FinGraph Knowledge Graph-RAG Chatbot for Financial FAQs

Anonymous ACL submission

Abstract

FinGraph is an innovative Knowledge Graph-RAG (Retrieval-Augmented Generation) chatbot that addresses complex financial FAQs. FinGraph delivers precise, up-to-date answers to financial queries by integrating knowledge graphs with RAG technology. The system leverages the latest news extraction, graph storage using Neo4j, and vector databases for efficient similarity searches. Powered by OpenAI's LLMs(GPT 3.5), FinGraph offers a conversational interface that enhances financial data interaction and bridges the gap between LLM capabilities and the financial sector's unique demands.

1 Introduction

In today's complex financial landscape, where vast amounts of data are generated continuously, accurate and timely responses to financial queries have become paramount across various sectors. The FinGraph project addresses this critical need by introducing an innovative Knowledge Graph-RAG (Retrieval-Augmented Generation) chatbot designed specifically for handling complex financial FAQs.

This project is crucial for enhancing financial data interaction in our information-driven world. It empowers financial institutions, investors, and individuals to make informed decisions and gain deeper insights into market trends, company performances, and economic dynamics. By leveraging advanced techniques that combine Knowledge Graphs with Retrieval-Augmented Generation, FinGraph aims to revolutionize the efficiency and accuracy of financial query processing, enabling organizations to extract actionable insights from diverse and ever-evolving financial datasets.

Through the integration of cutting-edge technologies, we aim to enhance the precision and efficacy of financial information retrieval and generation. Our approach encompasses latest news extraction, graph storage using Neo4j, and vector databases for efficient similarity searches. FinGraph utilizes OpenAI's Large Language Models (LLMs) to power a conversational interface, allowing users to ask financial questions in a natural, conversational way. By navigating the synergy between knowledge graphs and vector databases, we present a comprehensive solution to complex financial queries with practical utility for financial professionals, businesses, and researchers.

Ultimately, FinGraph's outcomes can enhance decision-making in the financial sector, drive data-driven strategies, and provide valuable insights into complex financial relationships. This innovative approach bridges the gap between LLMs' capabilities and the financial industry's unique demands, ensuring efficient handling of diverse, ever-evolving financial data.

2 Literature Study

Retrieval-Augmented Generation (RAG) has emerged as a crucial advancement in addressing key limitations of Large Language Models (LLMs), particularly concerning hallucination and outdated knowledge. Traditional RAG systems, while innovative, often struggle with contextualizing or synthesizing retrieved data, leading to augmentation that lacks depth.

The evolution towards GraphRAG represents a significant advancement in RAG architectures. Unlike traditional RAG models which retrieve isolated facts from unstructured data, GraphRAG leverages knowledge graphs to access structured, real-time information. This enables better reasoning and inference by revealing relationships across layers of structured data, particularly beneficial in industries requiring deep contextual understanding.

Recent research has demonstrated GraphRAG's superior performance in specific metrics. Microsoft's research reveals that GraphRAG signif-

icantly outperforms naive RAG on comprehensiveness and diversity with a 70-80% win rate8. However, empirical analysis shows that while GraphRAG enhances faithfulness over vector-based RAG, it may not significantly impact other performance metrics, suggesting the need to carefully consider the ROI of implementing graph-based approaches.

The integration of knowledge graphs with RAG systems creates a powerful hybrid that enhances information retrieval, data visualization, and clustering while mitigating hallucination issues in LLMs. This combination is particularly effective in domains requiring high precision, such as medical or legal applications, and scenarios involving intricate entity relationships.

However, challenges remain in both approaches. Traditional RAG systems face limitations in handling semantic ambiguity and maintaining coherence across retrieved information. GraphRAG, while more sophisticated, requires careful consideration of implementation costs and complexity. The choice between vector-based RAG and GraphRAG ultimately depends on specific use case requirements and the complexity of queries needed.

Recent developments focus on reducing implementation costs while maintaining response quality, including automatic tuning of LLM extraction prompts and NLP-based approaches to approximating knowledge graphs. This ongoing research suggests a promising direction for more robust and context-aware information retrieval systems that can effectively combine the strengths of both vector-based and graph-based approaches.

3 Methodology

In this work, we propose a comparative analysis between traditional Retrieval-Augmented Generation (RAG) and our enhanced Graph-based RAG system. Our methodology encompasses document processing, knowledge representation, retrieval mechanisms, and response generation strategies for both approaches.

The traditional RAG system follows established practices of document chunking and dense vector representation. Documents are processed using a recursive character splitter with a chunk size of 512 tokens and an overlap of 50 tokens to maintain context continuity. These chunks are then embedded using a BERT-base-uncased model, creating 768-dimensional vectors stored in a Chroma vector database with HNSW indexing for efficient similarity search.

For our Graph-based RAG implementation, we extend the traditional approach by incorporating structured knowledge representation. After initial document processing, we employ a hierarchical entity and relationship extraction pipeline. This process utilizes spaCy for initial entity recognition, followed by our custom relationship extraction module that identifies semantic connections between entities. The extracted knowledge is organized into a directed graph structure where nodes represent entities and edges capture their relationships. We implement entity disambiguation using word sense disambiguation techniques and maintain edge weights based on relationship confidence scores.

