Enhancing Retrieval with LLM-Driven Knowledge Graphs

Brian Mann bm3772@nyu.edu
Jaulie Goe jg5059@nyu.edu
Rebecca Glick rmg9724@nyu.edu
Ryan Ghayour rpg8343@nyu.edu

1 Introduction

1.1 Knowledge Graph Retrieval Augmented Generation

A knowledge graph (KG) is a structured representation of named entities (e.g., people, places, organizations, concepts) and their relationships, forming a graph-based knowledge structure for organizing and retrieving data. Unlike relational databases that store data in a tabular form, KGs model relationships explicitly, allowing for richer contextual understanding and pathways between related concepts (Hogan et al., 2020).

KGs have recently gained in popularity as a key component of the Retrieval-Augmented Generation (RAG) architecture, an AI framework that uses information retrieval from external documents to generate responses. RAG was first introduced in 2020 to tackle limitations of large pre-trained language models, particularly precise knowledge retrieval and knowledge updates without re-training. RAG was proposed as a hybrid approach that combines parametric memory (stored within the model's parameters) and non-parametric memory (retrieved from external sources), allowing models to dynamically access relevant information from a database instead of relying solely on their pre-trained knowledge (Lewis et al., 2020).

To enhance retrieval efficiency, vector-based models became essential components of modern RAG implementations. A RAG vector database uses vector retrieval to locate relevant data for the large language model (LLM) to process by breaking data into smaller vector embeddings and then matching a query with the closest vectors using common algorithms like K-Nearest Neighbors or Approximate Nearest Neighbors (Berry et al., 1999). This approach, however, does not always capture the nuances or relationships between data points, which can make it difficult to handle complex data interactions.

For this reason, instead of vector databases, researchers have started integrating KGs into RAG systems to further improve knowledge retrieval, reasoning, disambiguation, and factual consistency. While traditional RAG models retrieve unstructured data, KG-based RAG instead retrieves structured knowledge from graph databases. Integration of KGs into the RAG framework improves relevant information retrieval performance, especially in domain-specific use cases (Zhong et al., 2023; Pan et al., 2024).

1.2 Automatic Knowledge Graph Construction (KGC)

Constructing a knowledge graph from a corpus is a complex process that requires structuring information into meaningful relationships. Knowledge graphs are often built from scratch by domain experts, derived from unstructured or semi-structured data, or integrated from existing knowledge graphs. This process typically involves entity extraction, relationship identification, and ontology development, which can require extensive human annotation and demand specialized expertise (Zhong et al., 2023).

Previous research has focused on building individual systems for these information extraction (IE) sub-tasks. When multiple tasks need to be performed together, as with KGC, it becomes complicated to combine these various specialized architectures together (Lu et al., 2022).

Recent advancements in LLMs have made it possible to not only automate the process of KGC, but also to build sequence-to-sequence systems that perform all sub-tasks necessary to construct KGs. Universal Information Extraction System is an early example that demonstrated the feasibility of this approach (Lu et al., 2022).

Early KGC systems fine-tuned pre-trained language models on information extraction sub-tasks, breaking the overall challenge into parts such as entity recognition, co-reference resolution, and relation extraction, then combining those sub-tasks into an overall system (Wang et al., 2020; Melnyk et al., 2022; Dognin et al., 2021).

1.3 KGC with Prompt Engineering

As pre-trained language models have increased in size, so has their potential for KGC through zero- and few-shot prompting. Compared to smaller pre-trained models that required task-specific fine-tuning, current state-of-the-art LLMs demonstrate remarkable latent capabilities for performing novel tasks. Prompting is a reliable alternative for fine-tuning for generation (Qian et al., 2022), but LLMs are sensitive to prompt engineering (Wei et al., 2023). Prompt-tuning has been used for upstream tasks, like relation extraction (Chen et al., 2021) and event detection (Wang et al., 2022).

