Quantitative Portfolio Management and Financial Decision-Making with Future Generation of AI Models

by

Ahmadreza Hajaghaie

A thesis submitted to
The Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
June 2025

Thesis advisor

Dr. Ruppa K. Thulasiram

Author

Ahmadreza Hajaghaie

Quantitative Portfolio Management and Financial Decision-Making with Future Generation of AI Models

Abstract

This study investigates Large language models (LLMs) and agentic artificial intelligence (Agentic AI) for their efficacy within the finance domain, particularly focusing on asset allocation, portfolio management, and financial decision-making. Portfolio construction is a complex task that requires careful consideration of multiple asset classes, risk profiles, market conditions, and user-inclusive investment objectives. LLMs and pre-trained models often hallucinate due to their lack of access to real-time and updated information. Additionally, they are not particularly effective at mathematical reasoning. While LLMs exhibit remarkable capabilities in understanding financial terminology and reasoning over structured or unstructured data, they face critical limitations in their access to real-time financial information and market dynamics, which are essential for making informed investment decisions. To address these challenges, our research explores these problems through a three-phase approach: first, the application of advanced prompt engineering techniques; second, the implementation of retrieval-augmented generation (RAG); and finally, the development of a multi-agent system architecture. We show through this research (i) a strong contribution to the intersection of AI and financial decision-making; (ii) that LLMs and agents could act as catalysts for one another rather than relying solely on pre-trained LLMs. Therefore, enhanced performance and more complex execution can be achieved by integrating external agents; (iii) that optimizing the quality and reliability of model-generated responses is possible; (iv) improving the overall effectiveness of financial decision-making processes and portfolio management. As an evaluation mechanism we have done a case-study of comparison with *Wealthsimple* portfolio classifications by replicating their asset class structure appropriately through our agentic workflow.

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I want to thank my family for always being there for me throughout this journey. Especially to my amazing sister, the one who always had my back, supported my decisions (even the crazy ones), and cheered me on no matter what. I truly couldn't have done this without your love and encouragement.

To my friends, lab mates, and everyone who patiently listened to me complain, stress, and obsess over every little detail, thank you for putting up with me! Your support, advice, and just being there to hear me out (again and again) made this process so much easier and way more enjoyable.

And last but absolutely not least, the biggest thank you goes to my advisor, Dr. Tulsi. Thank you for your care, guidance, and constant support throughout this research. I'm genuinely grateful for your patience, encouragement, and for always steering me in the right direction. Couldn't have asked for a better mentor!

To my family

هر لحظه که تسلیمم در کارگه تقدیر   آرام تر از آهو بی باک تر از شیرم

هر لحظه که می کوشم در کار کنم تدبیر   رنج از پی رنج آید زنجیر پی زنجیر

Chapter 1

Introduction

The world of finance revolves around fast, accurate, and well-informed decision-making. Before the advent of computers and artificial intelligence, much of the work in the financial sector was performed manually. While these manual processes were often precise, they were time-consuming, labor-intensive, and lacked efficiency, making it difficult to keep pace with the complex finance market.

With the rise of the internet, global connectivity, and real-time data flows, staying informed and reacting quickly to market dynamics became even more challenging. This is particularly critical in portfolio management, where professionals must continuously monitor global events, economic indicators, and geopolitical news, all of which can significantly influence market performance and risk exposure. As technology advanced and we transitioned into the era of computational intelligence, the scale and speed of financial markets expanded exponentially. With these advances, being able to rapidly adjust investment strategies in response to these factors is essential for effective risk management and successful portfolio optimization.

Over the years, many theories, models, and architectures have been designed to support and enhance financial decision-making. However, the rapid pace of advancements driven by modern AI technologies and the emergence of exotic financial products have introduced a new wave of tools and methodologies, fundamentally reshaping the landscape of finance. In particular, large language models (LLMs) and agent-based approaches under the broad category of natural language processing (NLP) are now emerging as powerful solutions for navigating the complexities of today's volatile and information-rich markets.

To keep up with these advancements in both finance and NLP, there is a growing need to adopt these cutting-edge technologies. Leveraging LLMs, retrieval-augmented generation (RAG), and agentic approaches can significantly enhance decision-making processes, allowing for faster, more informed, and more adaptive financial strategies.

In this research, we have studied this new generation of AI technologies in the context of asset allocation & portfolio management and have done a comprehensive evaluation of these technologies. Our study explored and identified how LLMs and agentic frameworks could be integrated to enhance the quality, speed, and robustness of financial decision-making. We began by assessing the baseline performance of vanilla LLM models using advanced prompt engineering techniques. In the next phase, we augmented these models with RAG to enhance their access to real-time, up-to-date financial data. Finally, we designed and implemented a multi-agent framework tailored for asset allocation and portfolio construction, enabling coordinated portfolio management and modular reasoning across different agents.

Through this multi-phase approach, our research demonstrates how the integration of LLMs, RAG, and agentic systems can lead to more effective, informed, and responsive portfolio management in the fast-moving world of modern finance.

1.1 Overview of Portfolio Management and Asset Allocation

Portfolio management stands as one of the fundamental pillars of modern finance. A portfolio is a collection of financial instruments such as stocks, bonds, cryptocurrencies, and other asset classes. At its core, it is a decision-making process centered on the strategic allocation of investments in various sectors and asset classes to optimize the trade-off between risk and return. The primary objective of portfolio management is to construct an investment portfolio that is tailored to an investor's specific financial goals, risk tolerance, investment horizon, and any additional constraints such as ethical considerations or liquidity needs.

This process typically involves three key stages:(i) the careful selection of assets; (ii) the assignment of appropriate weights to each asset; and (iii) the ongoing rebalancing of the portfolio to maintain alignment with the investor's evolving objectives and market dynamics. Financial instruments incorporated into a portfolio can span a diverse array of categories, including equities, fixed income securities (bonds), commodities, exchange-traded funds (ETFs) [1], real estate, and cryptocurrencies.

Effective portfolio management is inherently dynamic, requiring continuous assessment and adjustment in response to changing economic conditions, asset performances, and investor circumstances. By systematically diversifying investments across many asset classes and sectors, portfolio managers seek to enhance returns while mitigating unsystematic risk [2] that are emanating from a specific sector or an industry. This principle of diversification, first formalized through Modern Portfolio Theory (MPT) by Markowitz [1], remains a cornerstone concept in contemporary portfolio construction practices.

