

# LLM Analyst: What stocks do you recommend today?

Hyunjong Kim<sup>1†</sup>

Master Student,  
Artificial Intelligence Convergence  
Sungkyunkwan University  
Seoul, Korea  
kim.hyunjong1124@gmail.com

Hayoung Oh<sup>2\*</sup>

Associative Professor,  
Artificial Intelligence Convergence  
Sungkyunkwan University  
Seoul, Korea  
hyoh79@gmail.com

## ABSTRACT

This paper introduces a pioneering approach to stock market analysis that leverages advanced Natural Language Processing (NLP) techniques and dynamic data retrieval systems integrated with Large Language Models (LLMs). Traditional financial analysis often fails to deliver targeted, accessible information for the average investor. Our method utilizes LLMs to interpret and synthesize vast amounts of data, producing comprehensive stock analysis reports. These reports provide nuanced insights into market trends and potential investment opportunities, significantly enhancing the decision-making tools available to retail investors.

By combining the Retrieval-Augmented Generation (RAG) framework—a method that integrates external data dynamically during the generation process—with the LangChain, our methodology enhances both the accuracy and relevance of financial predictions. This integration allows for the production of timely, contextually relevant stock analysis reports, democratizing access to sophisticated financial analysis typically reserved for experts.

Our approach distinctly improves upon traditional methods by providing real-time updates and reducing reliance on manual data interpretation. This can be particularly beneficial for stakeholders ranging from novice investors to seasoned analysts by offering them deeper, actionable insights. Future work could explore incorporating broader data types like real-time news sentiment and global economic indicators, potentially expanding the reach and accuracy of our predictive models. By doing so, this methodology could reshape how financial markets are monitored and analyzed, making advanced market analysis more accessible and actionable for a broader audience.

## 1 Introduction

The stock market generates a huge amount of data every day, including stock prices, trading volumes, financial statements, news articles, and more. This information is not easily understood and utilized by the average investor, which is why analysts' reports play an important role in the stock market, providing investors with deep insights and decision-making [1]. These insights help investors to improve their investment portfolios, make decisions, reduce potential losses, and increase returns. However, it is time-consuming and complicated for ordinary

investors to easily obtain analysts' analysis for every stock, or to analyze data on a daily basis. To solve this problem, we aim to develop a solution that can utilize various structured and unstructured data to provide users with answers similar to analyst reports on the stocks they want.

To this end, we plan to develop an LLM-based chatbot solution that utilizes LangChain, an open-source framework, to obtain data suitable for user queries through APIs and generate reliable answers and analytical reports based on the data. This approach has the advantage of improving information accessibility for ordinary investors, improving the consistency and quality of reports, and instantly reflecting rapidly changing market data.

- We propose to generate analytical reports through a combination of specialized tools within the LangChain framework.

- These tools are uncomplicated, instantly reflect the latest data, and are easy to upgrade or extend.

In this work, we first introduce the utility of equity analysis and LLMs and elaborate on the details of our proposal in Sections 3 and 4. We present the results of our proposed method in Section 5 and Section 6 and draw conclusions.

## 2 Related work

### 2.1 Analysis on Analyst Reports

Publicly traded companies provide investors with a summary of their recent performance on a quarterly basis, and these announcements often highlight revenue realized, earnings per share, and guidance on future business plans. Four times a year, this information is shared with the investing public during quarterly earnings calls. However, in the period between the two earnings calls, equity analysts at various financial institutions attempt to forecast the company's sales and earnings based on their own assumptions. Each group of analysts provides their expectations for quarterly sales and earnings. These analysts' estimates are aggregated in the days and weeks before the actual earnings announcement to give us a general sense of "market expectations" [1].

The importance of analyst reports, highlighted by prominent researchers Jegadeesh et al [2], as a reliable and insightful source for identifying stock and market trends, has inspired much subsequent research [3, 4, 5]. Much of the current literature focuses on the financial metrics or analysts' opinions within the reports [6, 7, 8, 9], pointing to the possibility that important information is hidden within the text, especially given the bias toward buy recommendations in these reports [7, 8].

## 2.2 Large Language Model

In recent years, the field of natural language processing (NLP) has made rapid progress. In particular, Large Language Models (LLMs) have played a key role in many NLP applications. These models have shown remarkable performance in a variety of tasks, including literature review, text summarization, sentiment analysis, and question answering systems [10]. Another study discusses how people's trust in the accuracy of online news articles affects their subsequent behavior and choices. The perceived credibility of digital news sources plays an important role in determining what information people will believe and how they will react to it [9], while another study investigated how natural language processing techniques, such as automated information retrieval, can be used to improve traditional literature review methods [11].

