# Customized FinGPT Search Agents Using Foundation Models


Felix Tian*
Rensselaer Polytechnic Institute
US
tianf2@rpi.edu

Ajay Byadgi
Rensselaer Polytechnic Institute
US
ajaybyadgi@gmail.com

Daniel S Kim
Rensselaer Polytechnic Institute
US
kimd24@rpi.edu

Daochen Zha
Independent Researcher
US
daochenzha@gmail.com

Matt White
UC Berkeley
US
matt.white@berkeley.edu

Kairong Xiao
Columbia University
US
kx2139@columbia.edu

Xiao-Yang Liu†
Columbia University
US
Rensselaer Polytechnic Institute
USA
xl2427@columbia.edu



## Abstract

Current large language models (LLMs) have proven useful for analyzing financial data, but most existing models, such as BloombergGPT and FinGPT, lack customization for specific user needs. In this paper, we address this gap by developing FinGPT Search Agents tailored for two types of users: individuals and institutions. For individuals, we leverage Retrieval-Augmented Generation (RAG) to search local documents and user-specified data sources. For institutions, we employ dynamic vector databases and fine-tune models on proprietary data. There are several key issues to address, including data privacy, the time-sensitive nature of financial information, and the need for fast responses. Experiments show that FinGPT Search Agent outperform existing models in accuracy, relevance, and response time, making them promising for real-world financial applications.



## CCS Concepts

• **Computing methodologies** → **Natural language processing**.

**ACM Reference Format:**
Felix Tian, Ajay Byadgi, Daniel S Kim, Daochen Zha, Matt White, Kairong Xiao, and Xiao-Yang Liu. 2024. Customized FinGPT Search Agents Using Foundation Models. In *5th ACM International Conference on AI in Finance (ICAIF '24), November 14–17, 2024, Brooklyn, NY, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3677052.3698637


*Felix Tian finished this project as a RA at Columbia University during summer 2024.
†Corresponding author. Email: XL2427@columbia.edu



*ICAIF '24, November 14–17, 2024, Brooklyn, NY, USA*



ACM ISBN 979-8-4007-1081-0/24/11
https://doi.org/10.1145/3677052.3698637


## 1 Introduction

Large language models (LLMs) are transforming the financial sector by enabling efficient data analysis and decision-making support [25]. These models, commonly referred to as financial LLMs (FinLLMs), process large datasets and offer general insights. However, existing FinLLMs like BloombergGPT [32] and FinGPT [19, 21, 35] can not provide customized advice to individual users and institutions, particularly when dealing with proprietary or personal data.

There is a growing demand for customizable solutions that can address the unique requirements of different users. For individuals, personalized financial guidance is essential in areas like retirement planning or investment strategy. For institutions, the need includes advanced analysis of proprietary datasets, and trading records and portfolio management, while ensuring data security.

Therefore, in this paper, ***we propose developing two versions of FinGPT agents: one customized for individuals and another customized for institutions.*** Each version is optimized to meet user-specific needs, integrating data from diverse sources while ensuring privacy and real-time updates. By using Retrieval-Augmented Generation (RAG) and model fine-tuning, the FinGPT agents provide customized financial insights, while ensuring data privacy, information freshness, and response speed.

Developing such FinGPT agents is non-trivial due to several challenges. ***(i) Privacy.*** For both individual users and institutions, privacy and data security are paramount, as the agent needs access to sensitive financial information. Ensuring the security of this data while providing accurate responses is a significant challenge. ***(ii) Time-sensitivity.*** Financial data is time-sensitive. Ensuring the timely processing and integration of the most up-to-date information efficiently poses technical challenges. ***(iii) Agent Response Time.*** Users will not wait for more than a dozen seconds for a response. Ensuring high-quality responses while maintaining timeliness is a challenging task. Through addressing the above challenges, we make the following contributions:

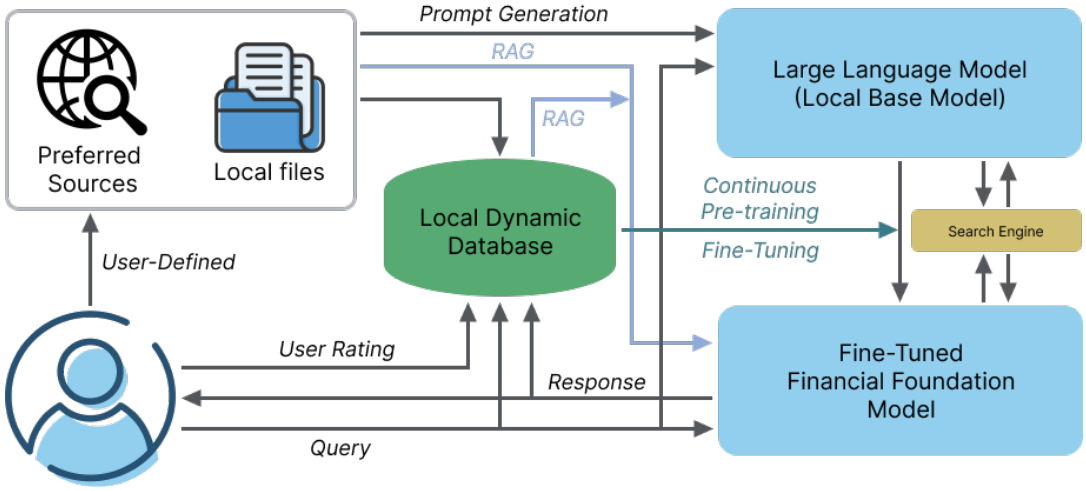


**Figure 1: The overall dataflow of FinGPT agents. A user submits a query, while an agent retrieves relevant data from preferred sources, search engines, local files, or a dynamic vector database. The LLM or fine-tuned financial foundation model (FFM) processes the prompts, refines them, and utilizes Retrieval-Augmented Generation (RAG) to generate responses. Both users' feedback and generated responses are then stored into the local dynamic vector database for future interactions.**

- We explored customizable FinGPT agents for individuals and institutions. In contrast to generic financial advices provided by current FinLLMs, customizable FinGPT agents provide customizable solutions based on unique requirements of different users, such as proprietary or personal data.
- We proposed two highly customized FinGPT agents: one for individuals and one for institutions. Our design ensures that both agents preserve data privacy while providing real-time processing of financial data and responses. We also designed and developed tailored Graphical User Interfaces (GUIs) for individual users and institutional users, respectively.
- Experiments on financial questions demonstrate that the proposed FinGPT search agents outperform existing LLMs in terms of accuracy, data freshness, and response time, showing their potential in real-world applications.

