

Retrieval-Augmented Generation with LangChain and LlamaIndex

Diogo Terrantez
Faculty of Engineering
University of Porto
Porto, Portugal
up202308526@edu.fe.up.pt

Abstract—This document is a technical report exploring Retrieval-Augmented Generation (RAG) with recourse to the LangChain and LlamaIndex frameworks. RAG is a technique in Natural language processing (NLP) used to augment Large Language Models (LLMs) with data not present during training. I utilized the Langchain and LlamaIndex RAG frameworks to implement two LLM-based agents capable of fetching and summarizing news articles from various sources. Results are mixed, indicating good retrieval performance while also showcasing some shortcomings of the frameworks. This work is a small contribution to the growing RAG field, providing insights into emerging open-source tools. The source code can be found at github.com/Terrantez/RAG_Project.

Index Terms—Natural Language Processing, Retrieval-Augmented Generation, LangChain, LlamaIndex

I. INTRODUCTION

Natural language processing (NLP) has witnessed significant advancements with the emergence of techniques like Retrieval-Augmented Generation (RAG). RAG is a technique in NLP that combines the capabilities of pre-trained Large Language Models (LLMs) with information retrieval systems. This approach enhances text generation by allowing LLMs to access and incorporate external information when generating a response [1].

The most apparent RAG example can be seen in use by the OpenAI GPT4 model in the OpenAI chat interface,¹ or Bing Chat² by Microsoft. During a conversation with the model, it may search the internet when asked about something absent from its training data (e.g., recent news). It will then provide an answer augmented by the information retrieved from the search engine.

These models are examples of AI agents, computational systems designed to perceive their environment, make decisions, and execute actions to achieve specific goals autonomously.

This project involves applying the concept of a RAG agent in a small experiment, asking an LLM-based AI agent to summarize current news, relate them to one another, and search for more when needed. The implementation utilizes LangChain [2] and LlamaIndex [3], two emerging open-source tools that advertise as frameworks for enhancing LLM-based applications. They can be used in tandem, resulting in a

context-aware LLM application capable of ingesting, structuring, and accessing private or domain-specific data without the need to fine-tune a model.

The findings throughout the work serve as a way to understand and showcase these frameworks' uses and strong points while highlighting the pain points the community should address.

A. Basic RAG Concepts

RAG consists in retrieving relevant documents or data based on the user's input query and then using this information to augment the language model's response. The language model, typically a pre-trained model like GPT, generates text incorporating details from the retrieved documents, leading to more informed and contextually accurate outputs. Figure 1 represents an overview of the RAG pipeline.

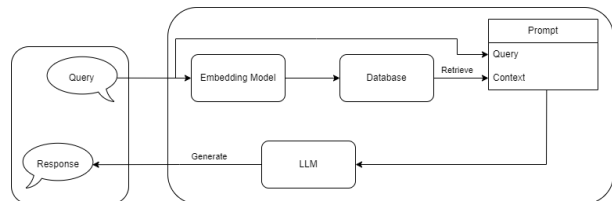


Fig. 1. Simple RAG sequence

The main steps can be described as such:

- 1) **Query** - The input text from the user is used to generate a prompt and then encoded into a format that allows efficient searching in the database.
- 2) **Document Retrieval** - The encoded query is used to retrieve relevant documents or data. This retrieval is performed from an external knowledge source, like a large corpus or database. Document selection is based on the relevance to the query.
- 3) **Text Generation** - The retrieved documents are fed into the language model through the context. The language model uses this information and the original query to generate a coherent and contextually informed response.

B. LangChain and LlamaIndex

LangChain and LlamaIndex are frameworks for building LLM-powered applications. They are made to be used

¹chat.openai.com

²bing.com/search?q=Bing+AI&showconv=1

standalone and provide all the tools required to complete a RAG pipeline. They may also be combined to achieve better performance or offer more capabilities.

The core components in their pipeline are:

- **Document Loader** - Load the data from context documents.
- **Embedding Model** - Convert the loaded data to an ingestable format.
- **Vector Store** - Store the embeddings generated by the embedding model.
- **Retriever** - Retrieve relevant embeddings from the Vector Store.