And GPT-3 [12], released in 2020, demonstrated the powerful benefits of a very large language model. GPT-3 has 175 billion parameters, a 100-fold increase over the previous GPT-2 model, and has demonstrated remarkable performance on a wide range of LLM tasks, including code generation, reading comprehension, and question answering, greatly expanding the possibilities in NLP.

Since GPT-3, model sizes have grown to 280 billion [13], 540 billion [14], and 1 trillion parameters [15]. Research has also explored other important aspects of achieving high-performance LLMs: research has extended to different training objectives [16], multilingual models [17], more efficient and smaller models [18], data-finding and parameter-efficient training sizes [19].

Along with these advances in NLP, chatbots are undoubtedly one of the most efficient and widely utilized technologies in human-computer interaction (HCI) [20]; therefore, it is essential to investigate how they have been implemented in customer support. Chatbots are increasingly being used by organizations to provide customer support, partly as a result of fundamental technological innovations [21]; therefore, it is important to understand how technology can help provide better customer service. The literature review therefore focused on two topics: technical assistance for customer service and chatbots in customer care. Recent studies have explored the potential benefits of chatbots for customer service in various industries: Liu and Steins [22] introduced a novel approach with multi-turn answer accuracy, Savavidya and Saha [23] demonstrated the potential for online banking and customer care, and Prabhu et al [24] supported rapid customer care with an artificial intelligence (AI) chatbot. Shehan and Lamont [25] studied customer acceptance leading to chatbot adoption, and Chung et al [26] found that users of chatbot e-services trust customer support

and personalization and scaling [27] by considering appearance and service factors.

## 3 Methodology

### 3.1 Data

Financial data is the foundation of global markets and serves as a key resource for any investment decision. Financial data basically contains various numerical information related to financial markets such as stock prices, financial statements, economic indicators, etc. In this study, we use the Alpha Vantage API to collect stock market data. Alpha Vantage is a service that provides a wide range of financial data and allows users to access different types of data. The types of data utilized in this study include company profiles, weekly stock price data, and income statements.

1. Company Overview: The company overview data provided by Alpha Vantage includes basic business information for each company, key financial metrics, and key closing information for the most recent fiscal year. This information forms the cornerstone of company analysis and serves as an important input variable to the model.
2. Weekly stock price data: Weekly stock price data provides the opening price, high, low, closing price, and trading volume over a specific period of time. This data is used to analyze market trends and the volatility of a stock.
3. income statement: A company's income statement includes its financial performance, including revenue, expenses, and net income, and is an important source of data for evaluating the economic health and profitability of a company.

```
{
  "symbol": "IBM",
  "annualReports": [
    {
      "fiscalDateEnding": "2023-12-31",
      "reportedCurrency": "USD",
      "grossProfit": "3430000000",
      "totalRevenue": "6186000000",
      "costOfRevenue": "2756000000",
      "costOfGoodsAndServicesSold": "245000000",
      "operatingIncome": "6979000000",
      "sellingGeneralAndAdministrative": "19003000000",
      "researchAndDevelopment": "6775000000",
      "operatingExpenses": "27321000000",
      "investmentIncomeNet": "None",
      "netInterestIncome": "-1607000000",
      "interestIncome": "591000000",
      "interestExpense": "1607000000",
      "nonInterestIncome": "-77000000",
      "otherNonOperatingIncome": "266000000",
      "depreciation": "2109000000",
      "depreciationAndAmortization": "2287000000",
      "incomeBeforeTax": "8678000000",
      "incomeTaxExpense": "1176000000",
      "interestAndDebtExpense": "1607000000",
      "netIncomeFromContinuingOperations": "7514000000",
      "comprehensiveIncomeNetOfTax": "5481000000",
      "ebit": "10285000000",
      "ebitda": "12572000000",
      "netIncome": "7502000000"
    }
  ]
}
```

Figure 1: Income statement API call formats

Based on these data, we build the Vector Database. The Vector Database is created through the process of feature extraction and data refinement, and is stored as vectorized data reflecting the

financial and market characteristics of each company. This database will be used in the construction of the Retrieval-Augmented Generation (RAG) model and integration with LangChain. The process of data collection and processing is described in detail to ensure transparency and reproducibility of the study.

The data collection and processing methodology of this study is focused on increasing the accuracy and efficiency of stock market analysis. The data obtained through the Alpha Vantage API comes from reliable sources, which will contribute to more accurate identification and prediction of stock market trends. The combination of the RAG model and LangChain will revolutionize the analysis and interpretation of stock market data. This technological integration will enable precise, data-driven investment decisions and take the application of AI in finance to a new level.