Prompting-based methods have been used for the task of sequence-to-sequence KGC (Zhang and Soh, 2024; Zhu et al., 2024) and for the related task of relational triple extraction (Ding et al., 2024). For instance, (Carta et al., 2023) introduced a pipeline leveraging GPT-3.5 to construct knowledge graphs through iterative zero-shot prompting. Each prompt operates without examples, allowing the LLM to autonomously extract relevant entities and relationships from unstructured text with no manual intervention. The pipeline systematically identifies entities, assigns types and descriptions, extracts relationships, resolves duplicates, and infers a schema—ensuring a structured and coherent knowledge graph. Another group of researchers introduced Iter-RetGen, a method that iteratively combines retrieval and generation processes by alternating between these phases. A given response is used to retrieve more relevant information, which informs the next iteration of generation, progressively refining the model's output. This effectively narrows down the search space, leading to more pertinent information being fetched in each iteration (Shao et al., 2023).

1.4 Evaluation Approaches

Previous work has often used information extraction benchmarks to evaluate systems' abilities to perform various IE subtasks in the KGC pipeline (Lu et al., 2022; Zhang and Soh, 2024; Zhu et al., 2024)

Other approaches have evaluated the entire KG-RAG pipeline as a whole on various downstream tasks, such as summarization (Edge et al., 2024), reasoning (Luo et al., 2024), and question-answering (Guo et al., 2024; Huang et al., 2025).

2 Approach

2.1 Novel Component

There have been uses of iteratively prompting an LLM to build knowledge graphs. These have had a variety of approaches to model evaluation, such as comparing steps of the pipeline to baseline models for specific tasks (Zhang and Soh, 2024), employing human assessors to evaluate the quality of retrieved documents (Carta et al., 2023), or evaluating on reasoning benchmark datasets as opposed to retrieval benchmark datasets (Wei et al., 2023). As it stands, there has not been research focused specifically on evaluating the document retrieval capabilities of a RAG model using a KG constructed by iteratively prompting an LLM, and evaluating those capabilities on a related ground-truth QA dataset.

2.2 Hypothesis

We propose that through prompt-tuning, chain-of-thought prompting, and iterative prompting, we can use LLMs to automatically generate richer knowledge graphs that better aid a RAG system in information retrieval than systems using a single-prompt generated knowledge graph or systems using vector databases. Performance will be measured by question-answering accuracy on the PubMed database abstracts.

3 Experiment Design

3.1 RAG Pipeline

Our experimental design for knowledge graph creation will involve testing systems that automatically generate knowledge graphs using various prompt-engineering protocols. Llama 3.2 will be our base model for KGC.

Resulting KGs will be incorporated into a generic RAG pipeline built with LangChain, a framework for building applications with integrated LLMs. The pipeline will include a retrieval system that employs an LLM to convert the given question into a Cypher query, which will be used to index the KG, stored as a graph database. We will be using the graph database management system Neo4j to host our KGs.

All KGs will be evaluated in the same pipeline with the same QA test set. Our chosen QA test set is PubMedQA, which builds on top of the PubMed database, adding question/answer pairs, as well as a gold-standard retrieval passage (Jin et al., 2019). PubMedQA is a domain-specific dataset of medical paper abstracts. Domain specific contexts are a key use case for KG-RAG, as state-of-the-art LLMs often struggle with insufficient domain knowledge (Ling et al., 2024). Domain-specific fine-tuning and pre-training are cost prohibitive, which make knowledge graphs a practical alternative in these scenarios.

The purpose of evaluating KGs in the context of a RAG pipeline is to mimic how knowledge graphs will be used in an industry setting. We would like to evaluate the final KGs on their ability to aid in answer generation rather than in their ability to effectively perform some sub-task, such as entity recognition. We will be, in effect, conducting an ablation study, testing a generic RAG pipeline with and without the inclusion of a KG, and with KGs constructed used various prompting protocols.