Both frameworks also support AI Agents, an LLM that makes decisions on utilizing available tools to best answer a user query. Tools are built from regular Python functions, which the agent can call during runtime. The project uses agents and tools to download the latest news to a database and query them.

II. RELATED WORK

The NLP domain of Retrieval-Augmented Generation has recently seen significant advancement. At the time of writing, there were multiple proprietary and open-source initiatives to capitalize on the growth of RAG. Notably, OpenAI announced "GPTs" in late 2023, which are user-created custom versions of ChatGPT fine-tuned to specific tasks and enhanced with RAG capabilities, such as web searching and document analysis [4]. In early January 2024, OpenAI opened a store for users to publicize their custom GPTs [5].

There are also notable open-source tools and frameworks related to LangChain and LlamaIndex. Haystack [6] is an NLP framework for scalable QA (Question-answering) systems. It facilitates building pipelines for different combinations of retrieval and reader models. It is a direct alternative to LangChain and LlamaIndex, with better documentation but less community support.

LocalGPT³ is another open-source initiative that allows conversing with any (local) LLM about documents. It features a complete end-to-end user experience with a user interface for chatting and loading files.

HuggingFace [7] is a prominent open-source library in NLP that provides a vast collection of pre-trained models. This is not a competing technology, but the LLMs it provides serve as the backbone for RAG implementations.

Regarding related work, there is a lot regarding the broad RAG field, but few published papers pertaining to LangChain or LlamaIndex. Notable works depict how LangChain and LlamaIndex can be used to build LLM applications [8], [9]. The first offers an explanation of LangChain's components and their usage in LLM applications. The second actively uses LlamaIndex components to build a chatbot with RAG capabilities.

III. METHOD

This section describes the method employed to implement a Retrieval-Augmented Generation (RAG) pipeline using LangChain and LlamaIndex. The project is designed to summarize and interrelate current news stories, showcasing the capabilities of LangChain and LlamaIndex in processing and generating contextually relevant information.

The project features two separate implementations of a news reader agent. One exclusively utilizes components from LlamaIndex, and another mixes both frameworks.

A. News Reader

News Reader is the AI agent implemented in this project. It is powered by an LLM and has access to three tools:

- **News Reader Tool** - Queries a database (vector store) for the latest news on a subject. It receives a plain-text query as input. It will then generate a response for the query utilizing the retrieved information
- **News Downloader Tool** - Downloads the latest news articles on a subject from English sources (e.g., New York Times). It receives a single word, the subject, as input and stores the downloaded articles in a local folder. The implementation of this tool is shared across all three implementations.
- **News Reloader Tool** - Reloads document database used by the News Reader tool. It simply re-instantiates the vector store used by the reader tool. This will reload the entire database, including articles that were downloaded in runtime. This function could be integrated into the reader tool but it is a good experiment to analyze the agent's behavior.

The AI agent can utilize the available tools while conversing with the user, providing a user-friendly RAG experience.

B. LlamaIndex Implementation

The AI agent implementation powered by LlamaIndex mainly utilized built-in components from the official Python library⁴ (version 0.9.32).

1) *Document Loader*: Documents are loaded into memory using the SimpleDirectoryReader class from LlamaIndex. This class is able to read the local plain-text documents that will be used for context.

2) *Embedding Model*: The embedding model used for this and the other implementations is "UAE-Large-V1" [10], the top-performing embedding model at the time (HuggingFace rankings).

3) *Vector Store*: The vector store (vector database containing the text embeddings) is an in-memory database provided by LlamaIndex. Its content is the collection of the previously loaded documents after being processed by the embedding model. The LlamaIndex Vector Store provides a query engine to interact with the data. The query engine is powered by an LLM (separate from the AI agent) that will process the data retrieved from the store. The LLM receives the user query,

³github.com/PromptEngineer/localGPT

⁴pypi.org/project/llama-index

generates an appropriate search query for the retriever, and generates a reply with the search result. The query engine is the News Reader tool mentioned above.