### 3.2 RAG(Retrieval-Augmented Generation)

Traditional Large Language Models (LLMs) have gained traction due to their superior performance, but they have several limitations in real-world applications. First, they require additional training to adapt to new data that they have not been trained on, which is time-consuming and costly. Second, they require resources and effort to customize, especially for enterprises or government agencies dealing with sensitive data. Third, these models can sometimes suffer from halos that produce information that is not true.

Retrieval-Augmented Generation (RAG) [28] is a way to address these problems with traditional LLMs. RAG is a way to improve the performance of a language model by adding an information retrieval phase to the model's answer generation process.

RAG consists of a retrieval phase and a generation phase. In the discovery phase, given a user's question, the RAG model first retrieves relevant data. This is typically done in a document database or knowledge base and selects documents that are highly relevant to the question. In the Generate phase, based on the selected documents, the language model generates contextualized answers. During this process, the model attempts to leverage information from the retrieved documents to provide more accurate and detailed answers.

The benefits of RAG include

- Improved accuracy: By utilizing a separate database that is not trained on the LLM, RAG can provide more accurate information than a simple language model.
- Expand the knowledge base: Rather than relying solely on the data that the model has been trained on, it can search for and incorporate external data in real time to leverage a broader body of knowledge.

Here's how RAG works.

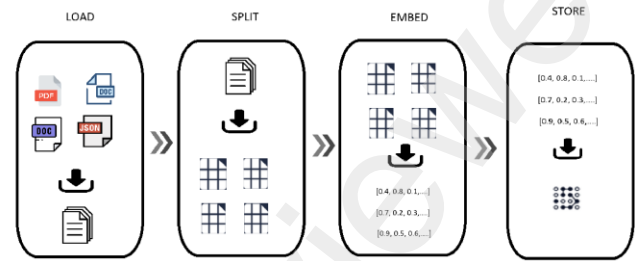


Figure 2: RAG(Retrieval-Augmented Generation): Load - Split - Embed - Store

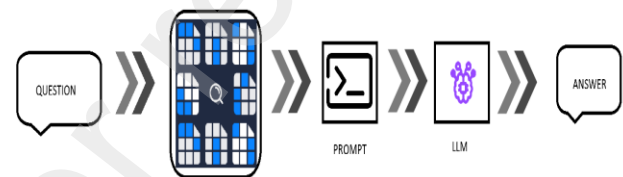


Figure 3: RAG(Retrieval-Augmented Generation): Question - Retrieve - Prompt - LLM - Answer

### 3.3 Langchain

LangChain is a framework for developing applications that leverage large-scale language models, with the goal of making it easy for developers to leverage different data sources and interact with other applications. To do this, LangChain provides components (modular abstractions) and chains (customizable, use-case-specific pipelines). We want to use Langchain to build a RAG. In Langchain, there are additional factors that can increase the performance of RAGs.

#### 3.3.1 Prompt

A prompt is an input statement in the form of a question or request in a conversation between a user and a language model. It plays an important role in determining what type of response the model will provide. The structure and content of the prompt has a significant impact on the quality of the model's output, and proper prompt design is very important.

Prompts typically consist of the following elements

**Question or request:** Clearly state the information you want the user to know or the action you are asking them to take. For example, it might include a specific question, such as "Summarize the recent earnings report for Apple stock."

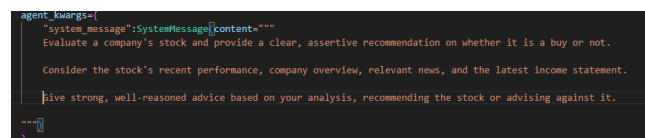
**Instructions:** You can include instructions on how the model should respond and in what format. This helps you specify the style or

structure of the response that the model will provide. For example, you might include instructions such as "Summarize, including recent performance, key financial metrics, and future outlook."

**Context:** Help the model generate more accurate and relevant responses by providing the background information it needs to generate responses. This can include previous conversations or additional comments.

**Example:** You can provide examples to help the model generate better responses. This is especially useful for improving the performance of the model through Few-Shot Learning.

In our research, we use gpt-3.5-turbo to generate stock analysis reports. gpt-3.5-turbo is a large-scale language model developed by OpenAI that can be trained on large amounts of data to provide deep understanding and responses on a variety of topics. To generate a stock analysis report using this model, we utilize the following prompt template:



```
agent_kwargs={
  "system_message": SystemMessage(content="""
    Evaluate a company's stock and provide a clear, assertive recommendation on whether it is a buy or not.
    Consider the stock's recent performance, company overview, relevant news, and the latest income statement.
    Give strong, well-reasoned advice based on your analysis, recommending the stock or advising against it.
  """)
}
```

**Figure 4: RAG(Retrieval-Augmented Generation): Load - Split - Embed - Store**

This prompt template is designed to take a company name from a user and allow the model to generate a comprehensive stock analysis report for that company. It also clearly defines the format and content of the response that the model will provide, ensuring that it provides consistent and useful information.

