BA 890 Research Paper

Title: Comparing effectiveness of Retrieval-Augmented Generation in financial querying

Objective: To test the effectiveness of two cutting edge LLM RAG methods in improving the quality of answers to financial queries. The goal is to determine the best way to integrate pretrained LLMs with **novel or proprietary data**. Our goal is to devise a proof-of-concept chatbot system that can accurately answer questions about any SEC filings that it has been given access to. This has the potential to automate a task that costs firms billions every year.[1]

Dataset: Our dataset consists of a 10-Q quarterly SEC filing made by Apple in Q3 of 2022. The 28-page report is accompanied by a set of related Q&As. This dataset has been compiled by Docugami for the purposes of benchmarking LLMs in document retrieval tasks.

- **Data Source**: https://github.com/docugami/KG-RAG-datasets/tree/main/sec-10-q
- **LLMs:** We will be using the ***Llama 3.1 8B*** model during this experiment as our primary LLM owing to its open-source nature, high-performance, small size and 2023 cutoff date. We will also use OpenAI's ***text-embedding-3-small model*** for our RAG application.

Proposed Methodologies: For testing, we have three main approaches

1. **Standard LLM (no modification):** Our Llama model has a knowledge cutoff of March, 2023. This means that the model has potentially been trained on data relevant to our experiment and may be able to answer questions related to these filings in a zero-shot environment. However, we expect results to be poor due to a lack of relevant context.
2. **Retrieval-Augmented Generation (RAG):** This technique allows us to provide our LLMs with direct access to a vector database consisting of embeddings of the SEC report's content to use as a source of ground truth. My hypothesis is that the additional context provided by these embeddings will improve the accuracy of answers.
3. **Self-RAG w/ Reflection[7]:** In this method, I employ a multi-step RAG methodology. The primary addition in this method is a self-reflection stage where the LLM evaluates the quality of the context retrieved using the RAG vector database. If it determines the retrieved context is not relevant to the question, it will attempt to rephrase the question to improve the quality of content retrieved. The idea is to improve performance through this self-grading and prompt modification mechanism.

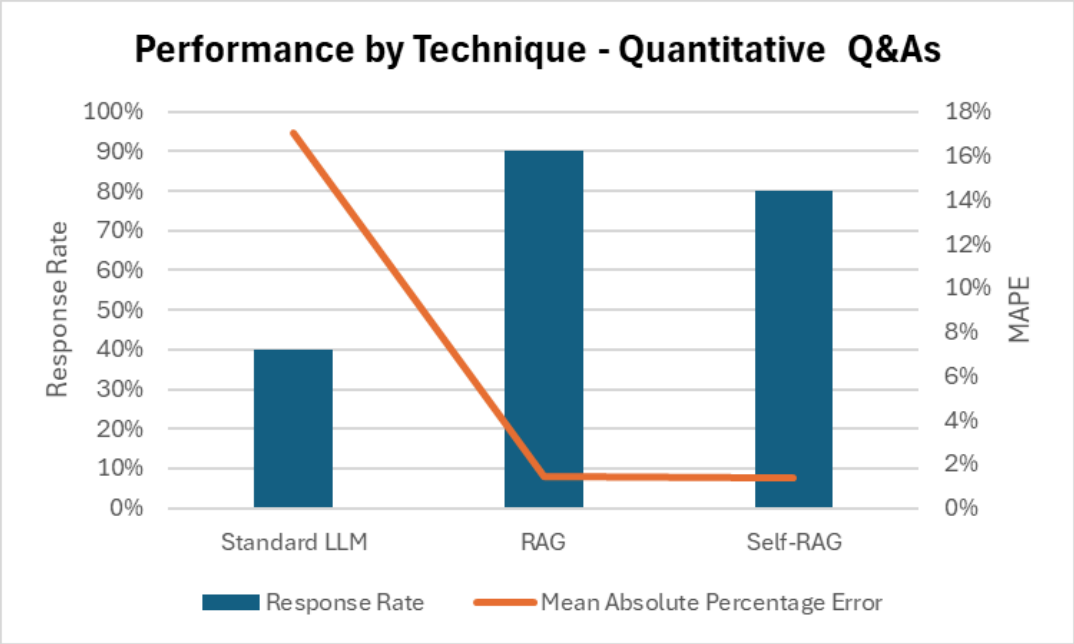
Experiment Design

- **Questions**: 20 Questions were used for this experiment (some from Docugami's question set and some created by myself). Half are quantitative (requiring a numerical answer) and half are qualitative (requiring a logical text-based answer).
- **Metrics:**
  - **Quantitative Qs**[2]:
    - Response Rate: % of responses w/ a numerical answer
    - Absolute percentage error: (Benchmark-Answer)/Benchmark
  - **Qualitative Qs**:
    - Quality of Answer (High/Complete | Medium/Partial | Low/Incomplete
    - Benchmark consists of a few critical sentences I believe reasonably answer the question. The benchmarks are created to be very general/basic.
- All tests were run locally using Ollama and LangChain in a Jupyter environment.[4][5]