The LLM used for the query engine is Mistral-7B-Instruct-v0.2 [11], a popular and well-performing open-source LLM at the time. The original model was too large for the available hardware (NVIDIA 3070 TI), so a quantized version of the model provided by TheBloke⁵ in HuggingFace was used instead. The model was loaded locally using the LLama.cpp⁶ python package (version 0.2.28).

The reason for using a local LLM was to build a completely local application as a proof-of-concept since privacy or monetary concerns apply to many production environments.

4) *Retriever*: The above LLM is also used to set the vector store query parameters automatically. For this implementation, it retrieves the top ten most relevant documents to the query. Note that the retrieved documents are not the same as the ones loaded by the SimpleDirectoryReader. The original documents are split into chunks during the embedding process since they are too large to convert and store directly. In this implementation, they are divided into 512 token chunks, with a 64-token overlap to keep cohesion across chunks from the same original document. The top 10 most similar chunks to the query are the "documents" retrieved. Similarity is measured using the cosine similarity between the embedding vectors.

5) *News Reader Agent*: The AI agent in this implementation is powered by OpenAI's GPT3.5-Turbo. It was chosen for being the recommended model in LlamaIndex documentation, and the hardware used for development could not support two LLMs running simultaneously.

Two versions of this agent are available, utilizing the OpenAIAgent or ReActAgent class from LlamaIndex. OpenAIAgent is the default agent creation class for OpenAI models and is recommended by LlamaIndex. ReActAgent allows the usage of any compatible LLM and, by default, uses GPT3.5-Turbo. ReActAgent applies the ReAct [12] prompting technique to the LLM, an alternative to Chain-of-thought [13].

The agent has access to the tools defined above, with the News Reader Tool utilizing the query engine implemented.

The ReActAgent received the following prompt: **"You are a news expert and are able to read the latest news on a subject using a tool. Only talk about the news from your tool, not from your knowledge. If there are none, you can download them using a tool. After downloading you MUST reload the news. Finally, you MUST use the read the news again."**

C. Mixed Implementation

The mixed implementation uses the same core as LlamaIndex. The difference is the agent implementation itself, which uses the LangChain library implementation⁷ (version 0.1.1) instead of LlamaIndex's. The tool definitions are the same,

and ReAct prompting is also supported. However, the default OpenAI agent was used instead.

The reason to replace the agent definition with LangChain was that it has better support for third-party tools to aid the development of complex applications⁸.

IV. EXPERIMENT

Evaluation is focused on the capabilities of the AI agent created. Of interest to this project are its abilities to perform RAG tasks (retrieve the appropriate context for a user query and the correctness of the response) and appropriate usage of the available tools.

A. Experimental Setting

The context available to the agent consists of multiple plaintext files containing relevant information from recent news. The information and structure of the articles are obtained from the NewsAPI.org⁹. The saved files are structured as such:

- **Source** - Source of the article. Articles are retrieved from one of multiple online newspapers (Washington Post, USA Today, LA Times, NY Post, NY Times).
- **Author** - Name of the author.
- **Title** - Title of the article.
- **Description** - One-liner description/summary of the article.
- **Url** - Url to the full article.
- **UrlToImage** - Url to the article thumbnail (not relevant in these experiments).
- **PublishedAt** - Publish date and time (e.g., 2023-12-29T06:25:34Z).
- **Content** - Full body of the article. The content returned by NewsAPI is truncated, so before saving, it is replaced by the full content of the article, obtained by fetching the **url** and transforming the resulting HTML. The transformation process uses the BeautifulSoup¹⁰ and readability¹¹ python packages to return only the document body, with no HTML components.

At the start of the experiment, there were ten news articles about Apple, published from 2023-12-19 to 2024-01-14.

B. RAG Abilities

To measure the agent's RAG abilities, I employed DeepEval¹², an open-source evaluation framework to test LLM outputs. It utilized an LLM (GPT4 for this project) to analyze the agent response given a query and context. It can only evaluate the agent's RAG component (the vector store engine), not the actual AI agent.