In conjunction with an Agent: Prompts are sometimes used alone, but for more complex tasks, they are used in conjunction with an Agent. Agents manage the process of collecting the necessary data based on prompts, analyzing it, and generating the final report. For example, an Agent triggered by a prompt can perform the following tasks

- Search for company symbols
- Collect data (company overview, weekly stock price data, income statements, news summaries, etc.)
- Analyze the data
- Generate a final report

This alignment of prompt design and Agent helps the model provide more accurate and specific answers to user questions. Especially in a complex domain like stock analysis, the clarity and

structure of the prompts contribute significantly to the performance of the model.

### 3.3.2 Agent

Agents function like representatives in LangChain, thinking for themselves and choosing the appropriate action for a given query. Agents interact with multiple tools to achieve a given goal, taking different steps to achieve optimal results. In LangChain, Agents are specifically designed to automate complex tasks and respond intelligently to user queries.

The main features and structure of an Agent include

**Autonomous decision-making:** When an Agent receives input from a user, it analyzes it and selects the most appropriate tools and actions. This involves planning and executing the best path to achieve a given goal.

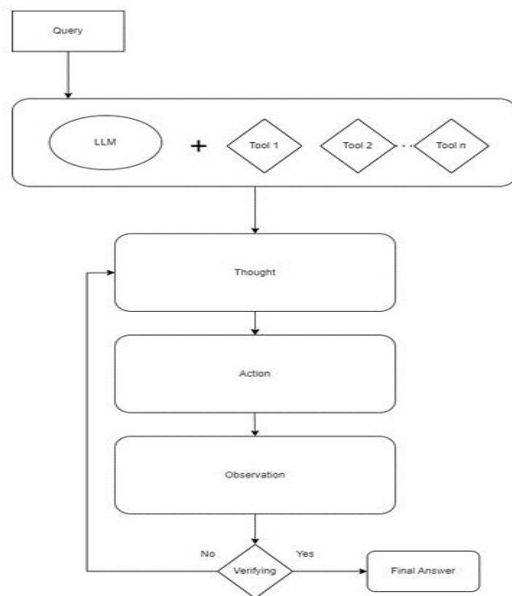
**Tool integration:** Agent is integrated with a variety of tools. These tools can perform multiple functions, including data discovery, information processing, and analysis. Based on a given query and goal, Agent selects the appropriate tools and utilizes them to accomplish its tasks.

**Sequential task processing:** Agent performs tasks in multiple steps. For example, it might first retrieve data, process that data, and then produce a final result. Each step is performed independently, and the results of each step are used as input for the next step.

**Feedback loop:** Agents continuously gather feedback as they perform their tasks, and adjust their behavior based on this feedback. This enables more accurate and efficient task performance.

In our research, we utilize LangChain's Agent to generate stock analysis reports. The Agent receives a query from the user, selects and executes the appropriate tools based on the query to generate the final report.

This autonomy and flexibility of the agent is one of the powerful features of LangChain and can be utilized to solve complex problems in a variety of domains. With the structure and functionality of the Agent, our research is able to effectively analyze stock market data and provide useful insights.



**Figure 4: How Agents and Tools Work**

### 3.3.3 Tool

In the LangChain framework, an Agent utilizes various Tools to perform a given task. In this study, we utilize multiple tools to generate stock analysis reports. Each tool performs a specific task, and the agent combines these tools to achieve optimal results. The main tools used in this study are

**Company Symbol Search Tool:** This tool searches for the stock symbol of a company based on the company name entered by the user. It uses the DuckDuckGoSearch API to search for company names and extracts the stock symbols of those companies from the results.

**Company Overview Tool:** This tool provides basic information about a specific company. It uses the Alpha Vantage API to get the company's basic business information, key financial metrics, closing information for the most recent fiscal year, and more.

**Income Statement Tool:** This tool provides an income statement for a specific company. It uses the Alpha Vantage API to get financial performance such as revenue, expenses, net income, and more.

**Weekly Stock Data Tool:** This tool provides weekly stock price data for a specific time period. Use the Alpha Vantage API to get opening, high, low, close, and volume data to analyze the volatility of a stock.

**Company News Summary Tool:** This tool provides a summary of the latest news related to a specific company. Based on data collected through the News API, it analyzes the latest news stories and summarizes the key takeaways for you to quickly understand.

Each tool fulfills a specific role, and the data obtained from these tools is combined to create a comprehensive stock analysis report. The agent calls these tools in sequence to gather the information it needs, which is then used to build the final report. The interaction between the tools and the agent goes through the following steps

**Input processing:** Takes the company name entered by the user and extracts stock symbols through the symbol search tool.

