matching based on text string properties within Chunk nodes [23].

Reranker

Reranker in the retrieval component plays a vital role in effectively filtering retrieved documents by identifying connections between them and selecting the most relevant ones. It leverages pre-trained models, such as BERT-based architectures, to enhance reranking performance by estimating relevance scores between questions and documents [24]. In this study, FlashRank is employed as a lightweight reranking library integrated with LangChain. The model ms-macro-MultiBert-L-12 was chosen for its multilingual capabilities, including proficiency in Indonesian as the primary query language [25].

2) Graph-based Retrieval

Recognizing the significance of Cypher for graph-based retrieval components, text-to-Cypher translation proves highly valuable in practical applications by converting natural language queries into Cypher [26]. The emergent capabilities of Large Language Models (LLMs) in code and query generation make them an excellent tool for generating accurate Cypher queries [27].

Cypher Query Generation

GPT-4o, developed by OpenAI, is used as a Cypher query generator. Although this model has been trained on limited Cypher-related data, it demonstrates strong in-context learning abilities. Therefore, few-shot prompting techniques are applied, along with the graph schema, to ensure the generated Cypher queries adhere to the required formats [28].

3) Answer Generation

The retrieved information from both vector-based and graph-based retrieval methods is utilized as context, alongside the user's initial query, to synthesize a response with the LLM. For this purpose, GPT-4o-mini has been selected based on its cost-effectiveness and optimized performance, efficiently handling complex, context-rich queries with quick response times..

III. Experiment

A. Experiment Design

The performance of the RAG framework was systematically evaluated using 50 manually crafted seed questions, each aligned with specific Cypher queries and answer templates, derived from high-frequency queries identified in data collection. A set of 1,000 nodes was generated and integrated with these seed questions, from which 200 nodes were randomly selected for processing. Using the GPT-4o-mini model, both questions and answers were rephrased to construct a ground truth dataset, introducing controlled variability. The rephrased data was then processed through the RAG pipeline and evaluated with the RAGAS (Retrieval-Augmented Generation Assessment) framework, providing an automated, reference-free evaluation using large language models to ensure rigorous assessment of the RAG pipeline's efficacy [29]. Three metrics are selected to evaluate RAG pipeline performance including: answer correctness, context precision, and context recall.

Answer Correctness

Answer correctness assesses the accuracy of the generated answer (GA) in comparison to the ground truth (GT) data, which comprises weighted factual information and answer similarity. The language model (LLM) is responsible for identifying True Positives (TP), False Positives (FP), and False Negatives (FN), which are subsequently used to calculate F1 score of factual correctness based on equation (3) [30].

$$F1\ Score\ = \frac{|TP|}{\left(|TP| + 0.5 \times (|TP| + |FN|)\right)} \quad (3)$$

where,
TP is true positive, fact in *GA* and *GT*
FP is false positive, fact in *GA* but not in *GT*
FN is false negative, fact in *GT* but not in *GA*

Context Precision

Precision is the ratio of correctly predicted positive instances to the total predicted positives, indicating the relevance of retrieved results [31]. In the RAGAS framework, each retrieved context is evaluated for relevance to generate a ground truth, with precision@k being calculated for each case. The overall context precision@K is measured by calculating the mean

precision@k, both of which are captured in Equation (4) and (5).

$$Precision@k = \frac{TP@k}{(TP@k+FP@k} \quad (4)$$

$$Context\ Precision@K = \frac{\sum_{k=1}^{K}(Precision@k \times v_k)}{Total\ of\ relevant\ items\ in\ k} \quad (5)$$

where,
k is the rank position being evaluated
TP is true positive, relevant context
FP is false positive, irrelevant context
K is total number of contexts
vk is relevance indicator at rank *k*