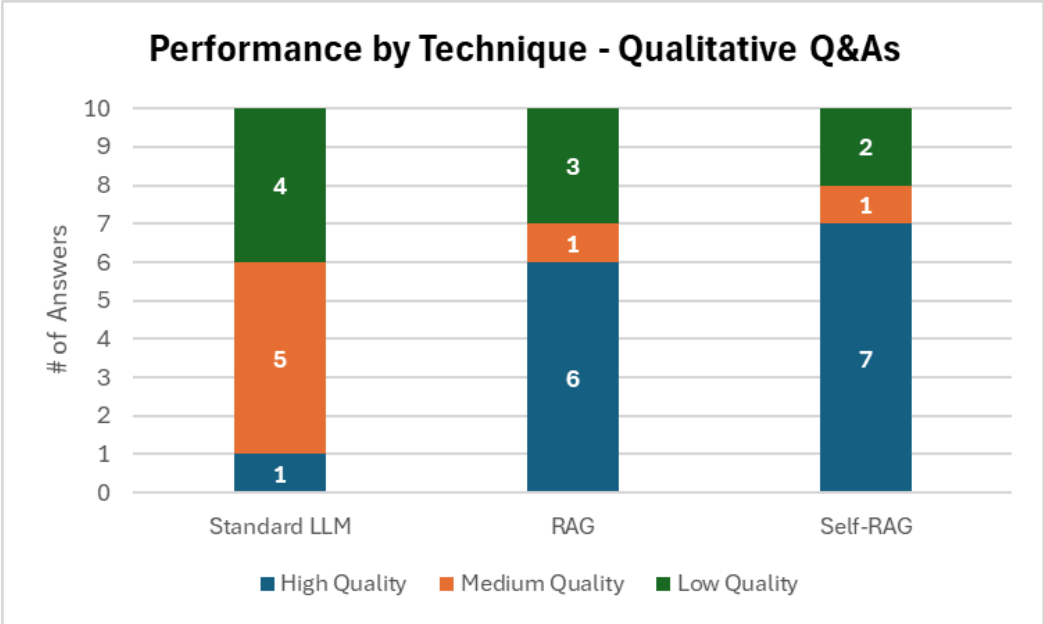
BA 890 Research Paper

Results

Links to Code and Results[3][6]: https://github.com/d-roho/FIN-RAG



Critical Summary: Both our RAG methods vastly outperform a zero-shot approach both in terms of willingness to answer the question and the accuracy of the numerical answer given. The Self-RAG is slightly more hesitant to answer questions compared to RAG, possibly due to the in-built mechanisms intended to prevent the system from providing poor quality answers.



Critical Summary: Again, our augmented methods outperform the standard LLM. Here, we can see how the Self-RAG's unique performance enhancing techniques help it outperform the standard RAG. Also, from a human perspective I felt the quality and depth of Self-RAG's answers were tangibly higher compared to the standard RAG system.

BA 890 Research Paper

Future Work:

- **Tuning:** I only conducted basic tuning using the standard workflow provided by LangGraph for Self-RAG. Further tuning and design modifications can help improve quality and consistency.
- **Multi-Document Analysis:** The next logical step in terms of scale would be to test both methods with a larger dataset with multiple companies' SEC filings to compare and contrast companies' performance.
- **Trends across time:** Another way to scale is temporally. In theory, it is possible to vectorize multiple years of financial reports for a single company and make queries about historical trends and forecasts.

Conclusions and Recommendations:

- RAG is the current gold standard for augmenting LLMs with novel data and context. It has proven to be reliable in our financial querying use case as well, in both needle-in-a-haystack (quantitative) and logical reasoning (qualitative) tasks.
- Self-RAG in general provides higher quality answers, and is more fool-proof in the sense that it refuses to mislead the user. However, in the initial stages workflow had several kinks that led it to fail in answering most of the questions. Self-RAG required significantly more tuning and testing than the standard RAG
- I would recommend using RAG as a prototyping tool and for simpler use cases, while saving self-RAG for deployment and/or more complicated tasks.

References:

[1] Freedman, R. (2022, January 13). Bad use of FP&A costs U.S. companies $7.8B a year. CFO Dive. Retrieved August 10, 2024, from https://www.cfodive.com/news/bad-use-of-fpa-costs-us-companies-78b-a-year/617159/
[2] Riedl, E., & Korganbekova, A. (2024, April 10). LLMS for Basic Financial Data Queries - Eddie Riedl & Aliya Korganbekova (Data Blitz 2024). *YouTube*. https://www.youtube.com/watch?v=Y7L1A85iUyI
[3] Blanco, S. P. (n.d.). *RAG_local_tutorial*GitHub. Retrieved August 10, 2024, from https://github.com/sergiopaniego/RAG_local_tutorial
[4] https://ollama.com/
[5] https://python.langchain.com/
[6]LangChain. (n.d.). *langgraph/examples/rag/langgraph_self_rag.ipynb at main · langchain-ai/langgraph*. GitHub. Retrieved August 10, 2024, from https://github.com/langchain-ai/langgraph/blob/main/examples/rag/langgraph_self_rag.ipynb
[7] Asai, A. et al . *Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection*. arXiv. Retrieved August 10, 2024, from https://arxiv.org/abs/2310.11511