**Data collection:** Use the extracted stock symbols to collect a company overview, income statement, weekly stock price data, and a summary of company news.

**Data Consolidation:** Consolidate the collected data to generate a comprehensive stock analysis report.

These tools are easily extensible within the LangChain framework, and additional tools can be integrated as needed. This will increase the accuracy and efficiency of stock analysis and provide users with more reliable information.

## 3.4 Vector Store

We will organize and build a Vector Store, which is essential when constructing a RAG. The key to the Vector Store is creating vectors to put into it, which requires vector embeddings. Vector embedding converts text data into vectors in a high-dimensional space to facilitate similarity search and clustering.

There are several vector embedding methods, but for this study, we used the Facebook AI Similarity Search (FAISS) library created by Meta. FAISS is a library designed for efficient similarity search and clustering, and is freely available under the MIT license. Below, we summarize the process of building a Vector Store using FAISS and its advantages.

### Building a vector store

#### 1. Load and preprocess data

- Load text data and preprocess it into the required format. This includes text cleaning, tokenization, and more.

#### 2. Create text embeddings

- Convert the preprocessed text data into vectors. To do this, we use pre-trained language models (BERT, GPT, etc.) to generate text embeddings.

- An embedding is a way of representing each document or chunk of text in a high-dimensional vector space.

#### 3. Store the vectors

- Store the generated vector embeddings in a vector store using FAISS.

- FAISS is designed to efficiently manage and search large amounts of vector data.

#### 4. Indexing

- Index the vector embeddings stored in the vector store for efficient searching.

- FAISS provides multiple indexing methods, which can be optimized for your data size and search requirements.

#### Benefits of FAISS

1. High-performance similarity search: FAISS can perform similarity searches between vector embeddings very quickly and efficiently. This helps maintain real-time search performance even on large datasets.

2. Scalability: FAISS is designed to handle millions to billions of vectors and performs well on large datasets.

3. Multiple indexing options: FAISS offers multiple indexing methods to optimize for different search requirements. For example, it offers options ranging from exact search for greater accuracy to approximate search for speed.

4. Clustering capabilities: In addition to similarity search, FAISS also provides the ability to cluster vector data, which is useful for analyzing the structural characteristics of your data.

5. Open source and community supported: As an open source project, FAISS is supported by an active community and is constantly being updated and improved.

6. Flexibility: FAISS provides Python and C++ APIs to integrate into a variety of applications, and supports a wide range of vector operations required by users.

Building a vector store with FAISS maximizes the performance of RAG models and enables efficient search and analysis on large-scale data.

### 3.5 UI/UX

In this study, we used Streamlit to implement the user interface (UI) and user experience (UX). Streamlit is an open source application framework that helps you quickly create interactive web applications without the need for complex programming. We leveraged the key features of this platform to design and implement the UI/UX for this study.

Streamlit's key features include

Easy to implement: Develop interactive web applications with simple Python scripts without complex coding.

Real-time updates: Data or code changes are reflected in real time and react instantly to user input.

Data visualization: Support for a variety of data visualization tools allows you to display data intuitively.

Flexible layouts: Dashboard layouts can be configured flexibly to meet user needs.

In this study, we implemented a dashboard that uses Streamlit to generate a stock analysis report. The dashboard allows the user to enter the name of a specific company, and a stock analysis report is generated in real-time for the entered company.

The Streamlit dashboard is easy for users to access and provides accurate and reliable stock analysis reports that reflect the latest information in real time. This dashboard effectively communicates research findings to users and is expected to provide practical help to investors.

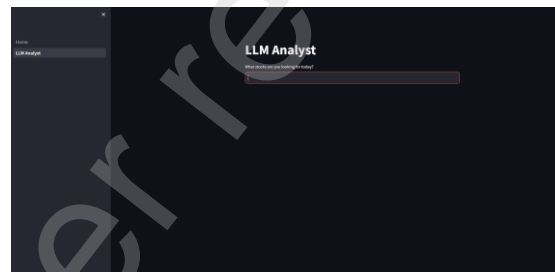


Figure 5: LLM Analyst Main Screen

## 4 Discussion

In this study, we developed a system that combines Large Language Model (LLM) and Retrieval-Augmented Generation (RAG) techniques to generate stock analysis reports. Built utilizing the LangChain framework and Streamlit, the system aims to provide users with reliable stock analysis reports in real-time. In this section, we discuss the implications, limitations, and future research directions of our findings.