Diversification and risk management have long been critical components of portfolio construction. Statman [6] provided a comprehensive review of these principles in his 1987 study. However, the financial markets have since evolved significantly, expanding beyond traditional equities and bonds. Today, new asset classes such as cryptocurrencies and a wide range of specialized ETFs, including those focused on digital assets and venture capital strategies, have emerged. This evolving landscape necessitates a new wave of adaptation in portfolio management practices to effectively address broader market dynamics and maintain optimal risk-return profiles.

Several frameworks and theories have historically shaped the landscape of portfolio construction. The MPT, introduced by Harry Markowitz in 1952 [1], established the concept of the efficient frontier (Figure 1.1), demonstrating how investors can achieve optimal returns for a given level of risk through diversification. Following this, the Capital Asset Pricing Model

[1] An Exchange-Traded Fund (ETF) is a marketable security that tracks an index, commodity, or a basket of assets, and was first conceptualized by Nathan Most and Steven Bloom to provide low-cost, flexible investment vehicles [5]

[2] This type of risk is just focused on a certain industry or sector

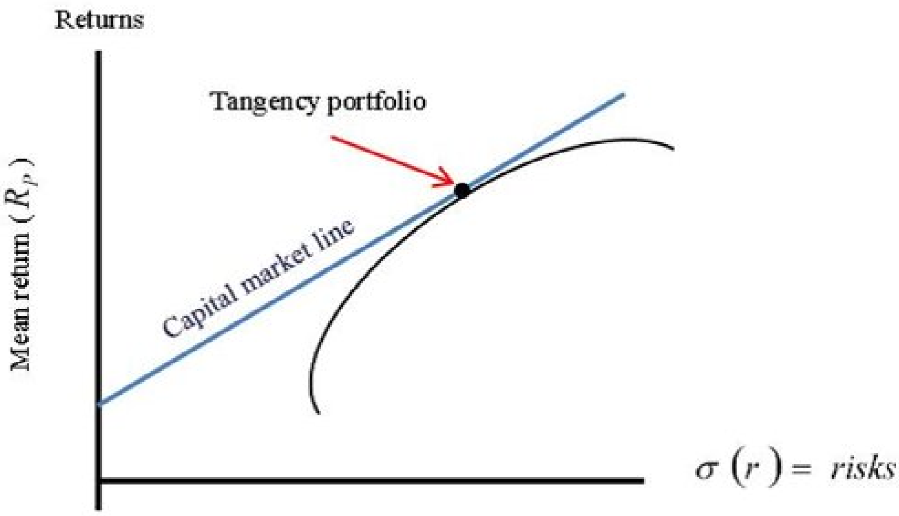


Figure 1.1: Modern Portfolio Theory by Harry Markowitz [1]

(CAPM) [3], introduced by William F. Sharpe [7], provided a systematic approach for evaluating expected returns based on market risk (beta) and the risk-free rate of return. Over time, extensions such as the Fama-French Three-Factor Model [8] and other multifactor models have further refined these theories, incorporating factors like size, value, momentum, and profitability. In this research, we adhere to the foundational principles of the CAPM for portfolio management and asset allocation. While CAPM provides a robust framework for understanding the relationship between risk and expected return, it is important to recognize that in more extensive and complex investment environments, additional factors beyond CAPM, such as multifactor models and alternative risk measures, may need to be considered. However, the exploration of such expanded frameworks lies beyond the scope of this study.

1.2 Relevance and Challenges in Financial Portfolio Management

Asset allocation and portfolio management remain essential backbones for achieving financial objectives. The relevance of financial portfolio management lies in the delicate balance required when constructing a portfolio tailored to a client's specific needs. These needs may encompass factors such as investment time horizon, liquidity requirements, risk tolerance, and

[3]See Chapter 2.1.2 for formal definition of CAPM 2

other customized constraints.

An effective portfolio must optimize the trade-off between risk and return, aiming to either maximize returns for a given level of risk or minimize risk for a targeted return level. Proper portfolio management enables individuals, corporations, and fund managers to navigate the uncertainties and volatilities inherent in financial markets while adhering to predefined investment strategies and constraints. In this context, disciplined asset allocation and continuous risk management are critical for sustaining long-term financial goals amidst evolving market dynamics.

The financial markets have evolved considerably, becoming increasingly globalized and interconnected. As a result, the complexity of investment environments has also grown substantially. New asset classes, evolving market conditions, regulatory changes, and the proliferation of alternative investment vehicles have intensified the demands placed on portfolio managers. In the contemporary landscape, successful portfolio construction requires not only a strong foundation in traditional and advanced models but also the capacity to adapt to real-time financial data, alternative data sources, and emerging market risks.

Financial portfolio management is currently facing numerous challenges:

1. **Information overload:** One of the major challenges in contemporary financial portfolio management is information overload. The volume of data and news generated across global markets has expanded exponentially, making it increasingly difficult for investors and portfolio managers to monitor, process, and interpret all relevant information in real time. Amidst this abundance of information, extracting valuable insights from noise has become a significant problem. The manual extraction and processing of relevant data for informed decision-making are no longer feasible at the scale and speed required by modern financial markets.

2. **Volatility and Complexity:** Another significant challenge is the heightened uncertainty and volatility in today's financial markets. Systematic risks, along with unpredictable factors such as geopolitical events (such as trade disputes, war etc.), regulatory shifts (such as new tariffs, etc.), and rapid technological developments (such as advanced computer

architectures, machine learning (ML), AI, etc.) have introduced major disruptions to market conditions. This increased volatility demands that portfolio managers remain flexible, aware, and continuously responsive in calibrating the models and adjusting portfolio allocations to preserve alignment with investment objectives.

3. **Biases and Human Limitations:** Human decision-makers are susceptible to cognitive biases such as herd behavior, loss aversion, and overconfidence. These biases can adversely impact portfolio construction and risk management decisions.

4. **Adaptation to New Technologies:** The rise of Deep Learning (DL), machine learning (ML), and NLP introduces powerful tools for portfolio optimization. However, integrating these technologies into traditional financial workflows presents its own challenges, including model interpretability, regulatory compliance, and data security.

These challenges necessitate the adaptation of portfolio management practices to the new wave of AI technologies. To navigate the increasing dynamism and complexity of financial markets, it is essential to develop more intelligent, adaptive, and scalable solutions. While certain major systematic risks and unforeseen market disruptions remain difficult to predict, even with the most advanced systems, adopting AI-driven approaches can significantly enhance portfolio management capabilities. In this research, we address these emerging issues by proposing novel solutions that integrate LLMs with advanced techniques such as RAGs and multi-agent frameworks. These approaches provide pathways to mitigate traditional challenges, address new constraints, and offer a competitive edge in today's increasingly complex financial landscape.