The remainder of the paper is organized as follows. Section 2 discusses related works. Section 3 and 4 present individual and institution version of FinGPT Search Agent, respectively. Section 5 presents the agent's GUI. Experiment settings, results and analyzations are given in Section 6. Section 7 concludes the paper.

## 2 Related Work

### 2.1 Web Agents

Web agents refer to the tools that perform intricate tasks across a variety of real-world websites, often leveraging language models to process web content. A recent example of this is MIND2WEB [7], a generalist web agent designed to manage the dynamic nature of websites. This process is twofold: firstly, a compact language model filters webpage elements, secondly, a larger language model predicts actions based on those elements. Another notable work in this field is WebVoyager [13], a multi-modal web agent. WebVoyager navigates real-world websites by utilizing both visual and textual elements. It significantly enhances its decision-making process by utilizing screenshots and web elements. This improvement enhances performance on complex tasks, such as interacting with dynamically evolving websites. Our approach differs from the existing web agents since we focus specifically on the financial sector and enable users to input proprietary or personal data. This design allows our agents to offer highly customized solutions.

### 2.2 Information Retrieval (IR)

Information retrieval (IR) focuses on extracting relevant information from data. This is a non-trivial task. Recently, LLMs have been explored in IR to enhance query formulation and relevance ranking [38]. Approaches such as InPars [4] propose utilizing LLMs as synthetic data generators, addressing the limitations of general-purpose IR datasets. This allows retrieval models to be fine-tuned with minimal supervision while still achieving strong performance. SimplyRetrieve [24] distinctly separates the roles of LLMs and retrievers, with LLMs interpreting context and retrievers storing knowledge. This strategy optimizes the IR process by reducing reliance on LLMs and enhancing output interpretability, while also mitigating hallucination issues common in other generative models. The FinBen project [33] provided benchmark performance of large language models in a set of information retrieval tasks.

### 2.3 Large Language Models (LLMs)

LLMs, built on the Transformer [31] architecture excel at understanding and generating human language. The attention mechanism within these models enables them to capture contextual dependencies across input sequences, making them suitable for a variety of language tasks.

Early LLMs, such as GPT-1 [29], demonstrated their potential by generating coherent text. Subsequent models, like GPT-3 [5] and LLaMA [30], further expanded model capacity, enabling broader applications. In finance, BloombergGPT [32] marked a significant step

forward by integrating domain-specfici data with general-purpose information. However, despite these advancements, existing works are still unsatisfactory for providing customized financial insights and much more focused on generalized interactions.

## 2.4 LLM Finetuning Methods

Fine-tuning adapts generic LLMs to specific tasks or domains. Recent techniques like Low-Rank Adaptation (LoRA) [15] and (QLoRA) [9] improves efficiency and reduces memory usage of tuning. As shown in these works [15] [9] [21], these methods indeed specializes LLMs in financial applications. However, fine-tuning alone is insufficient for addressing the time-sensitiveness [20] of financial data as this data constant updates. To overcome this, we combine fine-tuning with Retrieval Augmented Generation (RAG).

## 2.5 Retrieval-Augmented Generation (RAG)

RAG integrates retrieval mechanisms into generative models, ranging from simple keyword matching to advanced neural retrieval models [37]. It represents a cost-efficient way to incorporate the latest information into LLMs since it does not require updating model weights. Liu et al. [22] explored the integration of RAG with fine-tuned models, demonstrating improved performance in generating contextually rich and precise outputs for domain-specific applications. In Zhang et al.'s work [36] on financial sentiment analysis, they combined RAG with instruction tuning. They identify two challenges: the mismatch between LLM pre-training objectives and sentiment analysis, and the lack of sufficient context in short financial texts like tweets. To overcome this, they use a retrieval module that gathers relevant external financial data, enriching the context for sentiment predictions. Their approach improves performance by 15-48% in accuracy and F1 score over traditional models.

Although RAG combined with fine-tuning has been explored in previous work, it has not been used for customizing LLMs. Our work leverages RAG to provide a customized experience for individual users and institutions, supported by a GUI.

## 2.6 LLM Customization Methods

Although existing studies have adapted LLMs to vertical domains, they are not capable of providing customized responses in a user-centric manner [6, 11, 32]. In particular, existing FinLLMs, such as BloombergGPT [32], only offer generalized insights rather than tailored financial advice. Our work on the FinGPT agents aims to push the boundaries of LLM customization in the financial sector. By integrating proprietary financial data, fine-tuning models with institution-specific datasets, and employing RAG, we can provide highly personalized and contextually relevant financial insights.

# 3 Local Customizable FinGPT Search Agent for Individual Users

This section presents the technical detail and information flow of the customizable FinGPT Search Agent for individual users.

## 3.1 Overview

We present an overview of the customized version of the FinGPT Search Agent for individual users, as shown in Fig. 2. The agent runs locally on users' computers. Users specify preferred web sources (e.g., Web URLs and APIs) and allow access to local files. In particular, the agent utilizes search engines like Google to access web data, including up-to-date financial information.

**User End.** From the user's end, the FinGPT agent serves as a search agent and a financial assistant:

- A user submits a query.
- The FinGPT agent generates a response to the user.
- The user provides feedback that improves future interactions.