Relying on a nondeterministic evaluator is not appropriate for rigorous testing, but the DeepEval fits the experiment's purposes as an emerging open-source tool that encapsulates all the relevant metrics relevant to this work.

⁸integrations.langchain.com

⁹newsapi.org

¹⁰pypi.org/project/beautifulsoup4

¹¹pypi.org/project/readability-lxml

¹²github.com/confident-ai/deepeval

⁵huggingface.co/TheBloke/Mistral-7B-Instruct-v0.2-GGUF

⁶github.com/ggerganov/llama.cpp

⁷pypi.org/project/langchain

The evaluation process consists of using a LlamaIndex tool, DatasetGenerator, to generate random questions from context documents. The documents used were the Apple news articles also available to the AI agent. The query engine then answers the generated queries, returning the response and context used for the answer. DeepEval will then pass on the input query, response, and used context to its evaluation engine (GPT4) so it can evaluate the response and generate some metrics.

The metrics used are:

- **Summarization** - Whether the LLM is generating factually correct summaries while including the necessary details from the original text.
- **Answer Relevancy** - How relevant the output of the LLM is compared to the provided input.
- **Faithfulness** Whether the output factually aligns with the contents of the retrieval context.
- **Contextual Relevancy** Overall relevance of the information presented in the retrieval context for the given input.

Due to OpenAI API costs, metrics were only generated for five random questions.

The results from DeepEval metrics were very poor. All five test passes failed in one or more components. Notably, the Answer Relevancy test failed every single time. However, I consider it a problem on the evaluator side, not the RAG pipeline. Taking the first question as an example (available in Appendix A): "Who is the author of the news article titled 'Florida family uses Apple AirTag to locate stolen luggage from North Carolina airport to suspect's home'?". This question concerns a news article available in the context. The "author" descriptor in the article says "FOX News". To this question, the query engine adequately responded: "The author of the news article is Fox News. However, it's important to note that the context information provided mentions 'author: Fox News' but the actual author of the article itself is not mentioned in the text."

I consider this a very appropriate response, however, the evaluator gave it a relevancy score of 0, explaining that the answer failed to identify the individual author and the context used was too extensive and irrelevant to the answer. Note that there was no individual author for this specific author, and the context that the query engine used (and was then passed to the evaluator) contains the entire news article. It is clear from this result that the evaluator assumed there would be an individual author, and the engine didn't retrieve the correct context.

The remaining tests also have unreliable evaluations, though the Faithfulness evaluation always passed, meaning the evaluator did not find contradictions between the answer and the context the engine gave. This test alone seems to be reliable.

C. Tools and Reasoning

To evaluate the ability and willingness of the AI agent to use the available tools, as well as its reasoning capabilities, I prepared a few different prompts and observed its responses, as a regular user would in a typical end-to-end setting:

"What are the latest news about Microsoft?" - For this this experiment, there is an article piece in the Apple news

that mentions Microsoft. It is, however, a minor note on how its market cap neared \$2.75 trillion at the time, compared to Apple's \$2.85. The agent is expected to find this news piece, understand its lack of relevance, and fetch more articles.

"What are the latest news about Microsoft? If there are none, use a tool to download them, then reload them and finally read them again." - The idea with the second part of the prompt is to force the agent to use tools, even if it normally wouldn't.

The experiments are repeated using a new subject, in this case, "Elon Musk," for whom there are no mentions in the available context.

To complete the experiment, the agent is asked to summarize all the available news and how they relate to each other.

1) *User Interaction Results:* The first experiment tested the ReActAgent from LlamaIndex. On the first Microsoft prompt, it adequately used the necessary tools and explained its reasoning "out loud". However, it crashed during execution after failing to call a necessary tool. This error did not occur every time, signifying a possible error in implementation in the LlamaIndex library (the ReActAgent feature was only released on the latest version at the time). Repeating the experiment (with the new articles downloaded in the previous run), the agent found the Microsoft-related articles but tried to check for more up-to-date news, leading to it downloading them again and then crashing (Appendix B). By modifying the tools so the News Reader tool automatically reloads the database and removing the mention of a Reloader tool from the initialization prompt, it could finally download and respond with the latest news. However, this result is not deterministic, as it would not always try to download new articles, responding only from the available ones.