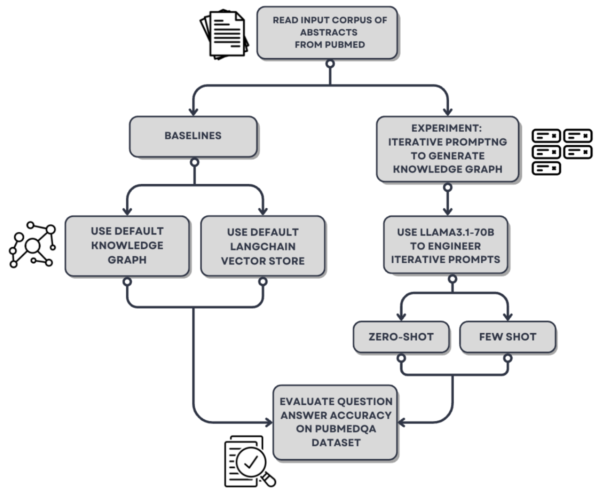

READ INPUT CORPUS OF ABSTRACTS FROM PUBMED
BASELINES
EXPERIMENT: ITERATIVE PROMPTNG TO GENERATE KNOWLEDGE GRAPH
USE DEFAULT KNOWLEDGE GRAPH
USE DEFAULT LANGCHAIN VECTOR STORE
USE LLAMA3.1-70B TO ENGINEER ITERATIVE PROMPTS
ZERO-SHOT
FEW SHOT
EVALUATE QUESTION ANSWER ACCURACY ON PUBMEDQA DATASET

Figure 1: Experiment Overview

3.2 Prompt Engineering

We will experiment with iterative refinement of LLM prompts, testing different strategies for extracting more granular relationships and improving the quality of the knowledge graph. This method of determining the optimal prompt will involve multi-step iterative prompting, when an initial extraction pass is refined through subsequent interactions. We will experiment with zero-shot prompting, with no examples to aid the model, as well as few-shot prompting, where relevant examples guide the model's relationship extraction process. In the context of medical questioning, the examples used in few-shot prompting would be specific to how the dataset expects the model to select data.

3.3 Dataset-Specific Decisions

In the PubMedQA dataset, there are four question types and three reasoning types (Jin et al., 2019). Furthermore, each point in the labeled section of the dataset contains important context to pull from relevant documents, and each of the important keywords relevant to the question. We plan to refine a prompting method that pushes the model to break down the questions into components based on the given question structure guide, and finds the correct answer to the question using a combination of context sentences and keywords as indicated in the dataset. In our few-shot approach, we will use examples from the labeled dataset to give our model examples of using context in this intended way to properly answer the question. We foresee our model using these predefined context sentence/question structure relationships to form useful and efficient nodes and edges for the purpose of answering medical questions. Our experimentation plan is to start with a set of prompting strategies and continuously improve upon the most successful subset of those prompts until progress levels off. The ultimate goal is to determine how structured knowledge representations impact retrieval quality, accuracy, and overall performance in biomedical tasks.

3.4 Baselines

In order to determine the relative impact of our knowledge graphs on the RAG model, we will evaluate performance against two primary baselines.

The first baseline will be a RAG model that utilizes one of the default LangChain vector stores. We plan to use PGVector, Chroma, Pinecone, FAISS, or Lance as the vector store for this baseline. After some preliminary research, these are the most popular and effective options, and we will use the performance of the most effective one as a baseline, as we do not want to "cherry-pick" a low accuracy, making it easier to beat. We are choosing this as a baseline because we want to test how effective knowledge graphs are in a RAG pipeline when compared to vector databases.

The second baseline will be a RAG model that utilizes the default knowledge graph provided by LangChain. This popular knowledge graph option is generated by a simple zero-shot prompt. Therefore, the accuracy of the resulting RAG model will serve as a great metric for the effectiveness of our prompt engineering. Theoretically, we will be able to quantify the impact of each prompting decision we make with this baseline as a comparison.