1.3 Introduction to Large Language Models (LLMs) and AI-driven finance

In recent years, the interconnection between AI and finance has grown substantially, driven by the increasing complexity of markets and the overwhelming volume of available data. As a result, major enterprises and financial institutions are actively investing in the development of data-driven, intelligent, and scalable decision-making systems. These systems address the limitations of traditional financial models by leveraging advanced AI techniques to enhance adaptability, accuracy, and real-time responsiveness in portfolio management and other financial applications.

One of the most revolutionary advancements in AI in recent years has been the emergence of LLMs. Initially introduced by Radford et al. [9] with generative pre-trained transformer (GPT)-1, these models are founded on the transformer architecture proposed by Vaswani et al. [2]. LLMs are deep neural networks trained on massive text corpora and exhibit remarkable capabilities in understanding, generating, and reasoning over natural language. The transformer architecture itself marked a significant shift in NLP, enabling scalable parallelization and improved performance across a wide range of language tasks from multiple technology industries, as captured in Figure 1.2. These foundational advancements have laid the groundwork for applying LLMs in complex domains such as financial portfolio management. A brief chronological development of transformer-based LLMs is presented in Appendix A.5.

Figure 1.2: Some of the main LLM models in the market, from left to right: ChatGPT from OpenAI, Claude from Anthropic, LLaMA from Meta, and Gemini from Google

Models such as ChatGPT by OpenAI [10], Claude by Anthropic [11], Gemini by Google [12], and LLaMA by Meta [3] illustrate the breadth of general-domain capabilities offered by frontier LLMs. In parallel, domain-specific LLMs have been developed to address the specialized demands of financial language and context. Notably, BloombergGPT [13] was trained on a large corpus of financial documents to support tasks like sentiment analysis and market forecasting, while InvestLM [14], based on LLaMA-65B, was instruction-tuned specifically for investment-focused applications, offering improved performance on finance-related benchmarks.

Since the emergence of LLMs, researchers have extensively investigated their applications in various financial tasks, including sentiment analysis, portfolio construction, asset management, and optimization. Traditional approaches such as CAPM often struggle to cope with the increasing complexity and volatility of financial markets, particularly in interpreting rapidly shifting market sentiment [15]. These challenges have prompted a shift toward more adaptive, intelligent systems capable of learning from diverse data sources. Few-shot learning has emerged as a particularly promising technique in this context. Wang et al. [16] demonstrated the efficacy of few-shot learning in enabling language models to generalize from limited examples. Brown et al. [17] further explored the potential of LLMs using few-shot prompting, showcasing their capacity to perform complex reasoning tasks without task-specific fine-tuning. Additionally, Lewis et al. [18] introduced RAG, a hybrid framework that enhances LLM performance by incorporating relevant external information into the generation process.

1.4 Research objectives

The primary objective of this research is to investigate the effectiveness of LLMs in the context of financial portfolio management, with a focus on improving asset allocation strategies through natural language understanding and reasoning. Given the growing complexity and volatility of modern financial markets, traditional portfolio construction methods often struggle to process large volumes of unstructured data, adapt to rapid market changes, or provide personalized investment strategies. This study aims to address these limitations by evaluating how LLMs can be enhanced and directed using advanced prompting techniques, data augmentation through RAG, and the deployment of a coordinated multi-agent architecture. By systematically comparing different LLM configurations, this research seeks to quantify their decision quality, consistency, and ability to generate well-balanced portfolios under realistic conditions.

Another very important objective is to explore futuristic agentic AI models for studying the feasibility of integrating modular agents, each assigned a specific task such as user profiling, data retrieval, metric computation, portfolio generation, and reporting, within a unified, LangGraph-based (see Chapter 5) pipeline. This agentic framework is designed to delegate responsibilities, improve interpretability, and increase scalability across the portfolio construction process. Additionally, the study evaluates how combining LLM reasoning with real-time financial information can mitigate known limitations such as hallucinations and mathematical errors.

Ultimately, the goal is to demonstrate that LLMs, when guided through structured inputs and supplemented with external data, can serve as intelligent decision-support systems in finance, capable of generating accurate, diverse, and ethically-aligned investment portfolios.

1.5 Thesis structure

This thesis is organized into several Chapters, each building upon a three-phase research framework as presented in Figure 1.3. At the core, the work is structured around the following sequential phases:

1. **Evaluation of LLMs for Portfolio Construction (Chapter 3):** This phase investigates the performance of eight frontier LLMs for asset allocation and portfolio management using an advanced few-shot prompt engineering technique.

2. **Enhancement with RAG (Chapter 4):** In this phase, LLMs are enhanced with external, up-to-date financial data using a RAG architecture, aimed at improving accuracy, contextual understanding, and reducing hallucinations.

3. **Deployment of a Multi-Agent Framework (Chapter 5):** The final phase integrates LLMs into a modular multi-agent system built with LangGraph, enabling intelligent, stateful, and explainable portfolio generation through agent collaboration.

Each of these phases is discussed in detail across the subsequent Chapters, supported by experimental evaluations, comparative analyses, and technical insights that demonstrate the viability and effectiveness of LLM-based approaches in financial portfolio construction, the main contributions to the field from this thesis research work.

Chapter 1 is the introduction, and in this Chapter, we provide the background and motivation for the study, introduce the research problem, outline the objectives, and briefly describe the contributions of the work. Chapter 2 presents a comprehensive review of the existing body of research relevant to this thesis. It begins with an overview of traditional portfolio management approaches, including foundational theories such as MPT and both quantitative and qualitative methods used in asset allocation. The Chapter then explores the integration of AI in finance, highlighting its current applications as well as known challenges, such as hallucinations and mathematical inconsistencies in model outputs.

Following this, the Chapter delves into LLMs, covering their transformer-based architecture, notable models such as GPT, LLaMA, and Claude, and their specific capabilities and limitations in financial contexts. The concept of RAG is also examined, emphasizing its role in enhancing LLM performance with external data sources and reviewing prior research in RAG-enhanced portfolio construction. Finally, the Chapter introduces multi-agent systems and frameworks, discussing agent-based design principles, the model context protocol (MCP) [19], and leading platforms including LangChain[4], LangGraph[5], Hugging Face[6], CrewAI[7], and OpenAI [20]. This literature review identifies the gaps and unresolved challenges in the field, which this thesis aims to address through its multi-phase research framework.

Chapter 3 presents the first phase of the study, where eight state-of-the-art LLMs were evaluated for asset allocation and portfolio management using an advanced few-shot prompting technique. Chapter 4 builds upon this foundation by introducing RAG as a technique to enhance LLM performance through integration with real-time financial data. Research methods and results from these two Chapters have been presented at conferences.