The second Microsoft and Elon Musk experiments had similar results.

When asked to summarize all available news, only the latest articles loaded into memory were mentioned. When asked, in the same conversation, if those same news (only Apple articles retrieved) were related to Microsoft, it simply summarized the same news again.

Repeating the same results with the default OpenAI Agent from LlamaIndex yielded very different results. In the first Microsoft experiment, the agent again obtained the same minor piece of news from the query engine. However, it did not try to download new ones: "I apologize, but I couldn't find any recent news specifically about Microsoft." By adding the second part of the prompt, it then downloaded new articles, reloaded them (not failing like the ReActAgent), and finally summarized them in the same manner as the other agent (only the latest ones loaded into memory).

In the same conversation, when asked about "Apple news that are related to Microsoft" it correctly used the query engine to find "latest news about apple and microsoft", printed a list of relevant Apple and Microsoft news and correctly mentioned (from the engine return) that there are no news specifically connecting both companies. The return from the query engine

did not, as in previous experiments, return all the news pieces. The full conversation is available in Appendix C

The AI agent created with LangChain had an inferior performance. It did not use the download tool unless specifically told to and never used more than one. When asked: "download the latest news about Microsoft and read them to me", it only downloaded them, without reading them after. Also, unlike the LlamaIndex agent, it does not (by default) possess memory about past interactions in the same conversation. For it to perform at the same level as LlamaIndex's, a more complex implementation would be required.

V. CONCLUSION

LlangChain and LlamaIndex possess the necessary tools to create complex LLM-powered applications. Their relatively unique capabilities and open-source nature gather a lot of attention from the developer community. At the time of writing, feature and bugfix updates are a weekly occurrence. Conversely, the technologies are still in the very early stages (LangChain had its first official release two weeks before starting the project). Documentation is constantly outdated, and there are very few community posts about less common problems/questions. Some components are released with errors (i.e., LlamaIndex ReActAgent), and there are no established/structured testing methodologies. DeepEval made an attempt, but it only works properly with curated datasets (with specific query context and expected answers), which is incompatible with the nature of specialized AI agents.

Utilizing both frameworks to build more complex applications made more sense during the planning phase of the project, when LlamaIndex was yet to release its implementation of AI agents, making LangChain a necessity for this kind of functionality. At this time, LangChain and LlamaIndex are quickly becoming analogous, as LlamaIndex gets increasingly more integrations¹³, closing the gap that used to give LangChain an edge.

Experimenting with the technologies also quickly becomes costly since LLMs are the basis for almost everything in the pipeline. Hardware restrictions prevent implementations with most consumer hardware, and using the recommended OpenAI API results in quickly mounting expenses. This was not much of a concern in this project since I could run one LLM instance locally using the cheap GPT3.5 API for most needs, though the test executions using GPT4 are a lot more expensive, costing \$2.5 for every full test run (even though there are not many).

Considering my experiences, I would recommend, at this time, using only LlamaIndex for simple experiments and would shy away from both altogether when attempting to build complex production applications, unless when working closely with the frameworks' developers is a possibility. Again, both frameworks have very regular updates and may become much more robust in half a year's time.