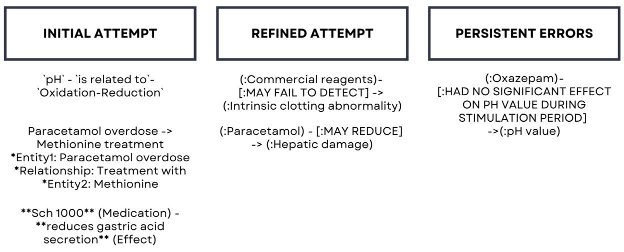
3.5 Evaluation

As stated previously, we will be evaluating our model on its effectiveness on the PubMedQA dataset. In practice, this will involve asking a question from the QA dataset (an unused one in the case of few-shot prompting) and restricting the model to answering yes, no, or maybe. Then, we will evaluate the accuracy of the provided answers, using that accuracy as our evaluation metric. When we iteratively prompt, we will partition a test set of questions into smaller subsets and then alternate between making changes to the KGC prompt and evaluating the resulting model on one of the subsets. The dataset contains a set of expert-labeled QA instances, a larger set unlabeled ones, and a much larger set of AI-generated QA instances. We will be using the labeled instances for for both few-shot prompting and evaluation, and will default to AI generated instances if we require more than the labeled questions for evaluation. Evaluation requires questions labeled with correct answers, so the unlabeled set will remain unused for this project.

4 Early Experiments

Our initial experimentation with Llama 3.2 to create a KG has helped identify several potential obstacles. The PubMed dataset consists of 1.3k XML files that each contain 15k abstracts. Prompting Llama 3.2 with the first XML file and asking it to find tuples to produce a knowledge graph only produced 9 tuples. Tweaking the prompt allowed for increased output, but most of the abstracts were unused even after various attempts at prompting the model to utilize all of the abstracts. We will continue to experiment to increase the quantity of tuples found, but based on our initial experiments, it seems that a preferable approach will be to split the abstracts into smaller batches and prompt the model to find tuples in each of these batches.

If we use smaller batches, it will be essential that the results are consistent in format so that they can be aggregated. We experimented with running the prompt on 10 abstracts at a time, but despite specifying the desired tuple format, each initial batch produced different output structures. We were able to adjust the prompt to get a consistent format for the results, but more work is required to ensure the content within that structure is reliable.



```
`pH` - `is related to`-
`Oxidation-Reduction`

Paracetamol overdose ->
Methionine treatment
*Entity1: Paracetamol overdose
*Relationship: Treatment with
*Entity2: Methionine

**Sch 1000** (Medication) -
**reduces gastric acid
secretion** (Effect)
```

```
(:Commercial reagents)-
[:MAY FAIL TO DETECT] ->
(:Intrinsic clotting abnormality)

(:Paracetamol) - [:MAY REDUCE]
-> (:Hepatic damage)
```

```
(:Oxazepam)-
[:HAD NO SIGNIFICANT EFFECT
ON PH VALUE DURING
STIMULATION PERIOD]
->(:pH value)
```

Figure 2: Early Experimental Results

Batch processing could potentially introduce other problems as well, such as duplicates or conflicting tuples. We plan to do further experimentation with a second Llama 3.2 pass to resolve inconsistencies once all tuples have been extracted.

We remain optimistic that further prompt engineering will enable Llama 3.2 to extract meaningful relationships. If we cannot overcome these limitations, we will explore alternative models with improved consistency and output volume. If the volume of abstracts proves too burdensome, we will limit the project to a smaller subset of the abstracts and select questions from PubMedQA that can be answered from this subset of abstracts. Based on early experimentation, we believe that this project is feasible given the available time and resources.

