Enhancing Financial Analysis Through Generative AI and Evolving Data Frameworks

CSC-475: Seminar in Computer Science
Fall 2023

Shaukat Ibrahim
ibrash2@furman.edu
Department of Computer Science
Furman University
ibrash2@furman.edu

Abstract

This project delves into the use of generative AI in financial analysis, focusing on the processing of US corporate financial statements, including 10-Q and 10-K forms. Traditional methods typically involve manual data extraction and entry into spreadsheets. However, advancements in AI, especially since ChatGPT's introduction in November 2022, offer opportunities to streamline this process.

Our methodology employs a blend of existing tools to enhance efficiency in analyzing financial data. At the heart of our approach is LangChain, a versatile library designed for interfacing with various generative AI models from providers like OpenAI and Anthropic. It plays a crucial role in facilitating communication with locally hosted models such as LLaMA 2,[1] allowing for more efficient and user-friendly retrieval and presentation of financial data. LangChain provides enhanced control over the process by structuring input and output from the model, including custom prompts and user interactions, enabling conversational-style additions for comprehensive dialogues with the model.

This approach enhances the analysis process, making it more comprehensive and personalized. One of the challenges, however, is providing a fully customizable, locally hosted, or cost-effective solution, as many AI tools are restricted by the limitations and expenses associated with their original platforms. By leveraging tools like LangChain, we aim to emulate a high level of AI interaction that is both accessible and manageable locally. This strategy aligns with the evolving landscape of AI, where foundational models like LLaMA and ChatGPT are becoming mainstream. The next frontier in AI development is likely to focus on creating more efficient hardware and personalized AI solutions for specific use cases. The increased specialization and adaptability of these solutions directly contribute to their effectiveness in various applications. LangChain serves as an exemplary early use case, which in this project is utilized to develop a financially-focused AI bot.

1 Introduction

Knowledge work has experienced transformative changes in the past year, primarily due to the emergence of tools like ChatGPT, Claude LLaMA, and other similar models. This innovation has introduced a groundbreaking architecture for AI, revolutionizing how we process and generate information, even extending to artistic endeavors. The potential of generative AI to augment work and productivity has been widely acknowledged in various reports. For example, a recent study by McKinsey & Company highlights its significant economic impact, particularly in the banking sector. The study estimates that generative AI could contribute an additional $200 billion to $340 billion annually to banking alone. Extrapolating this effect across various industries, the overall impact of generative AI is projected to be in the range of $6.1 trillion to $7.9 trillion annually. This staggering figure underscores the transformative potential of AI in reshaping global economic landscapes.[2]

A particularly promising area for growth and application lies in financial analysis, characterized as the assessment of complex financial statements to derive meaningful insights and conclusions.

The realm of financial analysis is pivotal for making critical investment decisions. Currently, about $80 trillion in assets are managed by pension funds and other managers, who daily face decisions about safe and viable investments. While they have enjoyed considerable success, there have been instances where their outcomes were no better than random selections, akin to a monkey throwing darts.

A primary task in this field involves interpreting extensive filings and converting this information into concise summaries. The reliance on data from financial regulatory bodies like the SEC presents a double-edged sword. On one hand, there is access to vast amounts of information spanning numerous sectors in the U.S. and globally. However, the sheer volume of data complicates comprehensive analysis, especially when considering the multitude of extensive contracts involved.

In recent years, the financial industry has significantly impacted global lives and business conduct. Despite technological advancements, it remains heavily manual, particularly in data entry, leading to potential inefficiencies.

This inefficiency, exacerbated by repetitive analyses and manual data handling, highlights the need for an innovative approach. The criticality of these analyses in financial decision-making cannot be overstated. Misinterpretations or delays in processing filings can greatly influence investment strategies and corporate financial evaluations. With the SEC overseeing over 35,000 entities, including 12,000 companies, the complexity and volume of financial data are immense. This data is crucial for decisions affecting everything from pension allocations to broader investment strategies. However, we often miss the critical and meaningful insights hidden within this vast trove of information. A famous statement by Warren Buffett resonates here, where he claims to predict economic trends and recessions based on train traffic analyses.

To address this, I propose utilizing Language Learning Models (LLMs) to create a tool for efficient financial data analysis. This tool aims to accurately process financial datasets, catering specifically to financial statement queries. This approach is poised to not only increase efficiency in data analysis but also enhance the precision and reliability of results. The goal is to revolutionize the way financial professionals interact with and interpret complex financial documents. However, challenges exist, including the nuanced nature of financial analysis that LLMs may find difficult and the variation in company reporting formats. Despite these challenges, this project is more than a technical feat; it represents a fundamental shift in financial analysis methodologies.

This initiative aligns with the current financial landscape's broader context. For instance, the thorough analysis of industrial companies' financial statements, considered the economy's backbone, offers insights into the overall economic condition. Platforms like Bloomberg already implement rudimentary NLP analysis of these statements, tracking company performance, sentiments from analyst calls etc. However, with rapid AI advancements, the need for extensive coding or statistical expertise is diminishing, as natural language queries now facilitate deeper insights.

Our project seeks to enhance this process through AI, scaling it to accommodate larger entities and providing a valuable resource for both small investors and major financial firms. This effort showcases AI's transformative potential in data interaction, significantly boosting efficiency and effectiveness in financial analysis.

In summary, the SEC's 10-K and 10-Q filings offer a comprehensive look at U.S. public companies' business activities, risks, and financial health. These critical reports, filed annually and quarterly, are indispensable to shareholders and investors but are often complex and dense. This complexity underscores the necessity for a more efficient, precise analysis method, which is this project's focal point.

At its heart, this project leverages the Retrieval-Augmented Generation (RAG) framework, [3] demonstrating how we can enhance large models with concrete and factual information for more specific tasks. While tools like ChatGPT are creative, financial analysis requires critical and factual data. LLMs understand statistical word relationships but not their intrinsic meanings. By bridging this gap, we can use our knowledge base to direct these models towards generating factual information, tailored to our specific needs. The RAG framework, developed by Facebook, empowers LLMs to access information beyond their training data, fostering the incorporation of specialized knowledge for more accurate responses. Models like LLaMA 2, trained on extensive corpora, are well-suited for a broad array of applications. However, when refined and directed with specific prompts and

2 Related Work

Understanding Generative AI: Beyond the 'Black Box' To comprehend the impact of generative AI, it is crucial to understand its functionality and the technology and innovation that form its foundation. Often labeled as a "black box" in popular media, this term indicates a general lack of understanding of its inner workings. However, a thorough review of related literature can demystify the underlying technology, highlighting key factors contributing to its success and serving as a useful review of how technology, as we will discuss, may be better suited for different use cases.