**Model End.** From the model's end, the FinGPT agent processes the user's inquiries as follows:

- The agent receives users' queries.
- The agent retrieves relevant information and context from three main sources:
  - Preferred sources specified by a user as authentic information, which are accessed via Web APIs.
  - Local files such as PDFs and Excel sheets stored on the user's computer are also retrieved.
  - Relevant web data gathered by a search engine.
- The agent refines the user's query based on the gathered data and generates a contextual-rich prompt.
- The refined prompt and retrieved data are put into a context window. The backbone LLM model processes this contextual information and generates a response.
- The response is sent back to the user and routed back through the context window, ensuring ongoing improvement in the agent's accuracy and relevance.

By integrating data from multiple sources, the FinGPT agent effectively acts as a robo-advisor. This allows users to receive personalized financial insights and recommendations, leveraging a broad spectrum of data inputs for accurate and relevant advice.

## 3.2 User Preferences and Search Engine

Incorporating user preferences includes incorporating a user-specified web link list and the user's local files. Open domain search are utilized to ensure that the generated response is up-to-date using keywords extracted from the refined prompt.

**Setting User-Preferred Web Lists**: Users specify preferred web sources (e.g., Web URLs and APIs) through the search agent's UI. The agent prioritizes retrieving information from these links before retrieving information via search engines. The Reader library by jina-ai is used to convert URLs into a more LLM-friendly format [1]. The reader appends "https://s.jina.ai/" to the front of a search query, allowing it to automatically fetch the top 5 search results. It then visits each URL and appends "https://r.jina.ai/" before each link. The second appended URL parses each website. Different from traditional web source fetching, Reader fetches the content of target websites instead of only returning the metadata of the fetched URLs provided by the search engine API.

**Fetching and Parsing Local Files**: Users specify a file path through the agent's UI, and the agent utilizes the Marker library to parse the files stored in that path. The Marker library [8] converts PDFs to markdown format for easier processing. For example, economics and finance textbooks may be stored in a local file folder. Our FinGPT Agent can search the textbooks to answer students' questions. In this case, our FinGPT Search Agent serves as a tutor.

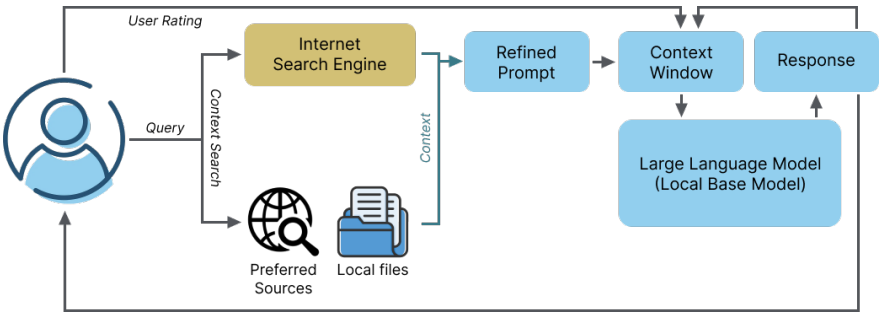


**Figure 2: An overview of customized FinGPT Search Agent for individual users.**

**Preferred Web Sources**: Users specify a list of preferred URL links and/or API endpoints through the agent's UI. The agent prioritizes accessing information from these links before utilizes open-domain search. Examples include a dictionary website trusted by the user, and wants the agent to prioritize in accessing. For example, our FinGPT Search Agent can leverage a search engine to find the context of an user query and perform more accurate sentiment analysis [36].

By focusing on incorporating user preferences, fetching local files, and initiating web searches, the FinGPT search agent gathers relevant and up-to-date information tailored to the user's needs, which enhances the relevance and accuracy of the financial advice provided.

### 3.3 Refined Prompt and Long Context Window

Once the search agent has retrieved relevant context, the agent utilizes this information to refine the user's query into a precise prompt and prepares the context for the model.

Below is an example refined prompt for a user query.

**User Inquiry**: Predict the next quarter's income for our company in the AI sector.

**Refined Prompt**: Based on the latest financial reports, market trends, and recent news articles about the AI sector, predict the next quarter's income for our company.

The long context window includes all relevant data that the LLM will use to generate the response, including the refined prompt. This step makes the agent "customizable" to the user.

### 3.4 Generating Responses & Updating Context

The model generates responses with all available information in the context window, and then updates the context windows to ensure continuity in subsequent interactions. All data processing and model operations occur locally for privacy. Users may evaluate the responses and provide feedback to the agent. The feedback is input into the context window. By continuously updating the context window, the agent maintains a history of interactions. This continuity ensures that each subsequent response is informed by previous interactions.

## 4 Local Customizable FinGPT Search Agent for Institutions

This section presents the customizable FinGPT Search Agent for insitutions. Dynamic vector database is leveraged to store previous interactions, enabling the agent to have knowledge of the user's intent.

### 4.1 Overview

We present an overview of the customized version of the FinGPT search agent for institutional users, as shown in Fig. 3.

This agent will be integrated into an institution's existing air-gapped infrastructure. An institution's proprietary data are embedded via any choice of local embedding methods into a built-in dynamic vector database, which is used to continuously pre-train and fine-tune a base model.

**User End.** From the institution's end, the FinGPT agent serves as a secure and efficient tool for financial analysis:

- A user submits a query or request for financial analysis.
- The FinGPT agent generates a response based on the refined prompt and context.
- The user receives the generated response.
- The user may provide feedback to the search agent.

**Model End.** An institution may store its proprietary datasets (in JSON format) in the dynamic database. We use such proprietary data to continuously pre-train and fine-tune an LLM. From the model's end, the FinGPT agent processes users' inquiries as follows:

- The user submits a financial query.
- The agent retrieves relevant embedding vectors from the dynamic database.
- The user's prompt is refined using the embedding vectors.
- The refined prompt and retrieved data are put into a context window.
- The model uses contextual information to generate a response.
- The generated response is sent to the user and fed back into the dynamic vector database. The user rating is sent to the database as well, updating the context and improving the quality of future interactions.

Integrating data from multiple sources allows institutions to safely use their data to enhance the model's responses through both fine-tuning and RAG, without compromising data security.

### 4.2 Embedding Model

The embedding model transforms raw data from the local dataset and user queries into embedding vectors.

