

Multi-Agent QA on Financial Documents

Prepathon 2025 NLP Project

Saksham Madan
IIT BHU (Varanasi)
Mathematics and Computing, Part II

October 7, 2025

Abstract

This project implements a multi-agent question answering (QA) system over financial documents. The system consists of two tasks: (i) retrieval of financial report content using Jina embeddings with FAISS indexing, and (ii) dynamic orchestration of multiple specialized agents under the supervision of a Groq LLM. This report documents the architecture, implementation, usage, and deliverables.

1 Introduction

Financial documents contain structured and unstructured information, including text, tables, and numerical data. Extracting insights requires both retrieval and reasoning. This project explores a multi-agent pipeline where specialized agents collaborate to answer user queries.

2 Project Tasks

2.1 Task 1: Document Retriever

- Convert PDF \rightarrow Images
- Embed images using **Jina Embeddings**
- Build a **FAISS index** for retrieval
- Retrieve relevant pages and parse them
- Apply RAPTOR-style clustering

2.2 Task 2: Multi-Agent QA

- Agents: Retriever, Table, Math, Summarizer
- **Groq LLM** acts as supervisor

- Dynamically selects next agent
- Produces final answer with reasoning trace

3 Project Structure

```
.
retriever.ipynb      # Task 1: Retrieval pipeline
multi_agent.ipynb    # Task 2: Multi-agent orchestration
data/                # Embeddings, indexes, memory
outputs/             # JSON traces
requirements.txt      # Dependencies
.gitignore
report.pdf
README.md
```

4 Installation

4.1 Clone Repo & Install Dependencies

```
git clone https://github.com/s4kr3d-w0r1d/FinanceRAG.git
cd FinanceRAG
pip install -r requirements.txt
```

4.2 Environment Variables

```
export JINA_API_KEY="your_jina_key_here"
export GROQ_API_KEY="your_groq_key_here"
```

5 Usage

5.1 Task 1: Retriever

1. Convert PDF to images
2. Embed with Jina
3. Build FAISS index
4. Retrieve pages
5. Parse + Cluster
6. Answer query

5.2 Task 2: Multi-Agent QA

1. Initialize agents
2. Supervisor decides next step
3. Execute query (e.g. YoY growth, risk analysis)
4. Output: final human-readable answer + JSON trace

6 Example Query

```
query = "Compare the YoY revenue growth and R&D spending  
        between 2021 and 2022, and summarize the risks  
        affecting future revenue."  
result = system.run(query)  
print(result["final_answer"])
```

7 Deliverables

- Working code in both notebooks
- JSON logs of agent traces in `/outputs/`
- Report including architecture, sample runs, and evaluation

8 Notes

- Table extraction is heuristic and depends on PDF structure.
- Groq is used as supervisor and summarizer.
- Jina embeddings handle page-level retrieval.

9 Conclusion

The project demonstrates a practical application of retrieval-augmented generation and multi-agent orchestration for financial document analysis. By combining embedding-based retrieval and reasoning with dynamic agent control, the system provides interpretable, structured answers to complex queries.