Title:

Enhancing Large Language Model Reasoning with Dynamic Knowledge Graph Retrieval

Research Problem:

Large Language Models (LLMs) acquire knowledge through a pre-training process on unstructured text. While this method excels at capturing linguistic patterns and general concepts,the knowledge gained is static, implicit, and non-verifiable which leads to factual inaccuracies, poor reasoning over complex topics, and frequent hallucinations. The model cannot distinguish between a confident guess and a verifiable fact from a trusted source.

Research Goal:

This project aims to design and evaluate a dynamic, on-demand graph-structured retrieval framework that addresses the limitations of pre-trained LLM knowledge. Instead of relying on a pre-existing, static knowledge graph, the system will construct a temporary, query-specific graph from a given context. This approach complements the LLM's pre-trained knowledge with explicit, verifiable facts, transforming its role from a pattern-matcher into a sophisticated reasoner grounded in a clear, structured knowledge base.

Research Question

Can a dynamic, query-specific graph construction framework significantly improve the factual accuracy, reasoning, and consistency of LLM responses across multiple domains, specifically for high-stakes use cases like medical decision support and personal knowledge management?

Literature Review

This research is informed by a variety of academic and industry work in the fields of Retrieval-Augmented Generation (RAG), Knowledge Graph construction, and LLM reasoning. The following papers provide key background and inspiration for this project:

- Query-Driven Multimodal GraphRAG: Dynamic Local Knowledge Graph Construction for Online Reasoning - ACL Anthology
- [2412.13782] Knowledge Editing with Dynamic Knowledge Graphs for Multi-Hop Question Answering
- Large Language Models-guided Dynamic Adaptation for Temporal Knowledge Graph Reasoning
- MedRAG: Enhancing Retrieval-augmented Generation with Knowledge Graph-Elicited Reasoning for Healthcare Copilot | Proceedings of the ACM on Web Conference 2025
- Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting
- Identifying Query-Relevant Neurons in Large Language Models for Long-Form Texts | Proceedings of the AAAI Conference on Artificial Intelligence

- Unleashing the Potential of Large Language Models as Prompt Optimizers: Analogical Analysis with Gradient-based Model Optimizers | Proceedings of the AAAI Conference on Artificial Intelligence
- Improving Retrieval Augmented Language Model with Self-Reasoning | Proceedings of the AAAI Conference on Artificial Intelligence
- Harnessing Large Language Models for Knowledge Graph Question Answering via Adaptive Multi-Aspect Retrieval-Augmentation | Proceedings of the AAAI Conference on Artificial Intelligence

Proposed Methodology

The proposed framework consists of four core components:

- Graph Builder: This system will transform raw, unstructured datasets into a temporary, query specific graph by extracting key entities and the explicit relationships between them (nodes = entities, edges = relationships).
- Retriever: This component will utilize the constructed graph to filter and prioritize the most relevant and connected information in relation to the user query, and enable multi hop reasoning.
- LLM Integration: The retrieved graph data is formatted, and injected into the LLMs prompt. It provides a clear “fact sheet” which will help ground the LLMs response and prevent it from relying on its internal implicit knowledge for the final output
- Evaluation: the output from graph-enhanced prompting will be compared to the LLMs baseline (using the LLm without graph retrieval) to measure the reduction in hallucination, improvement in factual accuracy, and better reasoning over complex contexts.

Ethical Considerations

Given the sensitive nature of the data involved in the patient history and personal assistant use cases, ensuring that data is ethically processed is paramount.

- **Privacy and Data Security:**For the medical domain, I will use only de-identified datasets such as MIMIC-IV to prevent the re-identification of patients. No Protected Health Information (PHI) will be handled. The entire data processing pipeline, including graph construction and retrieval, will be conducted in a secure, local or sandboxed environment. For the personal assistant domain, my own private data will be used for a proof-of-concept. The data will not be shared, and all processing will be done locally to prevent any unauthorized access or data leaks.
- **Informed Consent:** While using de-identified data from a public dataset like MIMIC-IV, I will adhere to all data use agreements and institutional policies. The framework will be designed for an opt-in application where a user explicitly grants the system access. The project will assume that in a real-world scenario, clear and informed consent would be obtained.
- **Mitigation of Bias:** The framework helps mitigate bias by grounding the LLM's response in a specific, verifiable context, rather than its generalized training data. The evaluation phase will include an analysis of fairness and consistency to identify any unintended

biases. The system is designed as an assistive tool, and the ultimate responsibility for any medical or personal decision remains with the human user.

Project Phases and Timeline

- **Phase 1 – Research Design & Dataset Selection:** (1 week) Finalize domains, select datasets, and write the proposal summary.
- **Phase 2 – Graph Construction:** (2–3 weeks) Convert raw data into graph format by preprocessing and extracting entities/relationships.
- **Phase 3 – Graph-Based Retrieval:** (2-3 weeks) Build logic to select relevant context from the graph and format it for the LLM.
- **Phase 4 – LLM Integration:** (2-3 weeks) Connect to LLM APIs and test with and without graph-enhanced prompts.
- **Phase 5 – Evaluation:** (1–2 weeks) Define criteria, run tests, and analyze results.
- **Phase 6 – Write-Up & Results:** (2 weeks) Document findings, prepare slides, and write a final paper or blog post.

Tools & Resources

- Python, Jupyter Notebooks
- networkx, spaCy, sentence-transformers, pandas
- Gemini API, Hugging Face Inference API, OpenRouter API
- Google Colab or local dev environment
- Dataset: https://physionet.org/content/mimiciv/3.1/