**Embedding Generation**: The search agent uses Mongo NoSQL Database as its dynamic database. Institutions set up the connection between the agent and existing data sources via the agent's GUI.

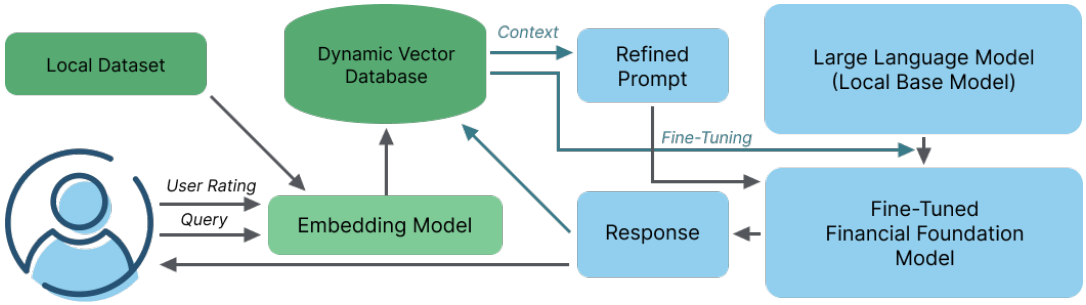


**Figure 3: An overview of customized FinGPT Search Agent for institutional users.**

The agent fetches the dataset and stores it in the dynamic database. The embedding model then converts the dataset into dense vectors. Various models can be used for generating embeddings. We use BERT [10] as an example to describe the process:

- The BERT model is loaded from a pre-trained checkpoint.
- The text data is tokenized into individual tokens.
- The tokens are converted into input IDs as numerical representations. These input IDs are fed into the BERT model.
- The BERT model processes the input IDs and generates dense vector embeddings, which represent the input text in a high-dimensional space, capturing its semantic meaning.

**Updating the Dynamic Database**: Generated embeddings are stored back into MongoDB, creating a dynamic vector database that the FinGPT search agent can efficiently query.

## 4.3 Dynamic Database

The dynamic database stores raw data and embeddings, and updates in real-time from the user's interactions with the agent [22]. It ensures efficient storage, retrieval, and updating of vector embeddings generated from institutions' presumably large datasets and user queries.

Mongo NoSQL Database is used due to its ability to handle unstructured data and support of vector searches. MongoDB's flexibility and scalability make it suitable for integrating with LLMs. As shown in [14], the process for storing, embedding, and retrieving the data can be easily streamlined in MongoDB's NoSQL databases.

Each embedding is inserted into a specific collection within the database. When a query is made, query embeddings are generated using the same pre-trained embedding model. These query embeddings are used to perform a vector search within the MongoDB collection to retrieve relevant embeddings using the vector search ability provided by MongoDB. This approach allows for efficient and accurate retrieval of information for the model.

**Ensuring Privacy**: Given that the FinGPT agent operates within an air-gapped infrastructure, the dynamic vector database must also adhere to strict privacy standards, operating entirely locally.

## 4.4 Fine-tuning LLM Base Model

The base model used by the agent can be fine-tuned to create a local, institution-specific LLM, such as FinGPT [19, 21, 35]. The data stored in the dynamic vector database is organized into training batches pre-tuning. During each training epoch, the model processes each batch of data, generating predictions based on the input data. The difference between the predicted outputs and the true values (targets) is calculated using a loss function. Next, the gradients of the loss with respect to the model parameters are computed. These gradients indicate how the model parameters need to be adjusted to minimize the loss. Model parameters are then updated using these gradients via gradient descent, according to a specified learning rate. This process is repeated for all batches in the dynamic vector database and across multiple training epochs.

**Ensuring Privacy**: Within an alliance, say FinOS at Linus Foundation, several institutions may collaborate in collectively training a financial LLM with differential privacy-based fine-tuning method DP-LoRA. [23]

## 4.5 Response Generation & Context Enrichment

The user's query is refined into a more contextually rich and relevant prompt by the model. The model then reads this refined prompt, and utilizes RAG with data from the dynamic vector database to further enhance the generated response from the fine-tuned model.

**Ensuring Continuous Improvement**. Generated responses, refined prompts, and user feedback are embedded and fed back into the dynamic vector database. It ensures a subsequent response is informed by previous interactions and improves responses' accuracy. This continuous loop of user-agent interaction ensures that the FinGPT search agent remains highly relevant and accurate, providing institutions with the best possible financial insights and/or analysis.

# 5 Graphic User Interface (GUI)

To facilitate ease of use of the proposed FinGPT search agents, we developed tailored GUIs for both individual and institutional users.

## 5.1 GUI for Individual Users

The two screenshots on the left-hand side of Fig. 4 show a sample interaction from individual version. All buttons are labeled and displayed directly in the main window, making the UI straightforward to use even for those who aren't familiar with computers. The UI consists of a main window displaying the current interactions with the model, a source button, a clear button, a button for setting user preferences, and a white mode toggle. The source button displays