Context Recall

Recall refers to one ability to successfully retrieve relevant information [32]. It measures the proportion of relevant cases that have been retrieved out of the total number of relevant cases [33]. Within the RAGAS framework, ground truth (GT) answers are decomposed into individual claim statements, which are then assessed for attribution to the retrieved context (yes/no). This assessment is subsequently quantified using the formula presented in Equation (6).

$$Context\ recall\ = \frac{|GT\ atttributed\ to\ Context|}{|Number\ of\ claims\ in\ GT|} \quad (6)$$

where,
GT is ground truth

B. *Result and Analysis*

The results of this study are summarized in Table I. The graph-based approach demonstrated superior performance, achieving a 21.09% and 19.34% improvement over the vector and hybrid approaches, respectively, in the answer correctness metric. While all methods achieved a context retrieval precision exceeding 70%, the hybrid approach exhibited an impressive context precision of 83.49%. However, its context recall was limited, with a mere 69.09% success in retrieving the necessary context for generating ground truth information. In contrast, the graph-based approach exhibited significantly better context recall, outperforming the other methods.

Table I. Evaluation Comparison

Metrics	Vector	Graph	Hybrid
Answer Correctness	54.67%	**75.76**%	56.42%
Context Precision	73.49%	79.49%	**83.49**%
Context Recall	56.33%	**74**%	69.09%

The graph-based approach excels in context recall and answer correctness due to its structured retrieval, which captures deep semantic and relational information aligned with the query's intent. In contrast, the vector-based method's reliance on surface-level similarity often retrieves irrelevant context, which can mislead the language model and diminish answer quality. This structured advantage in graph-based retrieval ensures that relevant context is prioritized, enhancing both accuracy and contextual completeness in generated responses.

IV. Conclusion and Future Work

This study introduces an approach that combines knowledge graphs with Retrieval-Augmented Generation (RAG) to enhance data traceability within the Master Data Management (MDM) of Company ABC. Results show the graph-based method significantly enhances answer accuracy and context recall, outperforming vector-based and hybrid approaches by 21.09% and 19.34%, respectively, while the hybrid method's lower context recall highlights challenges in combining retrieval techniques. These findings confirm the value of graph-based retrieval for high-accuracy, contextually rich RAG applications.

Limitations and Future Directions

This approach faces scalability and bias issues, requiring improved storage and optimization as data grows. Broader, cross-domain data could enhance generalizability and robustness. Future research should develop context-weighting schemes and advanced prompting to improve model performance, address errors in vector contexts, and enhance reliability. These methods are vital for fields like healthcare, finance, and logistics, requiring tailored schemas and refined context-weighting. Improvements in hybrid retrieval are also needed to overcome context recall limitations and broaden applicability.

Acknowledgment

This research was supported by the Institution of Research and Community Services at Universitas Multimedia Nusantara, with additional contributions from colleagues at the university's Big Data Laboratory and Company ABC, whose collaboration was essential to the study's success.

References

[1] A. Prasad, "Impact of Poor Data Quality on Business Performance: Challenges, Costs, and Solutions," May 27, 2024, *Rochester, NY*: 4843991. doi: 10.2139/ssrn.4843991.

[2] S. Hikmawati, P. I. Santosa, and I. Hidayah, "Improving Data Quality and Data Governance Using Master Data Management: A Review," *IJITEE Int. J. Inf. Technol. Electr. Eng.*, vol. 5, no. 3, Art. no. 3, Sep. 2021, doi: 10.22146/ijitee.66307.

[3] J. Miao, C. Thongprayoon, S. Suppadungsuk, O. A. Garcia Valencia, and W. Cheungpasitporn, "Integrating Retrieval-Augmented Generation with Large Language Models in Nephrology: Advancing Practical Applications," *Medicina (Mex.)*, vol. 60, no. 3, p. 445, Mar. 2024, doi: 10.3390/medicina60030445.

[4] S. Wu *et al.*, "Retrieval-Augmented Generation for Natural Language Processing: A Survey," Jul. 18, 2024, *arXiv*: arXiv:2407.13193. doi: 10.48550/arXiv.2407.13193.