Chapter 5 introduces a novel multi-agent architecture for portfolio management that integrates LLM-based and function-based agents using LangGraph for stateful orchestration. It details the system design, agent roles, data pipelines, and workflow logic that collectively enable intelligent, modular, and adaptive portfolio construction aligned with investor profiles and

[4] https://www.langchain.com/
[5] https://www.langchain.com/langgraph
[6] https://huggingface.co/
[7] https://www.crewai.com/

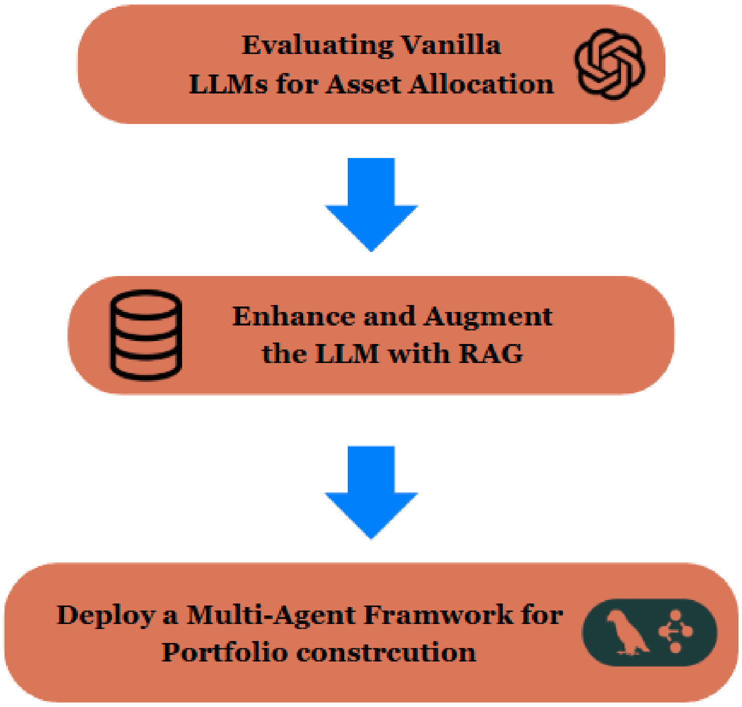


Figure 1.3: Thesis Structure

real-time financial data.

Lastly, Chapters 6 and 7 synthesize the findings from all previous phases. In Chapter 6, a comparative analysis is done to evaluate the effectiveness of each approach, including few-shot LLM prompting, RAG-enhanced models, and the multi-agent framework, highlighting their relative strengths in portfolio performance, risk management, and adaptability. Chapter 7 concludes the thesis by outlining the key contributions to the field, discussing limitations, and suggesting directions for future research.

Chapter 2

Background and Literature Review

This Chapter presents the theoretical and technical foundations upon which the thesis research is built. It begins by reviewing traditional approaches to portfolio management, with a focus on modern portfolio theory (MPT) [1] and its subsequent extensions. It then explores the growing role of AI in finance, highlighting both the capabilities and limitations of current AI-driven systems. Special attention is given to LLMs, their underlying transformer architecture, and their application in financial domains. The Chapter also reviews retrieval augmented generation (RAG) as a technique to enhance LLM performance using external data sources. Finally, it discusses multi-agent systems and their emerging relevance in decision-making frameworks, particularly within the context of agentic AI portfolio construction.

2.1 Traditional Portfolio Management

2.1.1 Modern Portfolio Theory

Modern Portfolio Theory (MPT), introduced by Harry Markowitz [1], remains one of the foundational pillars of financial portfolio construction. At the time of its inception, financial markets were relatively simpler, with limited data availability and less globalization. In contrast, today's markets are significantly more complex, highly interconnected across geographies, and composed of a broader array of asset classes, many of which exhibit higher volatility and behavior that classical models like MPT cannot fully capture [1].

While MPT continues to serve as a valuable benchmark in academic research, its assumptions, such as normally distributed returns, investor rationality, and market efficiency, are often too restrictive for real-world financial environments. Large-scale portfolio management in institutional settings, where vast capital is allocated across diverse and dynamic markets, requires more comprehensive frameworks. These include extensions of MPT and alternative theories that incorporate behavioral finance, macroeconomic variables, multifactor risk models, and tail risk considerations.

In this thesis, MPT is used as a reference point for portfolio construction and evaluation. However, it is acknowledged that enterprise-level applications demand more robust methodologies that account for a wider range of influential factors beyond market risk and basic diversification.

MPT forms the foundation of modern investment theory. The core idea of MPT is that an investor can construct an optimal portfolio by maximizing expected return for a given level of risk, or equivalently, minimizing risk for a given expected return, through diversification.

Let a portfolio consist of n assets. The weight of each asset in the portfolio is denoted as w_i, where $i \in \{1, 2, \ldots, n\}$ and $\sum_{i=1}^{n} w_i = 1$.

Expected Return: The expected return of the portfolio, denoted by R_p, is computed as a weighted sum of the expected returns of the individual assets:

$$R_p = \sum_{i=1}^{n} w_i \mu_i = \mathbf{w}^\top \boldsymbol{\mu} \tag{2.1}$$

where μ_i is the expected return of asset i, w_i is the weight coefficient of expected return, $\boldsymbol{\mu}$ is the vector of expected returns, and $\mathbf{w}$ is the weight vector.

Portfolio Variance and Risk: The risk or variance of the portfolio's return, denoted by σ_p^2, accounts for both the variances of individual assets and their covariances:

[1] This section contains many acronyms and definitions from finance methodology, check A.6 Appendix for more information.

$$\sigma_p^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij} = \mathbf{w}^\top \mathbf{C} \mathbf{w} \tag{2.2}$$

Here, σ_{ij} is the covariance between asset i and j, and C is the $n \times n$ covariance matrix of asset returns.

Efficient Frontier: The set of all portfolios that offer the highest expected return for a given level of risk constitutes the efficient frontier. These portfolios are considered optimal in MPT (refer to Figure 1.1).

Optimization Problem: To determine the optimal portfolio, one solves the following quadratic programming problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^\top \Sigma \mathbf{w} \\ \text{subject to} \quad & \mathbf{w}^\top \boldsymbol{\mu} = R_t \\ & \sum_{i=1}^{n} w_i = 1 \\ & w_i \geq 0, \quad \forall i \quad \text{(no short selling)} \end{aligned} \tag{2.3}$$

where R_t is the target expected return [2].

Sharpe Ratio: Developed by William Sharpe, the Sharpe ratio [21] is often used to evaluate the performance of a portfolio relative to its risk:

$$\text{Sharpe Ratio} = \frac{R_p - R_f}{\sigma_p} \tag{2.4}$$

where R_f is the risk-free rate of return.