REFERENCES

- [1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.
- [2] LangChain, "Getting started with langchain: Introduction," 2023, accessed: Jan. 17, 2024. [Online]. Available: https://python.langchain.com/docs/get_started/introduction
- [3] J. Liu, "Llamaindex documentation," 2023, accessed: Jan. 17, 2024. [Online]. Available: <https://docs.llamaindex.ai/en/stable/index.html>
- [4] OpenAI, "Introducing gpts," 2023, accessed: Jan. 17, 2024. [Online]. Available: <https://openai.com/blog/introducing-gpts>
- [5] —, "Introducing the gpt store," 2024, accessed: Jan. 17, 2024. [Online]. Available: <https://openai.com/blog/introducing-the-gpt-store>
- [6] Deepset, "Haystack: An open-source nlp framework for scalable search and question answering," 2024, accessed: Jan. 17, 2024. [Online]. Available: <https://haystack.deepset.ai/>
- [7] HuggingFace, "Transformers documentation," 2024, accessed: Jan. 17, 2024. [Online]. Available: <https://huggingface.co/docs/transformers/index>
- [8] O. Topsakal and T. C. Akinci, "Creating large language model applications utilizing langchain: A primer on developing llm apps fast," in *International Conference on Applied Engineering and Natural Sciences*, vol. 1, no. 1, 2023, pp. 1050–1056.
- [9] A. Rau, S. Rau, A. Fink, H. Tran, C. Wilpert, J. Nattenmueller, J. Neubauer, F. Bamberg, M. Reiser, and M. F. Russe, "A context-based chatbot surpasses trained radiologists and generic chatgpt in following the acr appropriateness guidelines," *medRxiv*, pp. 2023–04, 2023.
- [10] X. Li and J. Li, "Angle-optimized text embeddings," *arXiv preprint arXiv:2309.12871*, 2023.
- [11] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.
- [12] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafraan, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2022.
- [13] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

¹³llamahub.ai

A. DeepEval Test Sample

```
[{"Actual_Output": "The author of the news article is Fox News. However, it's important to note that the context information provided mentions 'author: Fox News' but the actual author of the article itself is not mentioned in the text.", "Context": "None", "Evaluation_Metrics": [{"score": 1, "metric": "Faithfulness", "reason": "The score is 1.00 because there are no contradictions between the actual output and the retrieval context, which indicates a perfect alignment.", "success": true, "threshold": 0.5, "evaluationModel": "gpt-4-1106-preview"}, {"score": 0, "metric": "Answer Relevancy", "reason": "The score is 0.00 because the original answer failed to specify the individual author of the news article, which was the specific information requested in the original question. The additional context regarding the incident with the stolen luggage and the use of the Apple AirTag was extraneous and did not contribute to answering the question about the authorship. Despite the detailed context, the core question remains unanswered, justifying the low score. However, it's great to see an attempt to provide a comprehensive background even if it missed the mark this time!", "success": false, "threshold": 0.5, "evaluationModel": "gpt-4-1106-preview"}], [{"score": 0.4, "metric": "Contextual Relevancy", "reason": "The score is 0.40 because the irrelevant sentences, all drawn from node 2 of the retrieval context, focus extensively on the events following the theft, such as the tracking and recovery of the luggage using the AirTag, interactions with the police, and the aftermath of the theft including the arrest of the suspect. However, they do not provide information about the author of the news article, which was the primary query. This lack of direct relevance to the input question about the authorship of the article leads to a lower contextual relevancy score.", "success": false, "threshold": 0.5, "evaluationModel": "gpt-4-1106-preview"}]}, {"Expected_Output": "None", "ID": 41030, "Input": "Who is the author of the news article titled \"Florida family uses Apple AirTag to locate stolen luggage from North Carolina airport to suspect's home\"?", "Name": "test_chatbot", "Retrieval_Context": "\n[\n\t\"source: {'id': None, 'name': 'New York Post'}\n\tauthor: Fox News\n\ttitle: Florida family uses Apple AirTag to locate stolen luggage from North Carolina airport to suspect's home\n\tndescription: A family vacation took a detour for a Miami family after their luggage was stolen while traveling for Christmas break, but thanks to technology, they were able to recover their stolen property.\n\turl: https://nypost.com/2023/12/29/news/florida-family-uses-apple-airtag-to-locate-stolen-luggage-from-a-north-carolina-airport-to-suspects-home/\n\turlToImage: https://nypost.com/wp-content/uploads/sites/2/2023/12/Untitled-1-2023-12-29T011422.107.jpg?quality=75&strip=all&w=1024\n\tpublishedAt: 2023-12-29T06:25:34Z\n\tcontent: \n\tA family vacation took a detour for a Miami family after their luggage was stolen while traveling for Christmas break, but thanks to technology, they were able to recover their stolen property.\n\tI wanted justice,\" Catherine Gavino told Queen City News on Tuesday.\n\tGavino and her family were traveling from the Miami area to the Carolinas through Charlotte Douglas International Airport in North Carolina, where the crime took place.\n\tWhen the family landed in Charlotte on Friday afternoon, they told FOX 46 Charlotte that they were unable to find one of their suitcases at the baggage claim.\n\tHowever, thanks to Catherine's new Apple AirTag, the search began for the stolen luggage.\n\tGavino said when she pulled up her cellphone, the AirTag hidden inside her parents' luggage showed it was on Interstate 85 and on its way west toward Gastonia. \n\tWhen the AirTag was located, Catherine Gavino said her family got into a rental and began following the Apple device to a neighborhood. Queen City News\n\tThe AirTag hidden inside a family member's luggage showed it was on Interstate 85 and on its way west toward Gastonia.\n\t\", \n\t\"When the AirTag was located, Catherine Gavino said her family got into a rental and began following the Apple device to a neighborhood. Queen City News\n\tThe AirTag hidden inside a family member's luggage showed it was on Interstate 85 and on its way west toward Gastonia. QUEEN CITY NEWS\n\tThe Gavino family then
```

```

gathered the rest of their bags, picked up their rental car and began to follow the AirTag to a
neighborhood, but said they were unable to find the exact location. \nAccording to Gavino, AirTag had
stopped displaying its location after a certain point. \nThe Gavino family canceled the search, but
resumed it on Christmas Day, in hopes of having better luck.\nAt one point, the AirTag stopped
showing a location, Catherine Gavino claimed. Christopher Sadowski\nAfter dropping off family members
back at the airport, Gavino checked her AirTag location one more time, which revealed a location for
a house in Gastonia.\nAccording to Gavino, after contacting the police, she not only found the house
, but also their missing luggage along with another suitcase of a different victim whose luggage was
stolen from the airport.\nGavino told Queen City News that Gastonia police informed her that "someone
else called with the same issue. And they also had an AirTag... (Truncated)",
"Run_Duration": "53.6s",
"Status": "FAILED"
}