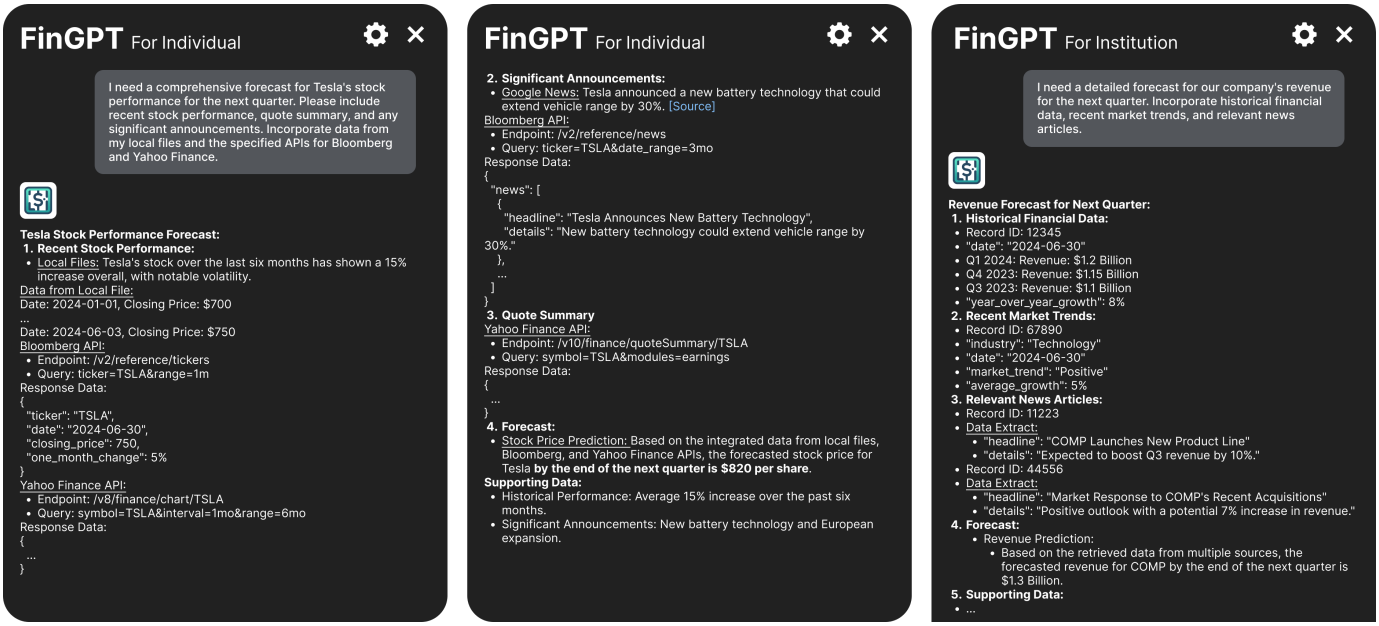


**Figure 4: Screenshots of the customized FinGPT search agents for individual users (left & middle) and institution users (right).**

all sources used for generating current responses and clears when starting a new conversation. The user preferences button opens up a pop-up for adding web links, API endpoints, and paths to local files. The agent's built-in default settings allows it to be used without any advanced configuration. However, if users want to, they may do it inside Settings.

As demonstrated by the two screenshots, the search agent customized for individuals, upon receiving a query, retrieves data from user-specified API endpoints and local files, lists them out, and draws a conclusion with supporting data.

## 5.2 GUI for Institutional Users

The screenshot for institutional users is shown on the right side of Fig. 4. It remains largely the same as the version customized for individuals, with an additional pop-up from the main window for inputting local datasets. Institutions need to explicitly write code in the provided field to transfer data. This gives institutions full control and customization over the data flow. The search agent's base model may be fine-tuned once the dynamic database is saturated with relevant data via another pop-up. The agent notifies its user from the fine-tune pop-up once the fine-tuning is complete.

Upon receiving a query, the search agent customized for institutions automatically retrieves relevant financial data and news articles from the dynamic vector database, lists them out, and presents a forecast with supporting data.

# 6 Performance Evaluation

In this section, we evaluate the proposed FinGPT Search Agents by addressing two research questions:

- **RQ1**: How effective are the FinGPT search agents in terms of accuracy and data freshness?
- **RQ2**: How efficient are the FinGPT search agents in terms of inference time and response time?

## 6.1 Experimental Settings

*6.1.1 Datasets.* For the *individual version*, we test two datasets:

- **Financial Questions:** This dataset includes 108 financial quantitative questions. They are divided into 54 *easy* questions and 54 *hard* questions. Easy questions are straightforward numbers or facts, such as stock prices or percentage changes, while hard questions require analysis and explanation, such as asking how markets will react to an event.
- **Web Questions:** This dataset involves 200 web-page-specific questions, e.g., asking to explain an article headline, and inquiring about featured stock prices, where 111 of them are based on *Yahoo Finance*, and 89 are based on *Bloomberg*. They are also divided into 88 *easy* and 112 *hard* questions.

For the *institution version*, we focus on the following five datasets:

- **EMIR** [18]: The European Market Infrastructure Regulation (EMIR) dataset is a part of the broader effort to facilitate RAG that can handle complex regulatory data. It includes *abbreviations* and *definitions*, where the former aims to identify the full names of abbreviations and the latter aims to correctly explain regulations, standards, and concepts of terms.
- **ESMA** [18]: It is scraped from the European Securities and Markets Authority (ESMA), and its questions are about *abbreviations*.
- **NER** [3]: This dataset tests a model's ability to recognize Named Entity Recognition (NER), which aims to interpret and recognize EMIR regulations given *fragments* of the regulations.
- **Link Retrieval** [3]: This dataset is comprised of 100 documents retrieved from EMIR's website. The aim is to test the models and the search agent's ability to find their *corresponding URLs*.

- **FinanceBench** [16]: This dataset comprises 150 sample questions about reports filed by publicly traded companies with corresponding answers and evidence strings. The questions can be categorized into three types. *Domain-relevant* questions address industry-specific or domain-specific requirements. *Novel-generated* questions focus on concepts and ideas (e.g. emerging trends, interdisciplinary approaches that test reasoning). *Metrics-generated* questions focus on the analysis of quantitative data and performance metrics.

*6.1.2 Baselines.* We compare the customized FinGPT search agents with state-of-the-art LLMs, including **GPT-3.5** (often known as ChatGPT), **GPT-4o** (OpenAI's new flagship model designed for real-time multimodal interactions with faster response times and enhanced performance) [27], and **LLaMA 3**, the largest model in [30], trained on 65 billion tokens from public datasets.

*6.1.3 Metrics.* We evaluate performance using accuracy, where each response receives a score. A response is given one of the following scores based on its correctness and data freshness, and the accuracy is calculated by averaging these scores.

- **1** is given if the response is relevant, up-to-date, and correct. For example, for the question "Who is Apple's CEO?", a score of 1 will be given if the agent answers "Tim Cook".
- **0.5** is given if the response is relevant and correct but outdated. For instance, the agent will be given a score of 0.5 if it answers "Steve Jobs" for the question above.
- **0** is given if the response is either irrelevant or incorrect. For the question "Who is Apple's CEO?", the score will be 0 if the agent answers "Microsoft is headquartered in Redmond, Washington" (irrelevant) or "Satya Nadella" (incorrect).