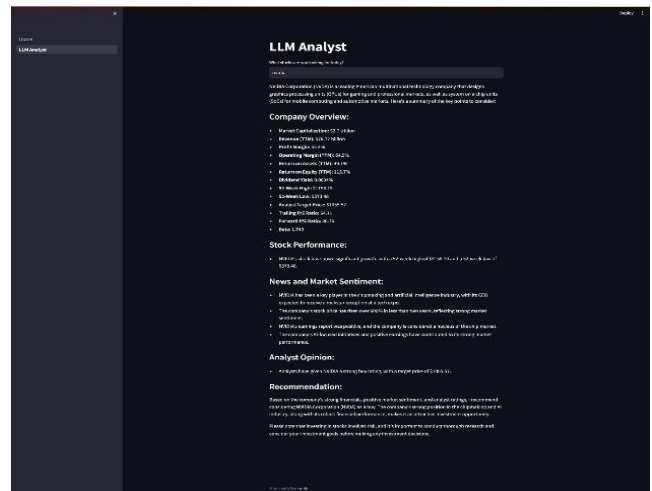


Figure 6: LLM Analyst Results: Nvidia.



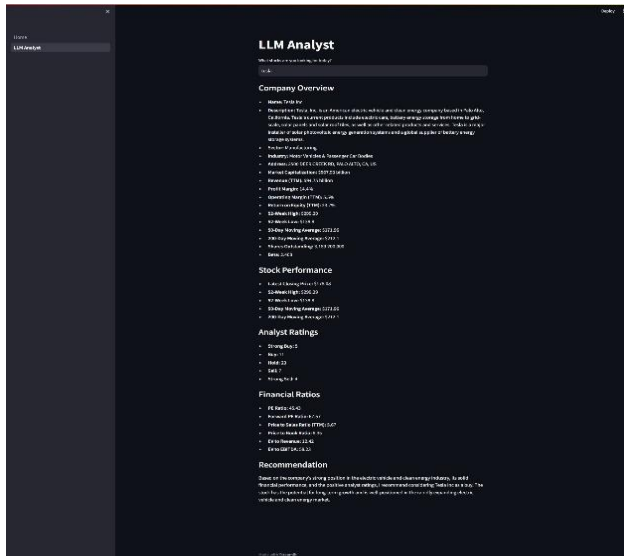


Figure 6: LLM Analyst Results: Tesla.

#### 4.1 What the findings mean

Stock analysis reports are an important resource for investors to make decisions. However, creating these reports takes a lot of time and effort, and it is often difficult for the average investor to get the information they need. The system developed in this research addresses these issues and provides investors with the following benefits

**Improved accessibility of information:** users can receive comprehensive stock analysis reports in real-time for the entered company name. This helps investors quickly get the information they need and make more accurate investment decisions.

**Data integration:** Data from various sources can be consolidated and included in stock analysis reports to provide a comprehensive view. This allows investors to analyze the stock market from a broader perspective, rather than relying on fragmented information.

**Real-time updates:** By generating reports that reflect the latest data, we can react quickly to market volatility. This ensures that investors are always making the most informed decisions.

#### 4.2 Study limitations

While this study has many strengths, it also has some limitations:

**Data quality:** The performance of the system is highly dependent on the quality of the data used. If the data provided by the Alpha Vantage API and the News API is inaccurate or incomplete, the quality of the reports generated may suffer.

**Model limitations:** While LLM and RAG models have many advantages, they are still limited in their ability to fully understand and process complex financial data, particularly in their ability to respond to unexpected events or new market trends.

**User experience:** While the UI/UX utilizing Streamlit is designed to be easily accessible to users, it may not fully meet the needs of certain user groups. For example, you may need a more advanced user interface or customized functionality.

#### 4.2 Future research directions

Future research could develop in the following directions

**Integration of more data sources:** By integrating more data sources, we can increase the accuracy and comprehensiveness of our analytical reports.

**Improving models:** Introduce newer, more powerful LLM and RAG models, and customize models for specific domains to improve performance.

**Add customizable features:** To meet different user needs, you can add personalized report generation capabilities or advanced analytics tools.

**Improve real-time data processing:** For faster data processing and analysis, you can optimize your system's performance and enhance its ability to process real-time data.

**Generate time series images:** Enhance user usability by presenting data not only as text but also as images in multimodal formats.

### 4 Conclusion

In this study, we developed a system that combines Large Language Model (LLM) and Retrieval-Augmented Generation (RAG) techniques to generate stock analysis reports. Utilizing the LangChain framework and Streamlit, we built a dashboard that provides reliable stock analysis reports in real-time when a user enters the name of a specific company.

#### Key achievements

**Efficient data collection and integration:** By integrating various financial data collected through the Alpha Vantage API, we were able to automatically collect and organize the information needed for stock analysis reports.