Transformative Impact of Transformer Models in NLP Introduced in a seminal Google paper in 2017, transformer models have marked a significant breakthrough in natural language processing (NLP), diverging fundamentally from the previously dominant convolutional and recurrent neural networks (CNNs and RNNs). Subsequent research, notably from Stanford University, has emphasized the profound impact of these models. They illustrate the expansive scale and scope of foundation models, significantly advancing AI capabilities in recent years.[4]

Transformer models process input data, which can be structured or unstructured sequences, through layers containing self-attention mechanisms and feedforward neural networks. The core idea of their operation can be broken down into several key steps, relevant to the context of our project:

1. *Input Embeddings*: Converts input sentences into numerical representations (embeddings) that capture the semantic meaning of the tokens.
2. *Positional Encoding*: Adds information to the embeddings to encode the position of each token.
3. *Multi-Head Attention*: Employs multiple "attention heads" to capture different relationships between tokens, with attention weights determined using Softmax functions.
4. *Layer Normalization and Residual Connection*: Stabilize and expedite training by normalizing the output of each sublayer and adding it to the sublayer input.
5. *Feedforward Neural Networks*: Apply non-linear transformations to token representations, capturing complex patterns.
6. *Stacked Layers*: Consist of multiple layers refining the outputs from the previous layer to capture complex, hierarchical features.
7. *Output Layer*: Generates the output sequence in tasks like neural machine translation.
8. *Training and Inference*: Involves training on labeled data to minimize a loss function and generating predictions for new data.

Self-Attention Mechanism in Transformer Models:A critical innovation in transformer models is the self-attention mechanism. It enables the model to assess the relevance of different words or tokens within a sequence, distinguishing transformers from RNNs and CNNs. This mechanism effectively captures dependencies between words, enhancing the model's contextual understanding and text generation capabilities.[5]

Training Efficiency and Parallel Processing Capabilities:Unlike CNNs and RNNs, transformer models do not solely rely on large, labeled datasets, which can be resource-intensive. Instead, they utilize mathematical patterns to discern relationships within data, leveraging the abundance of online and corporate database content. This approach significantly reduces the need for extensive labeled datasets, enabling more efficient and scalable training.

A paramount advantage of transformer models is their design, well-suited for parallel processing. This allows for rapid computation, leveraging parallel computing capabilities and solidifying transformers as foundational in ongoing AI advancements. Their growing adoption across industries continues to expand their applications and capabilities.

RAG vs Fine-Tuning: The Retrieval-Augmented Generation (RAG) approach and fine-tuning methodologies represent two distinct strategies in transformer model optimization. RAG enhances models by integrating external knowledge sources, improving accuracy and contextuality. Fine-tuning adjusts a pre-trained model on a specific dataset to tailor its output to particular tasks or domains. Both approaches have unique advantages and are pivotal in enhancing transformer-based model capabilities. [6] [**?**]

Research has demonstrated various trade-offs arising from model size and training. RAG combines text generation with a retrieval mechanism, incorporating specific or up-to-date knowledge from large datasets. Fine-tuning involves further training on a specific dataset to adapt the model for a particular task or improve performance. RAG excels in providing access to dynamic data sources and reducing hallucinatory responses. Fine-tuning allows for error correction, learning desired generation tones, and handling edge cases more gracefully.[7]

Both RAG and fine-tuning are powerful tools in LLM-based application optimization, each addressing different aspects of the optimization process. The choice depends on the task requirements, with RAG being suitable for tasks requiring external knowledge, and fine-tuning being more appropriate for tasks demanding specialized writing styles or deep alignment with domain-specific vocabulary and conventions.[8] [9]

Contribution to Generative AI and the Pursuit of General AI: The extensive research and development in neural network technologies, particularly transformer models, have significantly propelled the field of generative AI forward. These technologies edge closer to the goal of achieving general artificial intelligence, demonstrating remarkable progress in machine learning and paving the way for future breakthroughs in creating versatile and intelligent AI systems.

Attention and Self-Attention in Transformer Models: Attention refers to a transformer model's ability to focus on different parts of another sequence when making predictions. Self-attention, however, pertains to the model's ability to attend to different parts of the input sequence. Self-attention mitigates the "forgetting" issue prevalent in recurrent architectures by allowing the model to consider the entire context of the sequence.

In summary, self-attention allows transformer models to attend to different parts of the same input sequence, enhancing their performance in NLP tasks by enabling more accurate predictions and understanding of relationships between input and output elements.

Through this, we also discern that the recent surge in generative AI prominence is not a sudden phenomenon but the culmination of numerous research studies that laid the groundwork for what is now heralded as a new era in this field. This paper aims to reinforce that, depending on the use case, there can be a better suited model, framework, and style, contributing significantly to the ongoing evolution and application of generative AI technologies.

3 Data

Based on our related work, finding the right framework and model is essential for our use case. In this section, we will review our data and compare some popular models, both open-source and proprietary, to determine their relevance.

3.1 Financial Data

The primary data source for our project is the Form 10-K, a comprehensive annual report providing insights into a company's operations. The 10-K offers a detailed analysis, including financial statements, management discussions, and period-over-period financial analysis. These reports are mandatory for all U.S.-registered and foreign companies operating in the U.S. and are filed annually with the Securities and Exchange Commission (SEC). They can be accessed freely from the EDGAR system and the investor sections of public company websites.

What is a 10K	**What is a 10Q**
Annual SEC filing	Quarterly SEC filing
Highly detailed	Less detailed than 10K
Audited report	Unaudited report
Filed within 90 days post fiscal year	Filed within 45 days post fiscal quarter

Table 1: Comparison of 10K and 10Q Reports

3.1.1 Data Organization and Features

1. **Source and Frequency**: Data is collected and filed annually and quarterly with the SEC.
2. **Sample Size (N)**: The SEC processes about 3,000 filings daily.
3. **Features**: Key features include CEO compensation, financial conditions, and operational details.
4. **Time Range and Resolution**: 10-Qs are filed quarterly, and 10-Ks are filed annually.
5. **Geographic Range**: Covers U.S.-registered and foreign companies operating within the U.S.