[5] S.-Q. Yan, J.-C. Gu, Y. Zhu, and Z.-H. Ling, "Corrective Retrieval Augmented Generation," Feb. 16, 2024, *arXiv*: arXiv:2401.15884. doi: 10.48550/arXiv.2401.15884.

[6] S. Ma, C. Xu, X. Jiang, M. Li, H. Qu, and J. Guo, "Think-on-Graph 2.0: Deep and Interpretable Large Language Model Reasoning with Knowledge Graph-guided Retrieval," Aug. 06, 2024, *arXiv*: arXiv:2407.10805. doi: 10.48550/arXiv.2407.10805.

[7] C. Peng, F. Xia, M. Naseriparsa, and F. Osborne, "Knowledge Graphs: Opportunities and Challenges,"

Mar. 24, 2023, *arXiv*: arXiv:2303.13948. doi: 10.48550/arXiv.2303.13948.

[8] A. Hogan *et al.*, "Knowledge Graphs," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 1–37, May 2022, doi: 10.1145/3447772.

[9] X. Zou, "A Survey on Application of Knowledge Graph," *J. Phys. Conf. Ser.*, vol. 1487, no. 1, p. 012016, Mar. 2020, doi: 10.1088/1742-6596/1487/1/012016.

[10] Z. Xu *et al.*, "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*, Jul. 2024, pp. 2905–2909. doi: 10.1145/3626772.3661370.

[11] N. Matsumoto *et al.*, "KRAGEN: a knowledge graph-enhanced RAG framework for biomedical problem solving using large language models," *Bioinformatics*, vol. 40, no. 6, p. btae353, Jun. 2024, doi: 10.1093/bioinformatics/btae353.

[12] D. Edge *et al.*, "From Local to Global: A Graph RAG Approach to Query-Focused Summarization," Apr. 24, 2024, *arXiv*: arXiv:2404.16130. doi: 10.48550/arXiv.2404.16130.

[13] D. Sanmartin, "KG-RAG: Bridging the Gap Between Knowledge and Creativity," May 20, 2024, *arXiv*: arXiv:2405.12035. doi: 10.48550/arXiv.2405.12035.

[14] C. Edwards, "Hybrid Context Retrieval Augmented Generation Pipeline: LLM-Augmented Knowledge Graphs and Vector Database for Accreditation Reporting Assistance," May 24, 2024, *arXiv*: arXiv:2405.15436. doi: 10.48550/arXiv.2405.15436.

[15] J. Delile, S. Mukherjee, A. Van Pamel, and L. Zhukov, "Graph-Based Retriever Captures the Long Tail of Biomedical Knowledge," Feb. 19, 2024, *arXiv*: arXiv:2402.12352. doi: 10.48550/arXiv.2402.12352.

[16] N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor, "Industry-scale Knowledge Graphs: Lessons and Challenges: Five diverse technology companies show how it's done," *Queue*, vol. 17, no. 2, p. Pages 20:48-Pages 20:75, Apr. 2019, doi: 10.1145/3329781.3332266.

[17] S. Setty, H. Thakkar, A. Lee, E. Chung, and N. Vidra, "Improving Retrieval for RAG based Question Answering Models on Financial Documents," arXiv.org. Accessed: Sep. 25, 2024. [Online]. Available: https://arxiv.org/abs/2404.07221v2

[18] A. Singh, A. Ehtesham, S. Mahmud, and J.-H. Kim, "Revolutionizing Mental Health Care through LangChain: A Journey with a Large Language Model," arXiv.org. Accessed: Sep. 27, 2024. [Online]. Available: https://arxiv.org/abs/2403.05568v1

[19] X. Wang *et al.*, "Searching for Best Practices in Retrieval-Augmented Generation," Jul. 01, 2024, *arXiv*: arXiv:2407.01219. doi: 10.48550/arXiv.2407.01219.