MPT operates under key assumptions, including investor risk aversion and market efficiency, conditions that do not always hold true in real-world financial environments. Nevertheless, it remains one of the most influential frameworks in the field of finance and has laid the ground-

[2] Short selling refers to the practice of selling a borrowed security with the expectation of buying it back later at a lower price to profit from a decline in its value.

work for numerous advanced models. These include the CAPM [7] and various multifactor extensions [8], which will be discussed in the following section.

2.1.2 Other Theories and Models in Portfolio Construction

While MPT remains foundational in finance, numerous alternative models and extensions have emerged to address its limitations and reflect more realistic investor behavior and market dynamics. These models often incorporate additional risk factors, behavioral insights, or more complex optimization techniques to improve upon the assumptions and practical shortcomings of MPT.

Capital Asset Pricing Model (CAPM): The CAPM, developed by Sharpe, Lintner, and Mossin [7], builds upon MPT by introducing a relationship between an asset's expected return and its systematic risk, measured by beta (β). The CAPM equation is:

$$E(R_i) = R_f + \beta_i \left(E(R_m) - R_f\right) \tag{2.5}$$

where $E(R_i)$ is the expected return of asset i, R_f is the risk-free rate, $E(R_m)$ is the expected return of the market, and β_i is the asset's sensitivity to market returns. CAPM assumes a single-factor model and efficient markets, but these simplifying assumptions have led to mixed empirical validation.

Fama-French Three-Factor and Five-Factor Models: Fama and French [8] extended the CAPM by introducing additional factors that capture size and value effects in stock returns:

$$E(R_i) = R_f + \beta_m(R_m - R_f) + \beta_s \text{SMB} + \beta_v \text{HML} \tag{2.6}$$

where SMB (Size premium, Small Minus Big: return of small-cap stocks minus large-cap stocks) accounts for the size premium and HML (Value premium, High Minus Low: return of high book-to-market stocks minus low book-to-market stocks). Later, the model was expanded into a five-factor version by adding profitability and investment factors. These models have shown better explanatory power over cross-sectional asset returns, particularly in equity markets [8; 22]. The Fama-French Five-Factor Model [22] is expressed as:

$$E(R_i) - R_f = \beta_i(E(R_m) - R_f) + s_i\text{SMB} + h_i\text{HML} + r_i\text{RMW} + c_i\text{CMA} \tag{2.7}$$

Where $E(R_i)$ is the expected return of asset i, R_f is the risk-free rate, $E(R_m) - R_f$ is the market risk premium, β_i is the sensitivity to market risk, SMB is the size premium (Small Minus Big), s_i is the sensitivity to the size premium, HML is the value premium (High Minus Low), h_i is the sensitivity to the value premium, RMW is the profitability premium (Robust Minus Weak), r_i is the sensitivity to the profitability premium, CMA is the investment premium (Conservative Minus Aggressive), and c_i is the sensitivity to the investment premium.

Arbitrage Pricing Theory (APT): Proposed by Stephen Ross [23], arbitrage pricing theory relaxes the assumptions of CAPM by modeling expected returns as a linear combination of multiple macroeconomic factors:

$$E(R_i) = R_f + \sum_{k=1}^{K} \beta_{ik} F_k \tag{2.8}$$

where β_{ik} is the sensitivity of asset i to factor k, and F_k is the risk premium associated with factor k. APT does not require a market portfolio and is theoretically attractive due to its flexibility, although its practical application depends heavily on identifying relevant risk factors.

Black-Litterman Model: The Black-Litterman model, developed by Black and Litterman [24] at Goldman Sachs, combines the equilibrium market returns from CAPM with investor views to produce more stable and intuitive portfolio allocations. It overcomes some of the instability and sensitivity issues found in mean-variance optimization by incorporating subjective information in a Bayesian framework [24].

Behavioral Portfolio Theory (BPT): Behavioral portfolio theory, introduced by Shefrin and Statman [25], incorporates psychological and cognitive factors into the portfolio construction process. It challenges the assumptions of rationality and utility maximization and suggests that investors mentally frame portfolios into layers corresponding to different goals (e.g., security, aspiration). BPT attempts to explain investor behavior that deviates from MPT predictions, such as preference for lottery-like assets or overconcentration.

Risk Parity: Risk parity is a modern asset allocation strategy introduced by Qian [26], that seeks to allocate risk equally across all assets in the portfolio, rather than capital. In its simplest form, it adjusts weights so that each asset contributes equally to the portfolio's overall volatility. This approach has gained popularity for producing more balanced and diversified portfolios, especially in multi-asset class contexts.

Machine Learning and AI-driven Models: More recently, portfolio construction has begun to leverage AI and ML techniques. These include optimization using reinforcement learning (RL), deep neural networks (DNN) for forecasting, and NLP for sentiment-based asset selection. AI-driven models often outperform traditional approaches in capturing nonlinear dependencies, incorporating unstructured data, and adapting to changing market environments [27] [28].

These new theories and models demonstrate the evolution of portfolio construction methods in response to the complexity and inefficiencies of real-world markets.

2.2 Artificial Intelligence in Finance

This Chapter discusses the applications of AI in finance with a quick overview.

2.2.1 Overview

Given the increasing complexity of global financial markets, many researchers have sought to expand and apply ML and DL algorithms in finance. These methods have introduced improvements not only in operational efficiency but also in the accuracy and optimization of financial decision-making processes. ML and DL techniques have been widely adopted for forecasting market trends, optimizing asset allocation, synthesizing financial indicators, and supporting a range of data-driven investment strategies.

The adoption of these AI-driven methods spans a wide range of financial applications, including algorithmic trading, credit scoring, fraud detection, robo-advisory services, and risk assessment. Financial institutions increasingly rely on supervised and unsupervised learning models to analyze large volumes of structured and unstructured data, extract meaningful patterns, and make predictive inferences. NLP, a subfield of AI, has also become instrumental in interpreting financial news, earnings reports, and market sentiment, enhancing both strategic and tactical portfolio decisions.

AI has become an increasingly integral component of the financial services industry, offering data-driven tools and methods that enhance decision-making, automate operations, and improve predictive analytics. AI technologies, including ML, DL, NLP, and reinforcement learning, are now being applied across a wide spectrum of financial tasks, ranging from fraud detection and credit scoring to algorithmic trading and portfolio optimization.