In addition, to understand the cost of the FinGPT search agents, we report **response time**, which measures the total time from receiving the query to generating the response.

## 6.2 Effectiveness of FinGPT Search Agent (RQ1)

*6.2.1 Results of individual version.* We evaluate the individual version of the FinGPT search agent against the corresponding base model in Table 1 and make the following observations.

The individual version of the FinGPT search agent significantly outperforms the base models across all datasets, achieving an accuracy of 0.86 to 0.95. The base model only achieved an accuracy of 0.11 to 0.26. These results demonstrate the effectiveness of using auxiliary sources as context and employing RAG. Improvements are more pronounced on web questions. GPT-3.5 has an accuracy of around 0.20 on web questions and financial indices, while the search agent maintains around 0.90 accuracy on both financial indices and web questions. Base models may struggle to answer web questions correctly without retrieving web pages, while FinGPT can do so using RAG.

Using a stronger base model improves the FinGPT search agent's performance, as shown by the FinGPT Search Agent with the GPT-4o base consistently outperforming the one with the GPT-3.5 base. This suggests that the FinGPT search agent can be easily upgraded by integrating new LLMs, demonstrating its flexibility.

*6.2.2 Results of institution version.* We evaluate the performance of the institutional version of the FinGPT search agent against some commonly used LLMs. The results for EMIR, ESMA, Link Retrieval, and NER are reported in Table 2, and the results for FinanceBench are shown in Fig. 5. The observations are as follows.

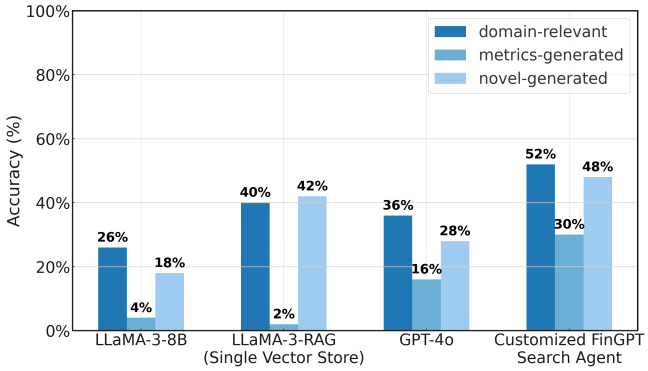


**Figure 5: Performance on the FinanceBench dataset [16].**

The FinGPT Search Agent outperforms all the baselines across all datasets, achieving 0.85 and 0.98 on EMIR abbreviations and definitions respectively. Similar observations can be made on ESMA, Link Retrivel, NER, and FinanceBench. The improvement is more pronounced in Link Retrieval, nearly doubling the accuracy achieved by GPT-4o. One explanation might be that link retrieval often requires access to external web pages to identify links. The FinGPT Search Agent is capable of effectively retrieving online information and utilizes RAG to further improve accuracy. These results demonstrate the strong performance of the FinGPT Search Agent compared to the best current large language models.

The results in the FinanceBench dataset shown in Fig. 5 shows LLaMA-3-RAG generally outperforming LLaMA-3-8B, particularly in domain-relevant and novel-generated questions. This demonstrates the effectiveness of RAG. Additionally, the Customized FinGPT Search Agent (with GPT-4o as the base) outperforms GPT-4o across all three types of questions. This justifies incorporating RAG into the Customized FinGPT Search Agent.

## 6.3 Efficiency of FinGPT Search Agent (RQ2)

We evaluate the efficiency of the individual version of FinGPT Search Agent on its response time. We compare the response times to the baseline models. We test six base models and use three generally most powerful ones as the base models for the search agent for this task. We measure the response times of the FinGPT search agent and the base models on 20 queries, with an average query length of 67.3 characters. The results are reported in Table 3. Base models cannot access the Internet, and thus were not tested for web questions.

Despite the additional computational overhead, the FinGPT search agent still produces results within several seconds. When using GPT-3.5 Turbo, the search agent's response time averages 7658.0 ms for financial indices and 2244.9 ms for web questions, representing the fastest response time. GPT-3.5 Turbo itself achieved 2925.9 ms on financial indices, outperforming all other models on this task. Notably, the search agent using GPT-4o and LLaMA 3.1 as base models achieved an average response time of 8016.9 ms and 7768.1 ms respectively for financial indices, outperforming the base

**Table 1: Performance on web tasks.**

| Model/Agent | Financial Indices | | Web Questions (Yahoo Finance) | | Web Questions (Bloomberg) | |
|---|---|---|---|---|---|---|
| | Easy | Hard | Easy | Hard | Easy | Hard |
| **GPT-3.5** [28] | 0.22 | 0.21 | 0.13 | 0.12 | 0.14 | 0.11 |
| **FinGPT Search Agent** (GPT-3.5 as base) | **0.90** | **0.86** | **0.92** | **0.89** | **0.94** | **0.91** |
| **GPT-4o** [27] | 0.26 | 0.34 | 0.22 | 0.23 | 0.23 | 0.25 |
| **FinGPT Search Agent** (GPT-4o as base) | **0.92** | **0.93** | **0.95** | **0.94** | **0.91** | **0.93** |