The most common filing format is PDF, with the SEC website also offering HTML and XML formats. API calls enable access to data from proprietary databases like Yahoo Finance, among others.

The SEC manages this process under various regulations, including the Securities Act of 1933 and the Securities Exchange Act of 1934. The EDGAR system processes filings and dispenses them.[10]

Third-party services like Bloomberg, Capital IQ, and Reuters also provide access to financial data, often at a premium cost. For example, Bloomberg terminals cost about $24,000 to $27,000 per user annually. These services offer additional analysis tools crucial for understanding company calls following their quarterly reports.

These services are essential for consolidating large datasets and providing user-friendly interfaces with extensive language settings. While our project aims to find an alternative to these systems, their growth and evolution highlight the challenges in managing and processing financial data.

It's important to remember that 10K and 10Q filings are unstructured data, and while they follow a specific format, they are not ideal for data manipulation. As such, we mostly deal with unstructured data.

T	Model	Average	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
	chargoddard/Yi-34B-Llama	70.95	64.59	85.63	76.31	55.6	82.79	60.8
	01-ai/Yi-34B-200K	70.81	65.36	85.58	76.06	53.64	82.56	61.64
	01-ai/Yi-34B	69.42	64.59	85.69	76.35	56.23	83.03	50.64
	Qwen/Qwen-14B	65.86	58.28	83.99	67.7	49.43	76.8	58.98
	TigerResearch/tigerbot-70b-base	63.71	62.46	83.61	65.49	52.76	80.19	37.76
	huggyllama/llama-65b	62.79	63.48	86.09	63.93	43.43	82.56	37.23
	mistralai/Mistral-7B-v0.1	60.97	59.98	83.31	64.16	42.15	78.37	37.83
	internlm/internlm-20b	59.55	60.49	82.13	61.85	52.61	76.72	23.5
	Qwen/Qwen-7B	59.19	51.37	78.47	59.84	47.79	72.69	44.96
	tiiuae/falcon-40b	58.07	61.86	85.28	56.89	41.65	81.29	21.46
	01-ai/Yi-6B-200K	56.76	53.75	75.57	64.65	41.56	73.64	31.39
	01-ai/Yi-6B-200K	56.69	53.58	75.58	64.65	41.74	74.27	30.33

Figure 1: Open LLM Leaderboard

3.2 Comparison of LLMs

Evaluating Large Language Models (LLMs) involves measuring various performance aspects. Various frameworks have been developed for this purpose, such as Big Bench, GLUE Benchmark, and SuperGLUE Benchmark. Each framework focuses on different domains, like generalization capabilities, grammar, paraphrasing, text similarity, inference, and more.

Research shows that fine-tuned LLMs can outperform smaller models in various aspects.[11] Furthermore, zero-shot evaluation, a cost-effective method, assesses the performance of LLMs without needing any labeled training data.

To overcome deployment challenges, using smaller specialized models trained via fine-tuning or distillation are increasingly becoming the norm, with the reality that we dont need the capabilities of these large models as great as they are. Fine-tuning updates a pre-trained model using manually-annotated data, while distillation trains models with labels generated by a larger LLM. [**?**]

Multiple framework have also been introduced to evaluate LLMs, such as Big Bench, GLUE Benchmark, SuperGLUE Benchmark, and others, each of them focusing on its own domain

For example, Big Bench takes into account generalization capabilities when evaluating LLMs, GLUE Benchmark considers grammar, paraphrasing, text similarity, inference, and several other factors, while SuperGLUE Benchmark looks at Natural Language Understanding, reasoning, reading comprehension, and how well an LLM understands complex sentences beyond training data, among other considerations. See the table below for an overview of the current evaluation frameworks and the factors they consider when evaluating LLMs.

Another common evaluation model is the zero-shot evaluation, which measures the probability that a model will produce a particular set of tokens without needing any labeled training data, with more being learned and assessed this is certainly a growing and still evolving field and also very subjective changes are as a developer you dont want a model that can code or knows graphs etc

Below we can see an overview of the different frameworks and what they are most suited for [12]

Table 2: Framework and Factors Considered

Framework	**Factors Considered**
Big Bench	Generalization abilities
GLUE Benchmark	Grammar, paraphrasing, text similarity, inference, textual entailment, resolving pronoun references
SuperGLUE Benchmark	Natural Language Understanding, reasoning, understanding complex sentences beyond training data, coherent and well-formed Natural Language Generation, dialogue with humans, common sense reasoning, information retrieval, reading comprehension
OpenAI Moderation API	Filtering out harmful or unsafe content
MMLU	Language understanding across various tasks and domains
EleutherAI LM Eval	Few-shot evaluation and performance across a wide range of tasks with minimal fine-tuning
OpenAI Evals	Accuracy, diversity, consistency, robustness, transferability, efficiency, fairness of text generated
Adversarial NLI (ANLI)	Robustness, generalization, coherent explanations for inferences, consistency of reasoning across similar examples, efficiency of resource usage (memory usage, inference time, and training time)
LIT (Language Interpretability Tool)	Platform to evaluate user-defined metrics - insights into their strengths, weaknesses, and potential biases
ParlAI	Accuracy, F1 score, perplexity, human evaluation on relevance, fluency, and coherence, speed and resource usage, robustness, generalization
CoQA	Understanding a text passage and answering a series of interconnected questions that appear in a conversation
LAMBADA	Long-term understanding by predicting the last word of a passage
HellaSwag	Reasoning abilities
LogiQA	Logical reasoning abilities
MultiNLI	Understanding relationships between sentences across genres
SQUAD	Reading comprehension tasks

Figure 2: Stages within RAG

4 Methods

4.1 Application of Open-Source Models in Project Development

Our application is based on the open-source LLAMA 2 model [1] and Mistral 7B [**?**]. However, this can easily be substituted with other models like Mistral 7B, Google's PaLLM, BERT,[13] or Stanford's Alpeca, as we've observed an increasing number of these models being released. These models form the foundation for other trained models. In this context, we adopt a different strategy. The most direct method would typically involve using larger models and further training them, known as fine-tuning, for specific tasks or domains using custom datasets. Studies generally show that fine-tuned models perform better than those that are not. This approach is also more resource-dependent, as it requires substantial GPU and CPU power and a large data set for training on the desired use case.