In the context of investment management, AI has shown particular promise in improving both the speed and accuracy of analysis. Traditional investment approaches often rely on structured numerical data and human judgment, which can be limited by information-processing capacity and susceptibility to bias. In contrast, AI systems can efficiently analyze massive volumes of both structured and unstructured data, such as market prices, economic indicators, news articles, earnings reports, and social media sentiment, to uncover hidden patterns, assess risk exposures, and identify profitable opportunities.

2.2.2 Popular AI Applications in Finance

Some of the most common AI applications in finance include:

- **Algorithmic Trading:** AI-powered algorithms can execute high-frequency trades, adapt to real-time market conditions, and optimize trading strategies based on evolving data. Théate and Ernst [29] introduce the Trading Deep Q-Network (TDQN), a deep reinforcement learning-based (DRL) algorithm tailored for algorithmic trading, aiming to maximize the Sharpe ratio by learning optimal trading positions through artificially generated trajectories from limited historical stock data. It also proposes a more rigorous evaluation framework to objectively assess trading performance. Li et al. [30] propose a deep reinforcement learning-based trading agent that integrates stacked denoising autoencoders (SDAEs) and long short-term memory (LSTM) networks to autonomously extract features and make trading decisions in dynamic financial markets, outperforming baseline strategies in both stock and futures trading. Huang [31] treats financial trading as a Markov decision process and applies DRL to train agents that can learn optimal trading strategies. It introduced a framework where agents trade multiple assets while handling transaction costs and market impact [31].

- **Credit Risk Assessment:** ML models can evaluate creditworthiness using alternative data sources beyond traditional credit scores, including transaction history, online behavior, and social signals. Sababipour et al. [32] compared deep learning approaches for credit score prediction and found that a hybrid LSTM-GRU model outperformed both standalone LSTM, GRU, and traditional methods, achieving superior accuracy and predictive power on real-world credit history data.

- **Fraud Detection and Compliance:** AI is used to monitor transaction anomalies and detect suspicious behavior, helping institutions maintain regulatory compliance and reduce financial crime. Raghavan and El Gayar [33] comprehensively evaluated recent ML and DL techniques for fraud identification. It discusses how these algorithms can effectively identify fraudulent transactions, minimize false positives, and maintain high precision and scalability.

- **Financial Forecasting:** Many ML and DL algorithms are applied to predict asset prices, volatility, or macroeconomic indicators, offering more dynamic and flexible forecasting tools. Kingdon reviewed this matter thoroughly, explored the automation of financial time series forecasting using intelligent systems. It delves into minimizing human intervention in modeling financial data and addresses the challenges of setting parameters for neural networks and genetic algorithms [34]. Also, Sababipour et al. used transformers architecture to forecast the stock volatility [35]. Dip Das et al. [36] introduce a novel AE-GRU algorithm for financial asset price prediction. The findings indicate that the proposed encoder–decoder-based GRU architecture outperforms existing models in financial time-series forecasting [36]. Evolutionary computing techniques such as genetic algorithms, ant brood clustering [37] have been used in the past for Value-at-Risk [38] analysis and portfolio diversification in finance.

- **Portfolio Management:** AI enhances asset allocation by integrating non-linear optimization, scenario analysis, and real-time adaptation based on dynamic inputs and risk preferences. Our previous A part of this thesis research using RAGs has indicated that LLMs can outperform the market average and enhance the decision-making process in portfolio management [28], which is discussed comprehensively in Chapter 4. Jiang et al. [39] introduced a novel, deep reinforcement learning framework for cryptocurrency portfolio management, utilizing an ensemble of identical independent evaluators (EIIE) topology, portfolio-vector memory (PVM), online stochastic batch learning (OSBL), and an explicit reward function. Zhang et al. [40] propose a deep learning framework that directly optimizes the portfolio Sharpe ratio by adjusting model parameters, eliminating the need for explicit asset return forecasting, and constructs portfolios using exchange-traded funds of market indexes, which simplifies the asset selection process due to the robust correlations among these indexes.

2.3 Large Language Models (LLMs)

LLMs represent one of the most significant advancements in the field of NLP and neural networks. They are primarily built upon the transformer architecture, introduced by Vaswani et al. [2] and developed by teams at Google and DeepMind. Later on they were enhanced by Mixture-of-Experts and some other techniques to increase the efficiency. LLMs were first introduced by Radford et al. [9] with the release of generative pretrained transformer or GPT-

1, marking the beginning of a new era in language modeling. These models are capable of a wide range of tasks, including natural language understanding, text generation, summarization, reasoning, and even code synthesis. In the case of multimodal variants, they can also process and generate other data types such as images. LLMs are typically characterized by their parameter count and the size of their training datasets; many of the latest models now exceed tens or hundreds of billions of parameters and are trained on trillions of tokens.

LLMs are pre-trained on vast and diverse corpora of textual data, including Wikipedia articles, web pages, books, financial reports, and technical documentation. This extensive training enables them to generate human-like, contextually aware, and professionally coherent responses based on user input. Their ability to generalize across domains makes them particularly valuable for tasks requiring language understanding and generation, even in specialized fields such as finance.

LLMs are controlled by a few major output settings, the major ones are "Temperature", "Top_p", "Top_k", and "Max Tokens".

1. Temperature: temperature is a parameter used to control the randomness of token selection in language model generation. It modifies the probability distribution before sampling. The adjusted (normalizing) softmax function is given by:

   $$P(x_i) = \frac{e^{z_i/\tau}}{\sum_j e^{z_j/\tau}} \tag{2.9}$$

   Here, z_i are the logits (raw output) from the model, and τ is the temperature value.

   When $\tau = 1$, the distribution remains unchanged.

   When $\tau < 1$, the output becomes more deterministic.

   When $\tau > 1$, the output becomes more random and diverse.

   Lower temperature is preferred for factual or structured tasks; higher temperature is used to encourage creativity or variation.

2. Top_p: Top-p sampling restricts generation to the smallest set of tokens V_p whose cumulative probability exceeds a threshold p:

$$\sum_{x_i \in V_p} P(x_i) \geq p \tag{2.10}$$

This dynamic approach balances diversity and relevance by adapting to the shape of the probability distribution.

3. Top_k: Top-k sampling restricts token selection to the k most probable tokens:

$$V_k = \{x_1, x_2, \ldots, x_k\} \quad \text{where} \quad P(x_1) \geq P(x_2) \geq \cdots \geq P(x_k) \tag{2.11}$$

These terms are formally defined in Chapter 3.1 (See 3) This fixed-size subset gives more control over randomness. Smaller k values lead to focused, predictable outputs.

4. Max Tokens: The `max_tokens` parameter sets an upper limit on the number of tokens the model is allowed to generate:

$$\text{Total tokens} \leq T_{\max} \tag{2.12}$$

This constraint helps control response length and ensures runtime and resource efficiency.