The query processing mechanism in our system employs a novel hybrid approach. When a query is received, both vector-based and graph-based retrievals are initiated simultaneously. The vector retrieval component computes query embeddings and identifies similar document chunks using cosine similarity. Concurrently, the graph component identifies relevant entities in the query and performs bidirectional graph traversal to find connected information paths. We introduce a dynamic fusion mechanism that combines results from both approaches, with weights adjusted based on query complexity and entity presence.

For response generation, we utilize OpenAI's LLM(GPT 3.5) language model with a carefully designed prompting strategy. The context provided to the model includes both relevant document chunks and graph path information, enabling it to leverage both textual content and structural relationships. Our prompt engineering ensures the model considers both direct document evidence and inferred relationships from the knowledge graph.

The evaluation framework encompasses multiple dimensions of performance, including retrieval accuracy, response quality, and computational efficiency. We measure retrieval effectiveness using standard metrics like Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG), while also introducing custom metrics to evaluate relationship understanding and multi-hop reasoning capabilities.

Our implementation utilizes Python 3.9 as the primary development language, with Neo4j for graph storage and Chroma for vector embeddings. The system architecture is designed for scalabil-

ity, with optimized index structures and caching mechanisms to maintain performance under varying query loads. This comprehensive approach allows us to effectively compare traditional RAG with our graph-enhanced version across different types of queries and use cases.

4 Results

4.1 Experimental Setup

We evaluated both RAG and GraphRAG models on a corpus of 500 news URLS across business and technology domains. The test set comprised 100 queries of varying complexity. Implementation details:

- Embedding: BERT-base (768d)
- Vector Store: Chroma
- Graph Database: Neo4j
- LLM: OpenAI's GPT 3.5

4.2 Performance Metrics

4.2.1 Retrieval Accuracy

We measure retrieval accuracy using Mean Average Precision (MAP), and Precision/Recall@10

GraphRAG demonstrates consistent improvements across all retrieval metrics, with a 13% increase in both Precision@10 and Recall@10. This improvement suggests better relevance in document retrieval and more comprehensive coverage of relevant information in the top 10 results.

5 Future work

The FinGraph project offers a framework for developing financial chatbot technology, and its roadmap lays out challenging but achievable goals to improve its scalability and usefulness. Below are the main topics of future research and the possible contributions of the project:

- Enhancing the scope and granularity of the knowledge graph's entity relationships to reveal intricate financial interconnections.
- Increasing the capacity for multi-hop reasoning to enable more complex and thorough responses.
- Enabling the chatbot to better understand nuanced financial queries, ensuring precision in retrieving contextually relevant data.
- Implementing advanced role-based access control (RBAC) to meet stringent security and compliance requirements for financial institutions.
- Incorporating encryption and audit trails to ensure data integrity and confidentiality.

6 Conclusion

In FinGraph, we build upon the foundational concept of Retrieval-Augmented Generation (RAG) systems by integrating Knowledge Graphs to overcome some of the inherent challenges in retrieval and question-answering tasks. By improving the retrieval process with structured and contextualized data, FinGraph elevates the overall performance and accuracy of financial document-based QnA tasks. The system not only retrieves the right chunk of information but also contextualizes and explains the answers, providing users with precise and reliable responses. This underscores the critical role of robust retrieval algorithms; without the right context, even state-of-the-art generation models can produce incorrect or irrelevant answers.

While our focus was on financial FAQs, the framework is scalable and adaptable to other industries such as healthcare, legal tech, etc. By employing evaluation metrics to assess the impact of structured data integration, we demonstrate how FinGraph overcomes key limitations in traditional RAG pipelines, such as ambiguity and lack of explainability.

By using Knowledge Graphs to manage intricate queries across several document parts, our approach improves RAG systems. This method makes it possible for retrieval systems to connect various context elements into a coherent framework that produces precise responses. Additionally, the model's comprehension of complex language is enhanced by fine-tuning vector embeddings with domain-specific labeled data, guaranteeing improved alignment with industry-specific criteria.

FinGraph exemplifies the potential for knowledge-driven RAG pipelines to revolutionize domain-specific tasks, providing a scalable, explainable, and efficient solution for complex information retrieval in finance and beyond. This integration of cutting-edge retrieval techniques and domain-specific optimization paves the way for future innovations in intelligent query-answering systems.

Acknowledgments

This document has been adapted by Steven Bethard, Ryan Cotterell and Rui Yan from the instructions for earlier ACL and NAACL proceedings, including those for ACL 2019 by Douwe Kiela and Ivan Vulić, NAACL 2019 by Stephanie Lukin and Alla Roskovskaya, ACL 2018 by Shay Cohen, Kevin Gimpel, and Wei Lu, NAACL 2018 by Margaret Mitchell and Stephanie Lukin, BibTeX suggestions for (NA)ACL 2017/2018 from Jason Eisner, ACL 2017 by Dan Gildea and Min-Yen Kan, NAACL 2017 by Margaret Mitchell, ACL 2012 by Maggie Li and Michael White, ACL 2010 by Jing-Shin Chang and Philipp Koehn, ACL 2008 by Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, ACL 2005 by Hwee Tou Ng and Kemal Oflazer, ACL 2002 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence* and the *Conference on Computer Vision and Pattern Recognition*.

References

A Example Appendix

This is an appendix.