**Generate accurate and comprehensive analytical reports:** Utilizing the RAG model, we generated comprehensive stock analysis reports that included up-to-date stock price data, financial statements, news summaries, and more.

**User-friendly interface:** The dashboard utilizing Streamlit provided an interface that was easy for users to access and get the information they wanted.

#### Significance of the study

This research has presented an effective solution to help the average investor get reliable stock analysis reports quickly and easily,

which can help them make better decisions and react quickly to stock market volatility.

This research has shown the potential of a stock analysis report generation system to help investors make better decisions. With continued research and improvement, we expect to develop a more sophisticated and useful system. This will take the application of AI in finance to a new level and provide practical help to the average investor.

## REFERENCES

- [1] Jose I. Alvarado & Lindsay C. Clark & Jose A. Gutierrez, 2021. "Stock performance subsequent to combinations in quarterly revenue surprise, earnings surprise, guidance, valuation, and report time," *Journal of Economics and Finance*, Springer;Academy of Economics and Finance, vol. 45(1), pages 95-117, January.
- [2] Narasimhan Jegadeesh, Joonghyuk Kim, Susan D Kriche, and Charles MC Lee. 2004. Analyzing the analysts: When do recommendations add value? *The journal of finance* 59, 3 (2004), 1083–1124.
- [3] Jeffrey A Busse, T Clifton Green, and Narasimhan Jegadeesh. 2012. Buy-side trades and sell-side recommendations: Interactions and information content. *Journal of Financial markets* 15, 2 (2012), 207–232.
- [4] Poongjin Cho, Ji Hwan Park, and Jae Wook Song. 2021. Equity research reportdriven investment strategy in Korea using binary classification on stock price direction. *IEEE Access* 9 (2021), 46364–46373.
- [5] Narasimhan Jegadeesh and Woojin Kim. 2006. Value of analyst recommendations: International evidence. *Journal of Financial Markets* 9, 3 (2006), 274–309.
- [6] Jennifer Francis and Leonard Soffer. 1997. The relative informativeness of analysts' stock recommendations and earnings forecast revisions. *Journal of Accounting Research* 35, 2 (1997), 193–211.
- [7] Narasimhan Jegadeesh, Joonghyuk Kim, Susan D Kriche, and Charles MC Lee. 2004. Analyzing the analysts: When do recommendations add value? *The journal of finance* 59, 3 (2004), 1083–1124.
- [8] Narasimhan Jegadeesh and Woojin Kim. 2006. Value of analyst recommendations: International evidence. *Journal of Financial Markets* 9, 3 (2006), 274–309.
- [9] Hsiou-wei Lin and Maureen F McNichols. 1998. Underwriting relationships, analysts' earnings forecasts and investment recommendations. *Journal of accounting and economics* 25, 1 (1998), 101–127.
- [10] Alquliti, Wajdi Homaid, and Norjihan Binti Abdul Ghani. "Convolutional neural network based for automatic text summarization." *International Journal of Advanced Computer Science and Applications* 10.4 (2019).
- [11] Lechtenberg, Fabian, et al. "Information retrieval from scientific abstract and citation databases: A query-by-documents approach based on Monte-Carlo sampling." *Expert Systems with Applications* 199 (2022): 116967.
- [12] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html>.
- [13] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorraine Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling language models: Methods, analysis & insights from training gopher. *arXiv*, 12 2021. URL <http://arxiv.org/abs/2112.11446>
- [14] Vijay Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeibi, and Bryan Catanzaro. Reducing activation recomputation in large transformer models, 2022. URL <https://arxiv.org/abs/2205.05198>.
- [15] Vijay Korthikanti, Jared Casper, Sangkug Lym, Lawrence McAfee, Michael Andersch, Mohammad Shoeibi, and Bryan Catanzaro. Reducing activation recomputation in large transformer models, 2022. URL <https://arxiv.org/abs/2205.05198>.
- [16] Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. U12: Unifying language learning paradigms, 2022b. URL <https://arxiv.org/abs/2205.05131>.
- [17] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, Fran'cois Yvon, Matthias Gall'e, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Beno'it Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Lauren'con, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris Emezue, Christopher Klamm, Colin Leong, Daniel van Strien, David Ifeoluwa Adelani, Dragomir Radev, Eduardo Gonz'alez Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, G'erard Dupont, Germ'an Kruszewski, Giada Pistilli, Hady Elsahar, Hamza Benaymin, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itzior Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, J'org Froberg, Joseph Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro Von Werra, Leon Weber, Long Phan, Loubna Ben allal, Ludovic Tanguy, Manan Dey, Manuel Romero Mu'noz, Maraim Masoud, Mar'ia Grandury, Mario Sa'sko, Max Huang, Maximin Coavoux, Mayank '51 Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad A. Jauhar, Mustafa Ghalib, Nishant Subramani, Nora Kassner, Nurulqaila Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rhea Harliman, Rishi Bommasani, Roberto Luis L'opez, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, Shayne Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Davut Emre Ta'sar, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesh Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Evry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiangru Tang, Zheng-Xin Yong, Zhiqing Sun, Shaked Brody, Yallow Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeibi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre Fran'cois Lavall'ee, R'emi Lacroix, Samyam Rajbhandari, Sanchit Gandhi,



