FinRAG: A Retrieval-Augmented Generation System for Stock Market Insights and Investment Strategies

Natural Language Processing

Instructor : Dr. Shivanjali Khare

Presented By:

Abhigya Malla
Kalyan Khatry
Suraj Thapa

11/30/2025

Introduction & Motivation

Problem Statement

- Financial information is scattered across multiple sources
- Need for accurate, context-aware financial advice
- Challenge: How to leverage multiple data sources effectively?

Solution: FinRAG

- Combines retrieval-based and generation-based approaches
- Uses multiple state-of-the-art LLMs
- Provides grounded, contextual financial insights

Data Sources Overview

Three Primary Data Sources

1. Reddit Financial Communities
   1. 6 subreddits: r/AskEconomics, r/Economics, r/investing, r/StockMarket, r/stocks, r/wallstreetbets
   2. Community discussions, sentiment, real-world experiences
2. Stock Market Data
   1. Top 10 stocks: AMZN, MSFT, NVDA, AVGO, ERIE, GOOGL, META, NOW, PYPL, CMG
   2. Historical price data (1 year)
   3. Daily metrics: Open, High, Low, Close, Volume
3. Financial Literature
   1. Books (13)
   2. Research Papers (8)

Total Corpus: 289,642 documents

Data Preprocessing Pipeline

Reddit Data	Stock Market Data	PDF Documents
Extracted posts and comments using JSONL format	Parsed CSV files with historical prices	Used PyMuPDF for text extraction
Combined threads (post + comments + replies)	Converted to natural language descriptions	Chunked using RecursiveCharacterTextSplitter
Preserved metadata: subreddit, upvotes, timestamps	Example: "On 24 Nov 2025, META closed at $614.69 (Open: 598.72, High: 615.40, Low: 597.63). Trading volume was 10,708,266 shares."	Chunk size: 1,500 characters with 200-character overlap

System Architecture

1. Document Indexing
   1. Embedding Model: BAAI/bge-large-en-v1.5 (1024 dimensions)
   2. Vector Store: FAISS (Facebook AI Similarity Search)
   3. Index Size: 289,642 vectors
2. Retrieval
   1. Semantic search using cosine similarity
   2. Top-k retrieval (k=5)
   3. Returns relevant context from all sources
3. Generation
   1. Three LLMs tested in parallel
   2. Prompt engineering for financial domain
   3. Context-grounded responses

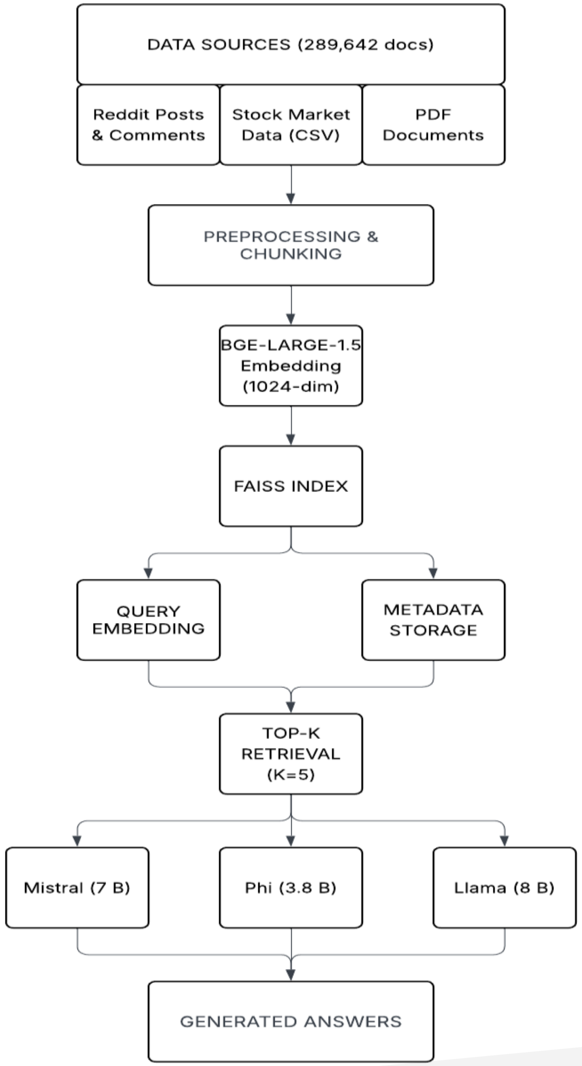

Fig: System Architecture

Language Models Compared

1. **Mistral-7B-Instruct-v0.3**
    1. 7 billion parameters
    2. Instruction-tuned for chat
    3. Known for strong reasoning
2. **Phi-3-Mini-4k-Instruct**
    1. Compact model (3.8B parameters)
    2. 4k context window
    3. Efficient for deployment
3. **Meta-Llama-3.1-8B-Instruct**
    1. 8 billion parameters
    2. Latest LLama architecture
    3. Strong performance on financial tasks

WHY ?

Open-source and reproducible
Instruction-following capabilities
Manageable size for single GPU inference

Code Implementation - Data Preparation

Reddit Processing

```
# Combined posts + comments into threads
for post_id, post in posts.items():
    comment_list = comments_by_link.get(post_id, [])
    full_text = format_thread_text(post, comment_list)
    processed_docs.append({
        "id": f"reddit_thread_{post_id}",
        "source": "reddit",
        "text": full_text,
        "meta": {...}
    })
```

Code Implementation - Data Preparation

Stock Data Processing

```
# Convert CSV rows to natural language
text = f"On {date}, {ticker} closed at ${close:.2f} " \
    f"(Open: {open:.2f}, High: {high:.2f}, Low: {low:.2f}). " \
    f"Trading volume was {volume:,} shares."
```

Google Collab Code Implementation

[Link to Google collab]