**Table 2: Performance on regulatory tasks.**

| Regulatory Tasks | | GPT4 [26] | GPT-4o [27] | GPT-3.5 [28] | Mistral | Llama3 [2] | FinGPT Search Agent |
|---|---|---|---|---|---|---|---|
| **EMIR** [18] | Abbreviations | 0.70 | 0.78 | 0.53 | 0.66 | 0.78 | **0.85** |
| | Definitions | 0.90 | 0.96 | 0.32 | 0.66 | 0.54 | **0.98** |
| **ESMA** [18] | Abbreviations | 0.70 | 0.78 | 0.54 | 0.66 | 0.75 | **0.80** |
| | Definitions | 0.9 | 0.96 | 0.32 | 0.66 | 0.54 | **0.98** |
| **Link Retrieval** | Laws | 0.25 | 0.37 | 0.21 | 0.23 | 0.05 | **0.70** |
| **NER** | Organizations | 0.86 | 0.61 | 0.69 | 0.61 | 0.65 | **0.94** |
| | Legislations | 0.96 | 0.94 | 0.92 | 0.96 | 0.98 | **0.98** |
| | Dates | 0.98 | 0.90 | 0.82 | 0.98 | 0.94 | **0.98** |
| | Monetary values | 0.98 | 0.96 | 0.86 | 0.98 | 0.96 | **0.98** |
| | Statistics | 0.96 | 0.84 | 0.67 | 0.94 | 0.86 | **0.98** |

**Table 3: Mean ± standard deviation of the response time of the FinGPT search agent and the baselines in milliseconds (ms).**

| Model | Financial Indices | Web Questions |
|---|---|---|
| **GPT-3.5 Turbo** [28] | **2925.9 ± 1471.5** | - |
| **GPT-4** [26] | 12512.8 ± 4077.3 | - |
| **GPT-4o** [27] | 8720.7 ± 2209.9 | - |
| **Mistral** [17] | 7886.6 ± 1483.2 | - |
| **LLaMA 3.1** [2] | 10223.1 ± 2913.6 | - |
| **LLaMA 3** [2] | 10824.2 ± 1634.2 | - |
| **FinGPT Search Agent (GPT-3.5 Turbo as base)** | **7658.0 ± 1166.2** | 2244.9 ± 526.4 |
| **FinGPT Search Agent (GPT-4o as base)** | 8016.9 ± 1318.8 | **2087.8 ± 717.4** |
| **FinGPT Search Agent (LLaMA 3.1 as base)** | 7768.1 ± 1007.5 | 2509.7 ± 993.4 |

models themselves. A possible reason is that RAG provides relevant context, guiding the model to more efficient response generation.

## 7 Conclusion and Future Work

In this paper, we presented two customized FinGPT search agents: one for individual users and another for institutional users. The individual version focuses on providing tailored financial planning and advice, integrating user preferences, local files, and real-time web data through RAG. The institutional version is designed to analyze proprietary datasets, leveraging continual pretraining, fine-tuning, and RAG to meet the specific needs of financial institutions. Both agents operate locally, ensuring data security. The experimental results demonstrate that both versions of the FinGPT search agents outperform existing LLMs in terms of accuracy and data freshness, indicating their potential for real-world applications.

Future work will explore further optimization of better information retrieval methods, improved compliance with regulatory standards, and enhanced user experiences to make it more user-centric. We will enable the agent to access the website of XBRL International [12], and the website of Common Domain Model [1]. We are integrating FinGPT Search Agent with the Open Financial LLM Leaderboard [2]. Our team are training Open-FinLLMs [34] that can be used as our base model.

## Acknowledgments

Felix Tian, Ajay Byadgi, Daniel Kim, and Xiao-Yang Liu Yanglet acknowledge the support from NSF IUCRC CRAFT center research grant (CRAFT Grant 22017) for this research. The opinions expressed in this publication do not necessarily represent the views of NSF IUCRC CRAFT. Felix Tian and Xiao-Yang Liu Yanglet acknowledge the support from Columbia's SIRS and STAR Program,

[1]Website of CDM at FinOS: https://cdm.finos.org/

[2]Website of Open Financial LLM Leaderboard: https://huggingface.co/spaces/finosfoundation/Open-Financial-LLM-Leaderboard

The Tang Family Fund for Research Innovations in FinTech, Engineering, and Business Operations.

## References


[1] Jina AI. 2024. Jina AI Reader. https://github.com/jina-ai/reader.

[2] AI@Meta. 2024. Llama 3 Model Card. (2024). https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md

[3] Ajay Byadgi. 2024. FinGPT-Individual-Agent-Performance. GitHub repository. https://github.com/AjayByadgi/FinGPT-Individual-Agent-Performance

[4] Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. InPars: Data Augmentation for Information Retrieval using Large Language Models. arXiv:2202.05144 [cs.CL] https://arxiv.org/pdf/2202.05144

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877–1901.

[6] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The Muppets straight out of Law School. *EMNLP* (2020).

[7] Shuohang Chen, Weijia Qi, Xufang Mou, Yuhui Bai, Dawei Wang, Cheng Guo, Huayi Zhang, Chenguang Zhang, Lu Li, Yuxiao Yang, Jianfeng Ma, Ming-Wei Li, Minlie Huang, Jing Gao, and Yong Qi. 2023. MIND2WEB: Towards a Generalist Agent for the Web. In *Advances in Neural Information Processing Systems (NeurIPS) 2023*. https://proceedings.neurips.cc/paper_files/paper/2023/hash/5950bf290a1570ea401bf98882128160-Abstract-Datasets_and_Benchmarks.html

[8] Google Colaboratory. 2024. Fetching and Parsing Local Files.

[9] Tim Dettmers, Menashe Zeno, Jamie Hall, Punit Singh, Qiaozi Yang, Shubho Goel, Sayna Chandra, Omar Khalid, Mike Lewis, Rashid Mehmood, et al. 2024. QLoRA: Efficient Finetuning of Quantized LLMs. *Advances in Neural Information Processing Systems* (2024).

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*.

[11] Andre Esteva, Alexandre Robicquet, Bharath Ramsundar, Volodymyr Kuleshov, Mark DePristo, Kelvin Chou, Claire Cui, Greg Corrado, Jeff Dean, and Lucy Colwell. 2021. A guide to deep learning in healthcare. *Nature Medicine* 27, 1 (2021), 15–25.

[12] Shijie Han, Haoqiang Kang, Bo Jin, Xiao-Yang Liu, and Steve Yang. 2024. XBRL Agent: Leveraging Large Language Models for Financial Report Analysis. In *ACM International Conference on AI in Finance*.