Shaden Smith, St'ephane Requena, Suraj Patil, Tim Dettmers, Ahmed Barua, Amanpreet Singh, Anastasia Cheveleva, AnneLaure Ligozat, Arjun Subramonian, Aur'elie N'ev'eol, Charles Lovering, Dan Garrette, Deepak Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, Shachar Mirkin, Shani Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zden'ek Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ana Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagholi, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behrooz, Benjamin Ajibade, Bharat Saxena, Carlos Mu'noz Ferrandis, Danish Contractor, David Lansky, Davis David, Douwe Kiela, Duong A. Nguyen, Edward Tan, Emi Baylor, Ezinwanne Ozoani, Fatima Mirza, Frankline Ononiwu, Habib Rezanejad, Hessie Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jesse Passmore, Josh Seltzer, Julio Bonis Sanz, Livia Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, Muhammed Ghauri, Mykola Buryok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nour Fahmy, Olanrewaju Samuel, Ran An, Rasmus Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas Wang, Sourav Roy, Sylvain Viguier, Thanh Le, Tobi Oyeade, Trieu Le, Yoyo Yang, Zach Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Singh, Benjamin Beilharz, Bo Wang, Caio Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Cl'ementine Fourrier, Daniel Le'on Peri'n'an, 52 Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrmann, Gabriel Altay, Giyaseddin Bayrak, Gully Burns, Helena U. Vrabec, Imane Bello, Ishani Dash, Jihyun Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthik Rangasai Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc P'amies, Maria A Castillo, Marianna Nezhurina, Mario S'anger, Matthias Samwald, Michael Cullan, Michael Weinberg, Michiel De Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patrick Haller, Ramya Chandrasekhar, Renata Eisenberg, Robert Martin, Rodrigo Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aaroonsiri, Srishti Kumar, Stefan Schweter, Sushil Bharati, Tanmay Laud, Th'eo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yash Shailesh Bajaj, Yash Venkatraman, Yifan Xu, Yingxin Xu, Yu Xu, Zhe Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. Bloom: A 176b-parameter open-access multilingual language model. arXiv, 11 2022. URL <http://arxiv.org/abs/2211.05100>.

[18] Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieter, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.bigscience-1.9. URL <https://aclanthology.org/2022.bigscience-1.9>.

[19] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack William Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=iBBcRUlOAPR>.

[20] B. A. Eren, "Determinants of customer satisfaction in chatbot use: evidence from a banking application in Turkey," *International Journal of Bank Marketing*, vol. 39, no. 2, pp. 294-311, 2021.

[21] R. C. Ho, "Chatbot for online customer service: Customer engagement in the era of artificial intelligence," in *Impact of globalization and advanced technologies on online business models*: IGI Global, 2021, pp. 16-31.

[22] X. Liu and M. Stynes, "An alternative finite difference stability analysis for a multiterm time-fractional initial-boundary value problem," *East Asian J. Appl. Math.*, vol. 10, pp. 427-436, 2020.

[23] S. Sarbabidya and T. Saha, "Role of chatbot in customer service: a study from the perspectives of the banking industry of Bangladesh," *International review of business research papers*, vol. 16, no. 1, pp. 231-248, 2020.

[24] M. Prabu, T. Sai Tarun, A. Shereef Naina Mohamed, and A. Vijay, "Enhancing customer service using Chatbot application through artificial intelligence," *Journal of Computational and Theoretical Nanoscience*, vol. 17, no. 4, pp. 1633-1637, 2020.

[25] L. Sheahan and S. Lamont, "Understanding ethical and legal obligations in a pandemic: A taxonomy of "duty" for health practitioners," *Journal of Bioethical Inquiry*, vol. 17, no. 4, pp. 697-701, 2020.

[26] M. Chung, E. Ko, H. Joung, and S. J. Kim, "Chatbot e-service and customer satisfaction regarding luxury brands," *Journal of Business Research*, vol. 117, pp. 587-595, 2020.

[27] L. Xu, L. Sanders, K. Li, and J. C. Chow, "Chatbot for health care and oncology applications using artificial intelligence and machine learning: Systematic review," *JMIR cancer*, vol. 7, no. 4, p. e27850, 2021.

[28] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 793, 9459–9474.