Prior to 2024, the capabilities of LLMs were largely dependent on the structure and quality of user instructions, primarily delivered through prompting techniques such as zero-shot, few-shot, and chain-of-thought (CoT) prompting. In zero-shot prompting, the user poses a direct question or task without providing examples, relying on the model's general knowledge. In contrast, few-shot and CoT prompting involve supplying exemplars or step-by-step reasoning paths to guide the model toward more accurate and contextually aligned responses.

However, with the introduction of advanced models such as ChatGPT-o1 [41] and DeepSeek-R1 (DeepSeek AI Team, Guo, and many others [42]), LLMs have increasingly been enhanced with reinforcement learning from human feedback (RLHF). These advancements have significantly improved the models' ability to interpret, reason over, and autonomously align with user intent, even with minimal instruction. While explicit prompting is still beneficial, the burden of extensive instruction has been reduced, marking a shift toward more intuitive and intelligent interaction paradigms.

In the context of finance, LLMs are increasingly being applied to automate a wide range of routine and knowledge-intensive tasks. These include financial sentiment analysis, earnings report summarization, regulatory document review, fraud detection, and portfolio optimization [43; 44; 45]. The ability of LLMs to process both structured and unstructured data with high linguistic accuracy makes them well-suited for supporting analytical workflows across financial services.

Several recent studies have focused on fine-tuning and training LLMs specifically for financial applications. One of the most notable efforts is BloombergGPT, introduced in 2023 by Bloomberg [13], which was trained predominantly on financial data, including proprietary datasets. While BloombergGPT demonstrated strong performance on financial benchmarks and domain-specific queries, its utility is limited by the static nature of its training data. Without access to external knowledge sources or real-time web APIs, the model can become outdated in rapidly evolving market conditions. Other notable models in this domain include InvestLM and FinGPT, which were both designed to address financial tasks with a more targeted training approach [14] [46].

InvestLM [14] is a LLM built by fine-tuning LLaMA-65B with a small, manually curated, and diverse instruction dataset covering topics from CFA exams to SEC filings and quantitative finance discussions. Evaluations by financial experts and zero-shot tests on financial NLP benchmarks indicate that InvestLM exhibits strong financial text understanding and provides helpful investment-related responses, comparable to top commercial models, suggesting that high-quality domain-specific LLMs can be developed with focused instruction tuning on strong foundation models. FinGPT [46] is an open-source LLM for finance, created to democratize access to financial data and the development of FinLLMs. It utilizes automatic data curation and efficient tuning to provide transparent resources for researchers and practitioners.

Romanko et al. [47] investigated the application of ChatGPT for asset allocation and portfolio optimization, concluding that while LLMs demonstrated efficiency in selecting stocks from the S&P 500 universe, they were not effective in determining optimal weight allocations. To address this limitation, the authors integrated traditional portfolio optimization methods to refine the model's output. Fieberg et al. [48] examined the use of ChatGPT-4 in the role of a financial advisor, assessing its potential to support investment decisions. Similarly, Pelster and Val [49] implemented an investment strategy guided by attractiveness ratings, using ChatGPT-4 to generate positive portfolio returns. Ko and Lee [50] evaluated ChatGPT's effectiveness in asset selection and diversification, reporting that its asset choices yielded superior diversity and performance metrics compared to randomly selected portfolios. Kim [51] proposed a quantitative investment framework leveraging ChatGPT's macroeconomic reasoning to guide asset class selection. Although ChatGPT was not explicitly trained for financial forecasting, the study showed that its recommendations contributed to improved portfolio efficiency.

Although LLMs have demonstrated strong performance in tasks such as sentiment analysis, summarization, and portfolio optimization, they are still prone to errors. A common type of error is known as "hallucination," where the model generates inaccurate, misleading, or fabricated content. Hallucinations often arise due to insufficient instruction, ambiguous prompts, or limitations in the model's training data. Prior to the integration of browser-based agents or API access, most commercial LLMs lacked real-time information and operated with a fixed knowledge cutoff date (December 2023). For instance, a query to ChatGPT-4 about the price of Apple Inc. stock on March 20th, 2025, would result in an incorrect or absent response, as such information falls outside its pretraining data. Kang and Liu [52] reviewed this matter thoroughly in their research by benchmarking LLMs for financial queries and explained how to mitigate the hallucinations.

Recent developments have mitigated this limitation. Many leading commercial LLMs, such as ChatGPT, Claude, and Gemini, now include access to the internet or external knowledge tools, enabling them to retrieve up-to-date information and generate more accurate, context-aware outputs. However, ensuring consistency, reliability, and source attribution in these retrieved responses remains an ongoing challenge.

Also, based on the recent study from Apple (Mirzazadeh et al. [53]), they are not efficient in mathematical operations; they often hallucinate over simple mathematical questions. The problems experienced in this thesis research and how these were mitigated are discussed in next Chapters thoroughly.

To overcome these limitations, researchers have explored techniques such as prompt engineering, fine-tuning, and RAG, as well as the use of LLMs within multi-agent systems. These enhancements improve LLM performance by grounding their outputs in real-time or domain-specific information, structuring complex workflows, and delegating specialized tasks across cooperating agents.

2.3.1 Transformer architecture

The contemporary LLMs arise from the transformer architecture (Figure 2.1) introduced by Vaswani et al. [2]. This groundbreaking work presented an attention mechanism that revolutionized language modeling, surpassing the performance of previous approaches.

The transformer model utilizes a self-attention mechanism. This mechanism allows the network to continually update and save the weights of important steps in a sequence, moving forward until the sequence is stored in matrix form. The core of the Transformer architecture is the attention, defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$$

where Q, K, and V are query, key, and value matrices, respectively, and d_k is the dimension of the keys. This mechanism enables Transformers to capture complex dependencies in sequences without relying on recurrence, leading to more parallelizable and effective models for various NLP tasks. Transformers have been used in financial applications for forecasting purposes in the recent past (For example, see [35]). More technical aspects of the transformers are explained in the Appendix A.5.