5 Division of Labor

We plan to work in parallel on all aspects of the project while assigning primary responsibilities to ensure steady progress towards completion. Jaulie and Rebecca will take the lead on investigating and setting up Neo4j and integrating Llama 3.2 into our workflow. Brian will focus on reading and processing documents from the PubMed database, which will serve as the source for the PubMedQA dataset retrieval. Ryan will be responsible for prompt engineering, optimizing queries and responses. To maintain alignment, we will meet weekly for 1-2 hours to review progress, troubleshoot challenges, and ensure everyone can reproduce the same results. For the final paper, we will divide the sections among all team members, writing them individually before collaboratively editing and refining the document at the end. This structured approach will help us efficiently complete the project while supporting each other throughout the process.

References

Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. 1999. Matrices, vector spaces, and information retrieval. *SIAM Review*, 41(2):335–362.

Salvatore Carta, Alessandro Giuliani, Leonardo Piano, Alessandro Sebastian Podda, Livio Pompianu, and Sandro Gabriele Tiddia. 2023. Iterative Zero-Shot LLM Prompting for Knowledge Graph Construction. ArXiv:2307.01128 [cs].

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. *CoRR*, abs/2104.07650.

Zepeng Ding, Wenhao Huang, Jiaqing Liang, Deqing Yang, and Yanghua Xiao. 2024. Improving Recall of Large Language Models: A Model Collaboration Approach for Relational Triple Extraction. ArXiv:2404.09593 [cs] version: 1.

Pierre Dognin, Inkit Padhi, Igor Melnyk, and Payel Das. 2021. ReGen: Reinforcement learning for text and knowledge base generation using pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1084–1099, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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].

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. LightRAG: Simple and Fast Retrieval-Augmented Generation. ArXiv:2410.05779 [cs].

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. 2020. Knowledge graphs. *CoRR*, abs/2003.02320.

Manzong Huang, Chenyang Bu, Yi He, and Xindong Wu. 2025. How to Mitigate Information Loss in Knowledge Graphs for GraphRAG: Leveraging Triple Context Restoration and Query-Driven Feedback. ArXiv:2501.15378 [cs].

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *CoRR*, abs/1909.06146.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. *CoRR*, abs/2005.11401.

Chen Ling, Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Yun Li, Hejie Cui, Xuchao Zhang, Tianjiao Zhao, Amit Panalkar, Dhagash Mehta, Stefano Pasquali, Wei Cheng, Haoyu Wang, Yanchi Liu, Zhengzhang Chen, Haifeng Chen, Chris White, Quanquan Gu, Jian Pei, Carl Yang, and Liang Zhao. 2024. Domain specialization as the key to make large language models disruptive: A comprehensive survey.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

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].

Igor Melnyk, Pierre Dognin, and Payel Das. 2022. Knowledge Graph Generation From Text. ArXiv:2211.10511 [cs].

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599. ArXiv:2306.08302 [cs].

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, Dublin, Ireland. Association for Computational Linguistics.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy.

Chenguang Wang, Xiao Liu, and Dawn Song. 2020. Language models are open knowledge graphs. *CoRR*, abs/2010.11967.

Sijia Wang, Mo Yu, and Lifu Huang. 2022. The Art of Prompting: Event Detection based on Type Specific Prompts. ArXiv:2204.07241 [cs].

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. ArXiv:2201.11903 [cs].

Bowen Zhang and Harold Soh. 2024. Extract, Define, Canonicalize: An LLM-based Framework for Knowledge Graph Construction. ArXiv:2404.03868 [cs].

Lingfeng Zhong, Jia Wu, Qian Li, Hao Peng, and Xindong Wu. 2023. A Comprehensive Survey on Automatic Knowledge Graph Construction. *ACM Comput. Surv.*, 56(4):94:1–94:62.

Yuqi Zhu, Xiaohan Wang, Jing Chen, Shuofei Qiao, Yixin Ou, Yunzhi Yao, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2024. LLMs for Knowledge Graph Construction and Reasoning: Recent Capabilities and Future Opportunities. ArXiv:2305.13168 [cs].