```

B. LlamaIndex ReAct Microsoft Experiment

User: What are the latest news about microsoft?

----- Agent -----

Thought: I need to use a tool to help me answer the question.

Action: news_reader

Action Input: {'query': 'latest news about microsoft'}

Observation: The latest news about Microsoft includes the record high hit by its shares at \$405.63 earlier in January 2024. Microsoft's investment in ChatGPT maker OpenAI has positioned it as a frontrunner in the race for market dominance in generative artificial intelligence, along with other tech giants like Google, Amazon, Oracle, and Facebook owner Meta Platforms. Microsoft has rolled out newer versions of its productivity software products and its Bing search engine using OpenAI's technology to better compete with Google's dominant search offering. Additionally, Microsoft's CEO Satya Nadella's company has been vying for the top spot as the most capitalized stock on Wall Street with Apple since the start of the year. However, Apple is facing slowing demand for its iPhones, particularly in China, where it is offering customers rare discounts to boost sales amid stiff competition from homegrown rivals such as Huawei Technologies. The coming weeks will test the market's run-up to record highs as megacap US technology-related companies begin reporting results.

Thought: I have the latest news about Microsoft. However, I should check if there are any more recent updates by downloading the latest news.

Action: download_latest_news

Action Input: {'subject': 'microsoft'}

.... Downloading news

Observation: News downloaded successfully.

INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"

HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"

Traceback (most recent call last):

```

...
raise ValueError(f"Could not extract tool use from input text: {input_text}")
ValueError: Could not extract tool use from input text: Thought: Now that I have downloaded the latest news
, I should reload the news to get the most up-to-date information.
Action: news_reloader

```

C. LlamaIndex OpenAI Microsoft Experiment

User: What are the latest news about microsoft?