[20] H. Steck, C. Ekanadham, and N. Kallus, "Is Cosine-Similarity of Embeddings Really About Similarity?," in *Companion Proceedings of the ACM Web Conference 2024*, May 2024, pp. 887–890. doi: 10.1145/3589335.3651526.

[21] S. Ashrafzadeh, G. B. Golding, S. Ilie, and L. Ilie, "Scoring alignments by embedding vector similarity," *Brief. Bioinform.*, vol. 25, no. 3, p. bbae178, May 2024, doi: 10.1093/bib/bbae178.

[22] D. A. Gholap and G. S. V, "An Effective Information Retrieval System Using Keyword Search Technique," *COMPUSOFT Int. J. Adv. Comput. Technol.*, vol. 4, no. 06, Art. no. 06, 2015.

[23] Neo4j, "Full-text indexes - Cypher Manual," Neo4j Graph Data Platform. Accessed: Sep. 27, 2024. [Online]. Available: https://neo4j.com/docs/cypher-manual/5/indexes/semantic-indexes/full-text-indexes/

[24] J. Dong, B. Fatemi, B. Perozzi, L. F. Yang, and A. Tsitsulin, "Don't Forget to Connect! Improving RAG with Graph-based Reranking," 2024, doi: 10.48550/ARXIV.2405.18414.

[25] P. Damodaran, *FlashRank, Lightest and Fastest 2nd Stage Reranker for search pipelines*. (Dec. 2023). Python. doi: 10.5281/zenodo.10426927.

[26] Q.-B.-H. Tran, A. A. Waheed, and S.-T. Chung, "Robust Text-to-Cypher Using Combination of BERT, GraphSAGE, and Transformer (CoBGT) Model," *Appl. Sci.*, vol. 14, no. 17, Art. no. 17, Jan. 2024, doi: 10.3390/app14177881.

[27] W. W. Baraki, *Leveraging large language models for accurate Cypher query generation : Natural language query to Cypher statements*. 2024. Accessed: Sep. 26, 2024. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-24158

[28] Z. Zhong, L. Zhong, Z. Sun, Q. Jin, Z. Qin, and X. Zhang, "SyntheT2C: Generating Synthetic Data for Fine-Tuning Large Language Models on the Text2Cypher Task," Jun. 15, 2024, *arXiv*: arXiv:2406.10710. doi: 10.48550/arXiv.2406.10710.

[29] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated Evaluation of Retrieval Augmented Generation," arXiv.org. Accessed: Sep. 27, 2024. [Online]. Available: https://arxiv.org/abs/2309.15217v1

[30] S. Roychowdhury, S. Soman, H. G. Ranjani, N. Gunda, V. Chhabra, and S. K. Bala, "Evaluation of RAG Metrics for Question Answering in the Telecom Domain," Jul. 15, 2024, *arXiv*: arXiv:2407.12873. doi: 10.48550/arXiv.2407.12873.

[31] N. M. S. Iswari, N. Afriliana, E. M. Dharma, and N. P. W. Yuniari, "Enhancing Aspect-based Sentiment Analysis in Visitor Review using Semantic Similarity," *J. Appl. Data Sci.*, vol. 5, no. 2, Art. no. 2, May 2024, doi: 10.47738/jads.v5i2.249.

[32] R. Taufiq, A. A. Permana, and A. Tiara, "Performance Comparison Of K-Nearest Neighbor And Naive Bayes Algorithms For Heart Disease Prediction," . *Vol.*, no. 21, 2023.

[33] H. Yu, A. Gan, K. Zhang, S. Tong, Q. Liu, and Z. Liu, "Evaluation of Retrieval-Augmented Generation: A Survey," Jul. 03, 2024, *arXiv*: arXiv:2405.07437. doi: 10.48550/arXiv.2405.07437.