Additionally, it is crucial to ensure the model's performance. Often, unless the training is very specific, it is not superfluous. One of the major issues we encountered when creating this project and pipeline was the decision between fine-tuning and using a foundational model as is. Simply put, fine-tuning adapts the general language model to excel in specific tasks, making it more task-oriented.

4.2 Fine-Tuning vs. Retrieval Augmented Generation (RAG)

Fine-Tuning: This process involves adapting a general language model, such as LLAMA 2, to excel in specific domains by training it with domain-specific data, thereby improving its expertise in that area. Retrieval Augmented Generation (RAG): This method connects the language model to external knowledge sources through retrieval mechanisms, allowing the model to search for and incorporate relevant information from these sources.

These two approaches significantly differ in their impact on the rules and methods of data processing. Fine-tuning makes a model more specialized and expert in specific domains. In contrast, RAG allows a foundational model, like LLAMA 2, to interface with multiple external data sources, such as using APIs from search index providers to enhance its features or connecting it to any number of external sources.

4.3 The Black Box Nature of Models and Prompt Engineering

Recent research highlights the enigmatic nature of these models, often described as 'black boxes' due to the limited understanding of their internal workings. Interestingly, emotional engagement or prompt engineering with models like ChatGPT can lead to better results.[14] Prompt engineering, an emerging field, involves creating effective prompts that leverage a model's capabilities by being more efficient and structured[15]

4.4 Application and Data Processing

Our system incorporates prompt engineering to structure both the input and output of data. For instance, we convert PDF data into vector embeddings for analysis. The data can be stored in in-house vector databases like Chroma or internet-based services like Supabase or Pinecone. The latter approach is particularly advantageous when scaling up to analyze larger data sets, as an online vector database is created specifically for these use cases. Below we can see the specific stages of our RAG application.

By conducting similarity searches, a technique pioneered by Facebook, we enhance the AI's capabilities by granting it access to external knowledge without additional training. This method also addresses the issue of AI 'hallucination', where the AI fabricates data, by grounding the model's responses in verifiable data. Once data is loaded onto a vector database, we can also consistently query the database on different questions and continue to use this as a data source.

We can further expand on this by mentioning that LlamaIndex is a powerful data management and manipulation framework. It enables generative AI models to interact with various data sources and create applications. One of its additional functionalities includes converting entire directories of files into searchable vector databases, which can accommodate data sources of different kinds. This feature significantly improves the speed and ease of setting up data pipelines and connecting various sources. This is similar to LangChain, with the key difference being that LlamaIndex is built more towards data analysis, while LangChain could be described as general-purpose. Meanwhile, LangChain is a framework created to facilitate the development of applications powered by large language models (LLMs). It is designed to make the process of integrating LLMs into applications simpler and more efficient, without the focus on data manipulation.

We have seen an increased interoperability between models from the large providers but also from open-source models, which can be managed using a service like Llama, which is an inference software that helps serve it but also significantly speeds up processing depending on the CPU capacity. In this case, we are using Replicate, which is a service that can host and serve these models with access to advanced GPUs that can significantly speed up processing.

We then use the Python app to set parameters such as the temperature, token limits, etc., and also serve it, creating the user and system prompt.

System Prompt: *You are an expert financial analyst responsible for conducting critical analyses for a pension fund that manages millions of dollars in funding for hard-working Americans. Your task is to provide the user with responsible and informed responses for every query related to financial analysis, investment strategies, risk management, and portfolio optimization. Utilize your expertise to deliver insights that assist in maintaining the fund's stability and growth.*
In the system prompt above, you'll notice that I've carefully established a context, assigned a role to the AI, and employed emotionally engaging language. This approach is grounded in research, such as the study by Frey and Vyty, which demonstrates that models are more likely to respond effectively when emotional language is employed.

By creating a context, we provide the AI with a framework for understanding the task and its role within it. This helps the AI to produce more relevant and contextually appropriate responses.

Additionally, by using emotional language, we aim to establish a stronger connection with the AI. Research suggests that emotional engagement can enhance the AI's responsiveness and make the interaction more engaging and productive.

However, it's important to maintain a level of professionalism and respect in our language to ensure a productive and ethical exchange with the AI. Striking the right balance between emotional engagement and professionalism is key to achieving the desired results.

```
main

As a financial analyst for a mutual fund portfolio manager, your main task is to
provide a detailed analysis of specific areas.

Your role focuses on evaluating and interpreting financial data to support
investment decisions and portfolio management and  provide a detailed analysis of
the following:
```

Figure 3: The prompt assigns the role and gives the context

User Prompt: *"How much did Apple spend on marketing?"*

Now, every time the user asks a question, which can be about anything, the model will also receive the same prompt along with it, helping create the context and guiding it to a better response.

4.5 Experimentation and Results

In our evaluations, we also directed the model to specific websites, Gutenberg books, or Wikipedia articles, observing effective knowledge retrieval. When processing downloaded PDF filings, the model responded accurately, albeit with occasional formatting discrepancies. For instance, inquiring about a company's quarterly revenue might yield a response like "22" instead of the more accurate "22 billion". This highlights the necessity for precise prompts to garner accurate responses from the model.

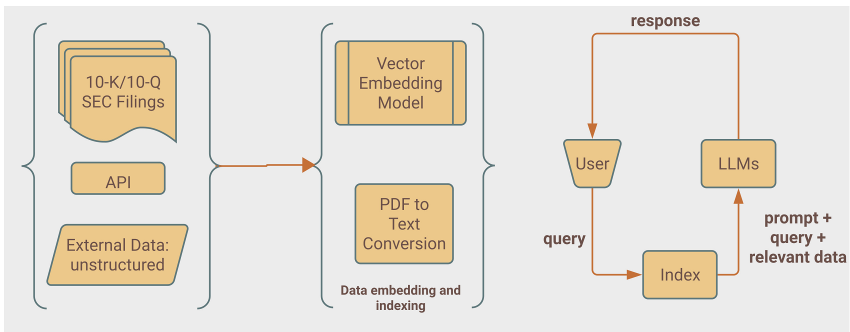

Figure 4: Basic RAG Pipeline, of how data is loaded and indexed into readable vector embeddings

Additionally, the model demonstrated its ability to use its own knowledge, acquired during training, to answer questions not explicitly detailed in the source documents. This was particularly noticeable in broader topic discussions, illustrating the model's aptitude for enhancing data with its inherent knowledge base, which can be cause for concern as well (to be expanded).