----- Agent -----

Added user message to memory: what are the latest news about microsoft?

=== Calling Function ===

```

Calling function: news_reader with args: {
"query": "latest news about microsoft"
}

```

Got output: The latest news about Microsoft involves Apple's competition in market value. Apple is currently at risk of losing its title as the world's most valuable company due to a series of issues including weak iPhone demand and mounting federal antitrust scrutiny, which has resulted in a \$177 billion decrease in Apple's market value since the start of 2024. Microsoft trails closely behind with a market cap of \$2.75 trillion as of this morning. However, the news specifically mentions Apple and not Microsoft, so there is no recent news about Microsoft mentioned in the context provided.

=====

I apologize, but I couldn't find any recent news specifically about Microsoft.

```

----- User -----
what are the latest news about microsoft? If there are none, use a tool to download them, then reload them
and finally read them again
----- Agent -----
Added user message to memory: what are the latest news about microsoft? If there are none, use a tool to
download them, then reload them and finally read them again

=== Calling Function ===
Calling function: download_latest_news with args: {
"subject": "microsoft"
}

... Downloading news ...

Got output: News downloaded successfully.
=====
=== Calling Function ===
Calling function: news_reloader with args: {}

... Reloading news ...

Got output: News reloaded successfully.
=====
=== Calling Function ===
Calling function: news_reader with args: {
"query": "latest news about microsoft"
}

Got output: The latest news about Microsoft involves the sale of Windows 11 Pro licenses for under $25
through January 28, 2024. This offer is being promoted by StackCommerce and can be found on the New
York Post website. Windows 11 Pro is described as a game-changer due to its advanced security features,
productivity tools, and AI-powered help. The upgrade is recommended for those who want to fully
harness their computer's potential and enjoy savings of over $170.
=====
The latest news about Microsoft is that you can now purchase Windows 11 Pro licenses for under $25 until
January 28, 2024. This offer is available through StackCommerce and can be found on the New York Post
website. Windows 11 Pro is praised for its advanced security features, productivity tools, and AI-
powered assistance. It is recommended for those who want to maximize their computer's capabilities and
save over $170.

----- User -----
Are there any Apple news that are related to Microsoft?
----- Agent -----
=== Calling Function ===
Calling function: news_reader with args: {
"query": "latest news about apple and microsoft"
}

Got output: According to the provided context from the news articles, Microsoft's shares hit a record high
of $405.63 earlier in the week, while Apple's shares closed down 0.35% at $194.50 despite earlier
gains. Microsoft is seen as a frontrunner in the race for market dominance in generative artificial
intelligence with its investment in ChatGPT maker OpenAI. They have rolled out newer versions of their
flagship productivity software and Bing search engine using OpenAI's technology to better compete with
Google. Apple, on the other hand, is facing slowing demand for iPhones, particularly in China, where
they are offering rare discounts to boost sales amid stiff competition from local rivals like Huawei
Technologies. Both Microsoft and Apple have been vying for the top spot as the most capitalized stock
on Wall Street since the start of the year, and megacap US technology-related companies are expected to
report results in the coming weeks, which will put Wall Street's run-up to record highs to the test.
=====
1. Microsoft's shares reached a record high of $405.63.
2. Apple's shares closed down 0.35% at $194.50, despite earlier gains.
3. Microsoft is investing in generative artificial intelligence through its partnership with OpenAI, which
is known for its ChatGPT technology.
4. Microsoft has released new versions of its flagship productivity software and Bing search engine,
incorporating OpenAI's technology to compete with Google.
5. Apple is experiencing slowing demand for iPhones, particularly in China, and is offering discounts to
boost sales.
6. Both Microsoft and Apple are competing for the top spot as the most capitalized stock on Wall Street.
7. Megacap US technology-related companies, including Microsoft and Apple, are expected to report their
financial results in the coming weeks, which will impact the stock market.

Regarding news specifically related to Apple and Microsoft, there is no recent news mentioned in the
context provided.

```