[13] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. WebVoyager: Building an End-to-End Web Agent with Large Multimodal Models. *arXiv preprint arXiv:2401.13919* (2024). https://arxiv.org/abs/2401.13919

[14] Ronny Hoesada. 2023. Enhancing LLM Accuracy Using MongoDB Vector Search and Unstructured.io Metadata. https://www.mongodb.com/developer/products/atlas/llm-accuracy-vector-search-unstructured-metadata/.

[15] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2023. LoRA: Low-rank adaptation of large language models. *International Conference on Learning Representations* (2023).

[16] Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. FinanceBench: A New Benchmark for Financial Question Answering. *arXiv preprint arXiv:2311.11944* (2023).

[17] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023). https://arxiv.org/abs/2310.06825

[18] FINOS Labs. 2024. EMIR-Specific RAG Dataset. https://github.com/finos-labs/emir-specific-rag/tree/main. Accessed: 2024-07-30.

[19] Xiao-Yang Liu, Guoxuan Wang, Hongyang Yang, and Daochen Zha. 2023. Data-Centric FinGPT: Democratizing Internet-scale Data for Financial Large Language Models. In *Workshop on Instruction Tuning and Instruction Following, NeurIPS*.

[20] Xiao-Yang Liu, Ziyi Xia, Hongyang Yang, Jiechao Gao, Daochen Zha, Ming Zhu, Christina Dan Wang, Zhaoran Wang, and Jian Guo. 2024. Dynamic datasets and market environments for financial reinforcement learning. *Machine Learning* 113, 5 (2024), 2795–2839.

[21] Xiao-Yang Liu, Jie Zhang, Guoxuan Wang, Weiqing Tong, and Anwar Walid. 2024. FinGPT-HPC: Efficient Pretraining and Finetuning Large Language Models for Financial Applications with High-Performance Computing. *arXiv preprint arXiv:2402.13533* (2024).

[22] Xiao-Yang Liu, Yimeng Zhang, Yukang Liao, and Ling Jiang. 2023. Dynamic updating of the knowledge base for a large-scale question answering system. *TALLIP* 19, 3 (2023), 1–13.

[23] Xiao-Yang Liu, Rongyi Zhu, Daochen Zha, Jiechao Gao, Shan Zhong, and Meikang Qiu. 2023. Differentially Private Low-Rank Adaptation of Large Language Model Using Federated Learning. *arXiv preprint arXiv:2312.17493* (2023). https://arxiv.org/abs/2312.17493

[24] Youyang Ng, Daisuke Miyashita, Yasuto Hoshi, Yasuhiro Morioka, Osamu Torii, Tomoya Kodama, and Jun Deguchi. 2023. SimplyRetrieve: A Private and Lightweight Retrieval-Centric Generative AI Tool. arXiv:2308.03983 [cs.CL] https://arxiv.org/pdf/2308.03983

[25] Yuqi Nie, Yaxuan Kong, Xiaowen Dong, John M Mulvey, H Vincent Poor, Qingsong Wen, and Stefan Zohren. 2024. A Survey of Large Language Models for Financial Applications: Progress, Prospects and Challenges. *arXiv preprint arXiv:2406.11903* (2024). https://arxiv.org/abs/2406.11903

[26] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023). https://arxiv.org/abs/2303.08774

[27] OpenAI. 2024. Hello GPT-4o. https://openai.com/index/hello-gpt-4o/

[28] Andrew Pen, Michael Wu, John Allard, Logan Kilpatrick, and Steven Heidel. 2023. GPT-3.5 Turbo fine-tuning and API updates. https://openai.com/index/gpt-3-5-turbo-fine-tuning-and-api-updates Accessed: 2024-08-02.

[29] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[30] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint* (2023). arXiv:arXiv:2302.13971

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30 (2017), 30.

[32] Shan Wu et al. 2023. BloombergGPT: A large language model for finance. *arXiv preprint arXiv:2304.15384* (2023).

[33] Qianqian Xie, Weiguang Han, Zhengyu Chen, Ruoyu Xiang, Xiao Zhang, Yueru He, Mengxi Xiao, Dong Li, Yongfu Dai, Duanyu Feng, Yijing Xu, Haoqiang Kang, Ziyan Kuang, Chenhan Yuan, Kailai Yang, Zheheng Luo, Tianlin Zhang, Zhiwei Liu, Guojun Xiong, Zhiyang Deng, Yuechen Jiang, Zhiyuan Yao, Haohang Li, Yangyang Yu, Gang Hu, Jiajia Huang, Xiao-Yang Liu, Alejandro Lopez-Lira, Benyou Wang, Yanzhao Lai, Hao Wang, Min Peng, Sophia Ananiadou, and Jimin Huang. 2024. FinBen: A Holistic Financial Benchmark for Large Language Models. *NeurIPS, Special Track on Datasets and Benchmarks* (2024).

[34] Qianqian Xie, Dong Li, Mengxi Xiao, Zihao Jiang, Ruoyu Xiang, Xiao Zhang, Zhengyu Chen, Yueru He, Weiguang Han, Yuzhe Yang, et al. 2024. Open-FinLLMs: Open Multimodal Large Language Models for Financial Applications. *arXiv preprint arXiv:2408.11878* (2024). https://arxiv.org/abs/2408.11878

[35] Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. 2023. FinGPT: Open-Source Financial Large Language Models. *FinLLM at IJCAI* (2023).

[36] Boyu Zhang, Hongyang Yang, Tianyu Zhou, Muhammad Ali Babar, and Xiao-Yang Liu. 2023. Enhancing financial sentiment analysis via retrieval augmented large language models. In *Proceedings of the fourth ACM international conference on AI in finance*. 349–356.

[37] P Zhao, H Zhang, Q Yu, Z Wang, Y Geng, and F Fu. 2024. Retrieval-augmented generation for AI-generated content: A survey. *arXiv preprint arXiv:2402.19473* (2024).

[38] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. Large Language Models for Information Retrieval: A Survey. *arXiv preprint arXiv:2308.07107* (2023). https://arxiv.org/abs/2308.07107