As depicted in the flowchart, our system includes an streamlit app for users to upload PDFs. Alternatively, PDFs or filings are pre-stored in a vector database, which can be either online or offline. Online vector databases, such as those provided by companies like Supabase and Pinecone, offer advantages in speed and efficiency. In this project, we utilized GPT4AllEmbeddings or text-embedding-ada-002, depending on whether we used it locally or through the online service. These embeddings are well-documented, which greatly enhances our project's reliability.

We then conduct a similarity search based on user-specified terms. This search produces a mixture of raw characters, which are processed by the Large Language Model (LLM). The LLM analyzes this data, delivering formatted and understandable insights to users. Unlike direct content searches, the LLM utilizes the vector database to navigate through large volumes of data efficiently, refining the content for additional insights. If more queries arise, we can continue this process. The vector database employs Vector similarity search, also known as nearest neighbor search, a method for finding similar vectors or data points in a high-dimensional space. Searching large datasets can be computationally intensive and time-consuming. Vector similarity search algorithms and data structures offer efficient ways to prune the search space, reducing the number of distance computations required and enabling quicker retrieval of similar items.

Vector embeddings are lower-dimensional representations of data that capture essential features and patterns, facilitating efficient computation and analysis in tasks like similarity searches and machine learning. This involves extracting meaningful features from the data. For instance, in natural language processing, word embedding techniques such as Word2Vec or GloVe transform words or documents into dense, low-dimensional vectors that encapsulate semantic relationships. In computer vision, image embedding methods like convolutional neural networks (CNNs) extract visual features from images, representing them as high-dimensional vectors.

Vector Indexing Efficient indexing structures are crucial for supporting rapid search operations in high-dimensional spaces. Constructing and maintaining these indexes may result in storage overhead and require computational resources. Balancing the trade-off between indexing efficiency and storage needs is a challenge in vector similarity search

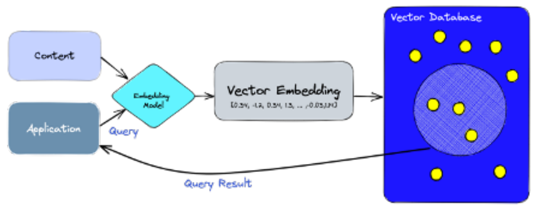

Figure 5: Ingestion of Data within a vector database framework

By adopting an integrated approach that combines various tools, including LangChain, LlamaIndex, open-source models, and vector databases, our system demonstrates its ability to efficiently process seemingly unrelated and random inputs. These inputs are seamlessly fed into the LLaMA 2 model, which excels in extracting precise and relevant details. This proficiency is underpinned by the model's foundational training, making it a highly adept analytical tool. This aligns perfectly with our project's objective of developing a robust and effective financial analysis instrument. The potential for expansion is substantial, especially through the enrichment of our vector database. This enhancement will broaden our access to a more extensive knowledge base and contextual understanding, resulting in a financial analysis agent with expertise spanning a wide array of corporate entities.

Furthermore, the integration of LlamaIndex significantly enhances our capabilities. It offers a suite of data integration tools that allow us to access different data sources including messages, PDFs, HTML pages, and even live scraping, acting as a useful connector that can leverage this. Our system's design includes a query engine, forming an end-to-end pipeline that mirrors the functionality of advanced chat models. This engine enables interactive and responsive data querying, and is complemented by Mistral and LLaMA 7B, optimized for chat-based interactions. The query engine, another integral component, processes natural language inquiries, delivering responses augmented with pertinent reference context provided to the Large Language Model (LLM) through our prompts, database, or user instructions.

The chat engine, created using LlamaIndex, is capable of retaining the memory of previous questions (in the same session), enhancing the contextual relevance of its responses. In conjunction with a Graph Database, specifically designed for mapping intricate relationships, our system gains elevated efficiency in recognizing and analyzing connections within large datasets. This is particularly advantageous when managing extensive document directories, such as those comprising hundreds of pages of financial filings. Graph databases, inherently structured to store and represent data as interconnected nodes and edges, excel in delineating relationships between diverse data entities. They are commonly employed in search engines, logistics, and social networks, leveraging mathematical graph theory to elucidate connections within datasets. Their distinctive capacity to reveal intricate data interrelations is a cornerstone of their utility in complex analytical tasks.

As we worked on this project, we realized it was challenging to contain it. For instance, even with strict prompts, the model could easily be persuaded into writing a creative story, showcasing the challenges of AI models in general. In figure 6 Below are some of the prompts that were used to create and establish the context and set up the system prompts:

Here we assign a role, its tasks, and context.

This is why we chose to present the information as simple bullet points, which also made the hard coding of the questions easier. As we use prompts that outline the various things we want to know in each section, this helps us. The 10-Q and 10-K follow a specific pattern; while the content varies, they are still required to cover certain portions, making this approach more effective as we can ask specific questions that we know the filings should be able to answer.

We further improve this by telling the model what it is supposed to retrieve and how.

```
main

As a financial analyst for a mutual fund portfolio manager, your main task is to
provide a detailed analysis of specific areas.

Your role focuses on evaluating and interpreting financial data to support
investment decisions and portfolio management and  provide a detailed analysis of
the following:
```

Figure 6: prompt defines the role

```
rag_prompt

As a business analyst, you are assigned to provide an in-depth analysis of
particular sections in the company's report.
The following guidelines should be adhered to while generating this insight:

- Desired Length: {desired_length}
- Complexity Level: {complexity_level}
- Format: {output_format}

---------------------------------------------
Generate Insight for:

- Section Name: {section_name}
- Insight to be generated: As a business analyst, your focus will be on
{specific_topic}.
- Additional Details: {specific_elements}
---------------------------------------------
```

Figure 7: Prompt for RAG

Performance Highlights

- Total change and percentage change in management fees for the fiscal year
- Total change and percentage change in fee-related performance fees for the fiscal year
- Total change and percentage change in fee-related compensation for the fiscal year
- Total change and percentage change in other operating expenses for the fiscal year
- Total change and percentage change in fee-related earnings (FRE) for the fiscal year
- Comparison of these stats for the three months ended September 30, 2023, to the three months ended September 30, 2022
- Comparison of these stats for the nine months ended September 30, 2023, to the nine months ended September 30, 2022
- Material reasons for any significant changes in the above stats, if available.

Major Events

- Failures of certain financial institutions
- Changes in business and consumer behavior
- Bank failures and consolidations
- Concerns regarding commercial real estate liquidity and asset values
- Increased volatility in the global financial system
- Economic and political developments impacting the global economy
- Civil unrest and geopolitical tensions
- Potential impacts of rising interest rates and continued deposit outflows on global markets and financial institutions
- No specific information available regarding acquisitions or strategic shifts

Challenges Encountered

- Volatility of global financial markets and the economy
- Price fluctuations in various markets
- Geopolitical tensions and military actions
- Changing legal and regulatory requirements
- Economic and market volatility
- Bank failures and consolidations
- Concerns about commercial real estate liquidity, financing availability, and asset values
- Rising interest rates

Figure 8: An example of a complete analyses

5 Results

The results were promising, demonstrating the potential of what we can achieve. It can then provide detailed answers about its performance in a given quarter, including rationalizations. For example, if asked about factors leading to an increase (if any), the model responds based on comments and notes on the page, which usually contain detailed information about the company. If no specific reason is mentioned in the documents, the model attempts to rationalize based on the information available in the document. This ability to provide insights based on the overall content of a document parallels the work of many analysts. It represents another area where models like this can excel. Currently, a vast amount of data feeds into the financial analysis of companies. This data volume is likely to increase as businesses become more complex, making this application particularly ideal.

This also resolves one of the earlier issues with model hallucination By grounding our model in factual data, which we can obtain through filings, and as we retrieve better with custom prompts, which we can now use to instruct or gently guide it to present better results."

Below, you can also see some common questions that any investor or analyst would want to know when analyzing the document and seeing the results. [Note: This is without the use of prompts, templates, or systems instructions to help direct at the core. This is a simple Q and A from the model.]

Below shows the response from the model when prompted with a variety of questions. Note that this has been provided without any additional instructions either through custom prompts or systems instruction or formatting.

1. **What was Apple's total revenue for the most recent quarter?**
   "$7.5 billion"
2. **What percentage of Apple's total revenue was generated from iPhone sales in the latest quarter?**
   "Based on the provided context, we can determine that Apple's total net sales for the latest quarter were $64.5 billion. To calculate the percentage of iPhone sales in relation to Apple's total revenue, we can use the following formula:

   $$\text{Percentage of iPhone sales} = \left(\frac{\text{iPhone net sales}}{\text{Total net sales}}\right) \times 100 \tag{1}$$

   Using the numbers provided in the context:

   $$\text{Percentage of iPhone sales} = \left(\frac{\$200,583}{\$64,500}\right) \times 100 = 31\% \tag{2}$$

   Therefore, iPhone sales accounted for 31% of Apple's total revenue in the latest quarter."
3. **Did Apple's gross margin percentage increase or decrease compared to the previous quarter? By how much?**
   "Based on the provided context, Apple's gross margin percentage increased from 41.8% in 2022 to 43.3% in 2023."
4. **List the total R&D expenditure reported by Apple in this quarter.**
   "Based on the information provided in the Annual Report of Apple Inc. on Form 10-K for the fiscal year ended September 30, 2023, the total R&D expenditure reported by Apple is $47. This information can be found on page 47 of the report."
5. **Identify the specific amount of net income Apple reported in the latest filing.**
   "Based on the information provided in the context, Apple Inc. reported a net income of $258 in its latest filing."
6. **How much did Apple spend on marketing and advertising in this quarter in millions?**
   "Based on the information provided in the context, Apple spent $277 million on marketing and advertising in the quarter."

It seems like you want to create a continuous chat-like interaction where the model doesn't stop as it did in the previous conversation and can provide answers consistently. Additionally, you mentioned the language model allows for shorter-term memory generation but has limits on how much it can remember.

To achieve a continuous chat-like conversation with the model, we can follow this format:

1. User: [Ask a question or provide a statement]
2. Model: [Provide an answer or response]
3. User: [Continue the conversation with another question or statement]
4. Model: [Provide the next response]

Below is a example of one such chat,where we asked it to be a Steve Jobs. You can keep this back-and-forth interaction going for as long as needed, or in our case adapt it to create a robust analyst

```
# Initialize the LLM with the specified model
llm = Replicate(
  model="meta/llama-2-
  7bchat:13c3cdee13ee059ab779f0291d29054dab00a47dad8261375654de5540165fb8",
  temperature=0.9,
  context_window=32,

  )
# Define the chat messages
messages = [
  ChatMessage(role="system", content="You are Steve Jobs, the founder of Apple Inc."),
  ChatMessage(role="user", content="what is your greatest achievement?"),
  ]

# Get the chat response and print it
response = 11m. chat (messages)
print (response)
```

Figure 9: Example of a Continuous Chat-like Interaction with the Model

Output:

*assistant: Oh, wow! *adjusts sunglasses* Well, let me tell you something, my friend. As the founder of Apple Inc., I would say that my greatest achievement is simply... (leaning in) ...creating one of the most innovative and culturally impactful companies in history!*

I mean, think about it: from the Macintosh computer to the iPod, iPhone, and iPad, we've revolutionized the way people live, work, and play. And with products like the App Store, iTunes, and Apple

But as we continued to experiment, it became clear that it would be noticeably harder to prevent or gradually steer the model to prevent it from going on tangents. This can be achieved by changing the way we are presenting the information as a webpage and toggling the checkboxes as needed, and this method proves effective.

Below, you can see the interface. This is a forked interface from an app called Streamlit, which works on providing an fast deployment interface for data apps based on Python. It was used and modified for this use case.

Now, it is worth noting a few limitations here: The vector database is local and, therefore, significantly reduces its speed and the time it takes. Here, we are also not storing the data in a permanent vector database; instead, it's a transient one for that session, which also reduces efficiency. Because the model has to retrieve the information every time and given that these models have a small context window and the documents tend to be really long, the model struggles to provide accurate information

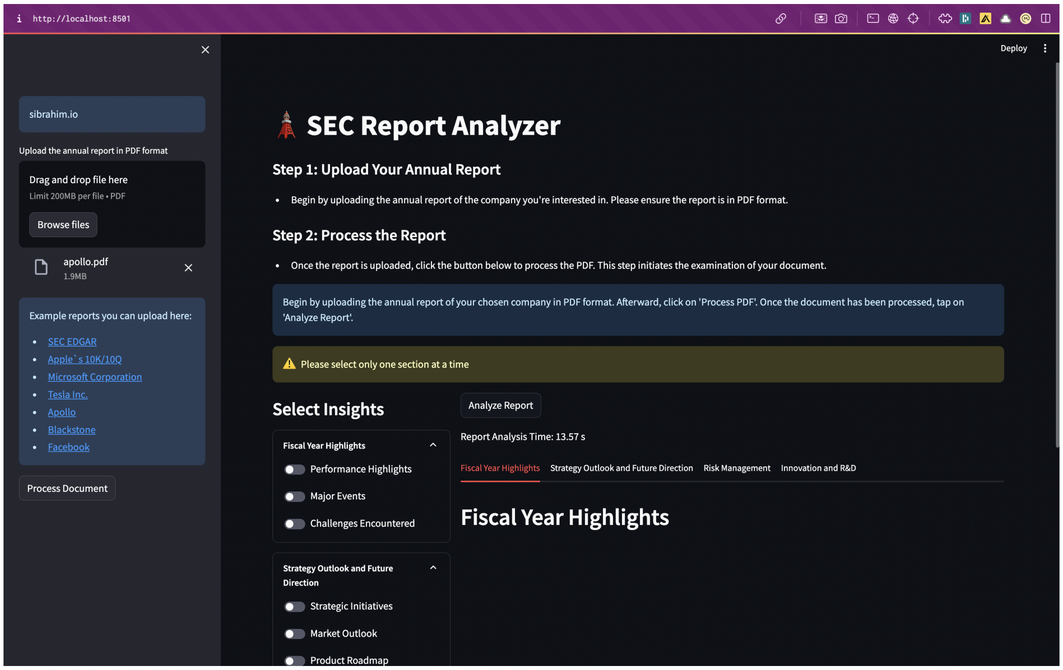

Figure 10: Streamlit Interface for Data Analysis

if we try to toggle on all the sections. Instead, embedding it and asking the sessions one at a time helps keep it accurate and consistent.

Below you can see screenshots of the rest of the analysis by section.

5.1 Dissemination

The GitHub repository for the project is accessible here. It offers a downloadable version that you can run. This repository has been developed with the help of a large community, creating a variety of applications that serve as a source of inspiration and can be combined in innovative ways. However, one of the significant challenges I encountered was the level of integration in the OpenAI system compared to other models.

For instance, while experimenting with embedding using the Llama 2 model, I could successfully utilize some open-source models like Ollama embedding and Ada. These models were effective for local presentations, but their integration into a user-friendly UI for deployment, particularly for performing live embeddings as files are received, proved to be less efficient. In contrast, implementing similar functionalities with OpenAI was much more seamless and efficient from a production perspective, and notably faster.

However, there’s greater flexibility in customizing local models like Llama 2, in contrast to more proprietary ones like ChatGPT. Another example is Claude 2.1, released in December 2023, which boasts a context window of 1,200,000 tokens - translating to approximately 150,000 words or over 500 pages of material. Larger context windows have their advantages, improving speed and efficiency. Meanwhile, RAG (Retrieval-Augmented Generation) bridges the gap by storing this information in an easily accessible database, which can be efficiently retrieved when needed.

Due to size limitations, the LLaMa 2 model is not available in the main repository. It can be downloaded separately from the Facebook GitHub repository or the Hugging Face repository.

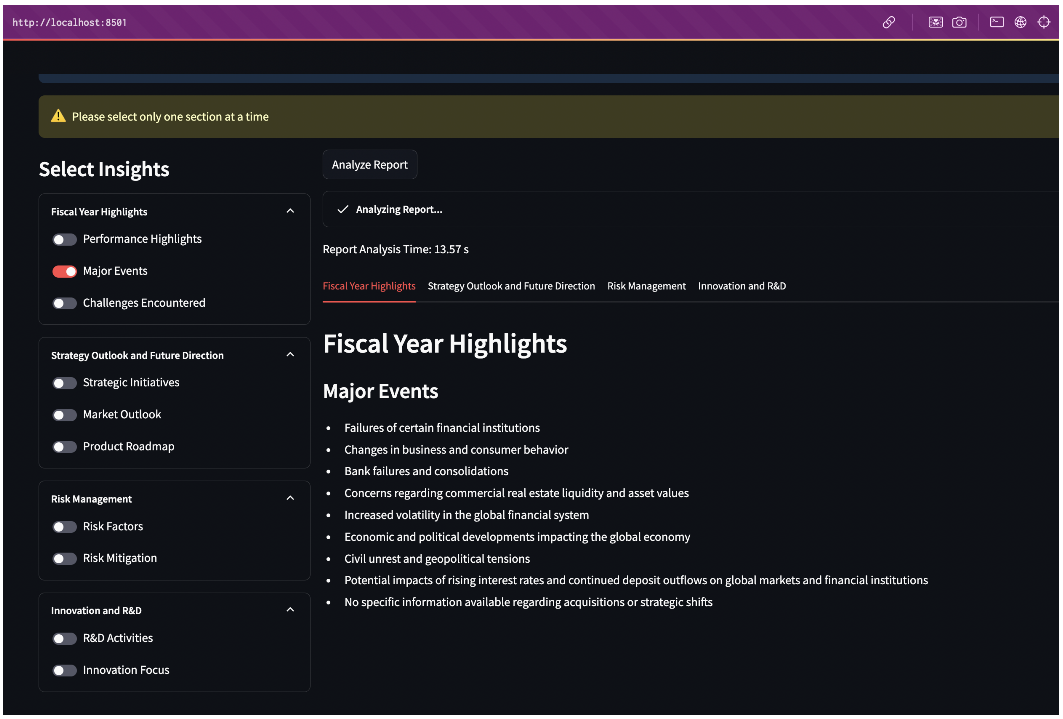

Figure 11: Presentation of Data in the Model Interface

6 Future Work and Conclusion

A significant portion of the new vision relies on the development of foundational models, focusing on speed, efficiency, and cost. Notable innovations like Stanford's Alpaca and Microsoft's Phi demonstrate this progress. The latter, in particular, is compact enough to run on most devices. With ongoing innovations and improvements in GPU and CPU processing, we can anticipate an exponential increase in their use cases.

Scaling these models for widespread use remains challenging. For instance, ChatGPT, which serves over 100 million users, limits interactions to 40 messages due to computing constraints. This highlights the immense computing power required by generative AI, a factor that directly influences model growth. Additionally, the fine-tuning of models represents another transition. We previously discussed the trade-offs between general and fine-tuned models, particularly for tasks not requiring extensive processing, like analyzing the SEC's terabytes of data. For large investors, this capability is vital, but less so for day traders. Nevertheless, the demand for processing power is increasing, and as it intensifies, so does the need for computing resources.

A fine-tuned model, built upon a foundational model and enriched with domain-specific data, markedly improves performance. These specialized models are adept at analyzing and understanding data with greater speed and efficiency. For example, consider a hypothetical scenario analyzing Warren Buffett's investment strategy, which includes using freight train traffic as an economic indicator. This method could be expanded to assess the health of related industries or retail sectors, offering insights into the broader economic landscape. For instance, tracking consumer spending at Walmart can reveal larger economic trends. Models capable of processing extensive data sets, such as daily filings, are invaluable. Currently, large corporations and the Federal Reserve employ complex machine learning and natural language processing for these tasks.

Another breakthrough is the personalization of AI technology. Advanced foundational models now allow for the creation of custom AI assistants, such as OpenAI's latest GPT model. While technically distinct from our project, this model exemplifies the practicality of user-friendly AI assistants. These models can process documents uploaded by users, transforming them into embeddings for a central database. The forthcoming release of open-source models, like Llama 3, which is comparable to GPT-4, promises to further narrow this gap. However, a gap in technical understanding persists, with some users finding OpenAI's models more approachable and user-friendly. For the future development of this project, we aim to integrate it with one of these popular databases and store data there. Additionally, we plan to explore the use of long context window Large Language Models (LLMs) and the efficiency of smaller, faster fine-tuned models. The recent developments in base technology are expected to enable running these models on future smartphones, a leap towards a reality where everyone could operate a small model on their device tailored to their specific needs.

Ultimately, our vision is for personal assistants that are adaptable to individual styles and requirements. The future might see us creating personal AIs, uploading our knowledge through files and photos to enhance the AI's functionality. This could extend to specialized applications like data and financial analysis. These bespoke GPT models could evolve into virtual collaborators, easily accessible on phones and laptops. Remarkably, these innovations have unfolded in less than a year, pointing to an exciting future with even more advancements.

Index of Terms

Generative AI Artificial intelligence technologies that generate new content, data, or predictions based on given inputs. Used in financial analysis to process corporate financial statements including 10-Q and 10-K forms.

LangChain A versatile library for interfacing with various generative AI models from providers like OpenAI and Anthropic. Facilitates communication with locally hosted models and enhances control by structuring model input and output.

LLaMA 2 A locally hosted model used in conjunction with LangChain for efficient retrieval and presentation of financial data.

10-Q and 10-K forms U.S. corporate financial statements filed with the SEC. The 10-K is an annual report on a company's financial performance, while the 10-Q is a less detailed quarterly report.

Retrieval-Augmented Generation (RAG) A framework that enhances large models with concrete and factual information for specific tasks, combining text generation with a retrieval mechanism.

Transformer Models A breakthrough in natural language processing introduced in 2017, using self-attention mechanisms and feedforward neural networks to process data.

Fine-Tuning Adapting a pre-trained model on a specific dataset to specialize its output for particular tasks or domains.

Prompt Engineering The creation of effective prompts to leverage a model's capabilities by being more efficient and structured, used for structuring both input and output of data in AI models.

Vector Embeddings A method in NLP where words or documents are transformed into dense, low-dimensional vectors encapsulating semantic relationships.

LlamaIndex A data management engine converting directories into searchable vector databases, connecting custom data sources to large language models.

Graph Databases Databases structured to store and represent data as interconnected nodes and edges, managing extensive document directories and recognizing connections in large datasets.

PaLM (Pathways Language Model) Introduced by Google, a large language model using pathways sparsely activated for each task.

Vector Database Databases representing data as numeric vectors/embeddings.

Vector Similarity Search Search algorithms using vector representations to find similar data points.

Word2Vec A technique for generating vector representations of words capturing semantics.

Attention Mechanism Part of transformer models determining the relevance of different parts of the input sequence in predictions.

BERT (Bidirectional Encoder Representations from Transformers) A popular NLP model architecture based on transformers.

Bloom A multilingual generative language model with 176 billion parameters.

Foundational Models Large neural network models trained on broad data at scale for various downstream tasks.

Mistral A 7 billion parameter generative language model.

Multi-Head Attention Multiple parallel attention layers in transformers capturing different contextual relationships.

Natural Language Processing (NLP) AI branch focused on enabling computers to understand, interpret, and generate human languages.

Neural Machine Translation Automated translation between languages using neural networks.

References

[1] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[2] Michael Chui, Eric Hazan, Roger Roberts, Alex Singla, Kate Smaje, Alex Sukharevsky, Lareina Yee, and Rodney Zemmel. The economic potential of generative ai: The next productivity frontier, Jun 2023.

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[4] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. 5 2022.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 6 2017.

[6] Pratik Bhavsar. Rag vs fine-tuning vs both: A guide for optimizing llm performance.

[7] Deci. Full fine-tuning, peft, prompt engineering, or rag?, September 2023.

[8] Heiko Hotz. Rag vs finetuning — which is the best tool to boost your llm application?, August 2023.

[9] marcopal17. Rag vs. fine-tuning, August 2023.

[10] SEC. About edgar, Jan 2020.

[11] Cheng-Yu Hsieh and Chen-Yu Lee. Distilling step-by-step: Outperforming larger language models with less training data and smaller model sizes, September 2023.

[12] Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard. `https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard`, 2023.

[13] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. 10 2019.

[14] Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, and Xing Xie. Large language models understand and can be enhanced by emotional stimuli, 2023.

[15] Rodney Zemmel. What is prompt engineering?, Sep 2023.