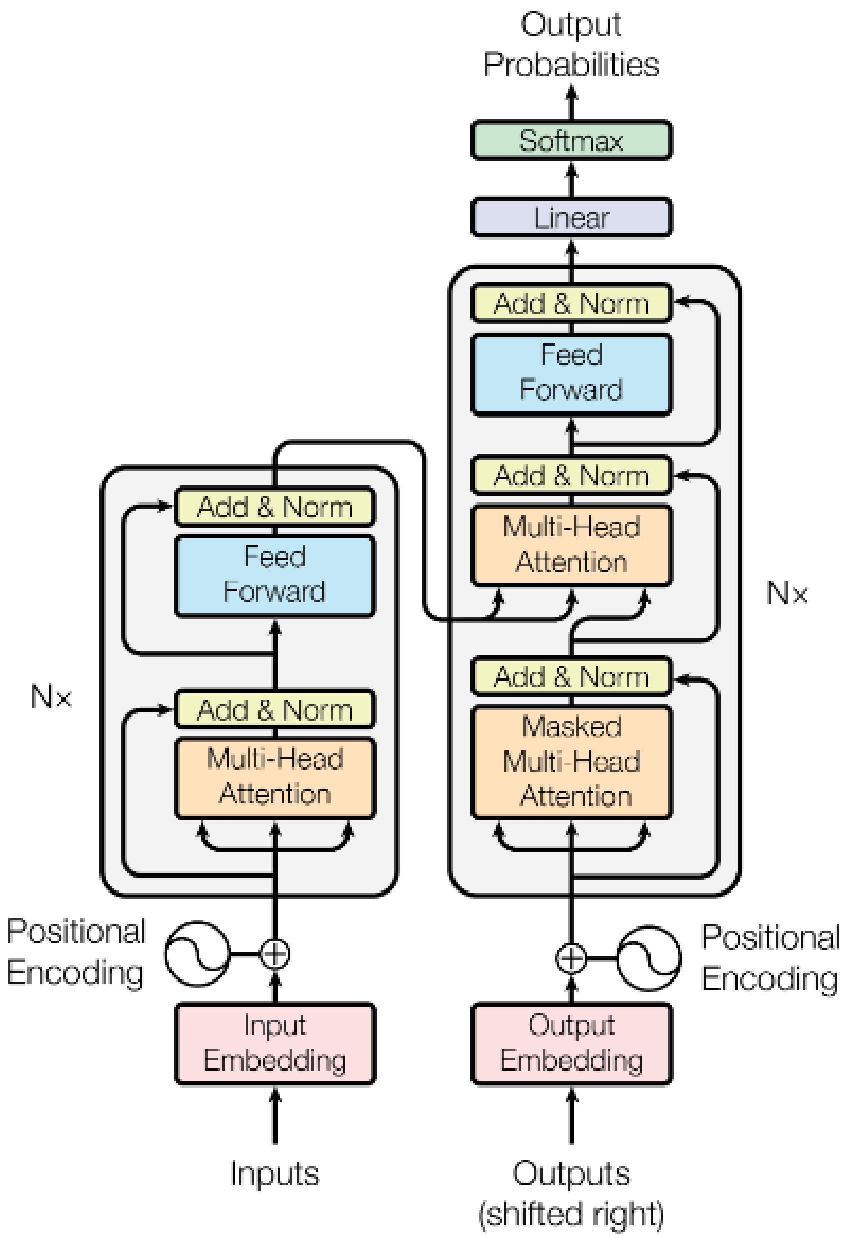


Figure 2.1: Transformer Architecture [2]

2.3.2 Notable Large Language Models (Eliza, GPT, Llama, Claude)

Since the introduction of GPT-1 [9], the field of NLP has witnessed rapid advancements, leading to the development of a wide range of powerful LLMs. These include general-purpose pre-trained models, reasoning variants, multimodal architectures capable of processing diverse data types, and domain-specific wrappers tailored to specialized tasks such as finance, law, and coding. It is worth noting that the foundations of conversational AI can be traced back to ELIZA (Weizenbaum [54]), one of the earliest NLP programs developed in the 1960s, which demonstrated the potential of human-computer interaction long before the advent of neural architectures. ELIZA, developed by Joseph Weizenbaum in 1966 [54], was an early NLP program that mimicked a psychotherapist by using pattern matching and substitution methodology, demonstrating how machines could appear to understand human language despite lacking real comprehension. In recent years, the evolution of LLMs has led to the development of highly sophisticated models capable of understanding natural language, engaging in human-like conversations, and exhibiting contextual awareness and reasoning. Among the most prominent are the ChatGPT models developed by OpenAI, including the ChatGPT-O series and the reasoning and optimized ChatGPT-4o. Anthropic's Claude models such as Claude 3.5 and 3.7 Sonnet, have similarly shown strong performance in natural language understanding. Meta's LLaMA family and Google's Gemini and Gemma models further contribute to the growing ecosystem of advanced LLMs, each offering distinct capabilities, architecture variations, and optimization goals aimed at aligning with diverse application needs, including those in the financial domain. Features of these models are captured in Table 2.1. The context window in language models refers to the maximum number of tokens (words, parts of words, or symbols) the model can "remember" and process at once in a conversation or prompt. The LLaMA 4 family introduces a refined Mixture-of-Experts (MoE) architecture (Figure 2.2), marking a significant evolution in LLM efficiency and scalability. In MoE-based models, only a subset of the model's parameters, specifically a few "expert" networks, are activated for each input token, rather than the full parameter set. This design allows the model to maintain high capacity while significantly reducing the computational cost per inference step. For example, if you ask "How to make a carrot cake?" it won't invoke "ALL" the parameters to find and process the carrot cake recipe, instead, it will only invoke the parameters related to the "cakes", "carrot", and "baking".

The LLaMA 4 MoE variant employs a sparse routing system where each token is dynamically assigned to the most relevant top-k experts based on learned scores. This selective activation improves both throughput and memory efficiency while enabling different model components to specialize in specific linguistic patterns or tasks. Consequently, MoE architectures deliver performance similar to dense models despite using significantly fewer active parameters per

Company	Model	Training Data	Type Of Model	Context Window	Release Date
OpenAI	GPT-4.1	Up to Jun 2024	Proprietary	1,000,000	14 Apr 2025
	GPT-4.1 Mini	Up to Jun 2024	Proprietary	1,000,000	14 Apr 2025
	GPT-4.1 Nano	Up to Jun 2024	Proprietary	1,000,000	14 Apr 2025
	GPT-4o	Up to Oct 2023	Proprietary	128,000	13 May 2024
	O3	Up to May 2024	Proprietary	200,000	16 Apr 2025
	O4-mini	Up to May 2024	Proprietary	200,000	16 Apr 2025
	O4-mini-high	Up to May 2024	Proprietary	200,000	16 Apr 2025
Anthropic	Claude 3.7 Sonnet	Up to Oct 2024	Proprietary	200,000	24 Feb 2025
Google	Gemini 2.5 Pro	Up to Jan 2025	Proprietary	1,048,576	28 Mar 2025
	Gemini 2.0 Flash	Up to Dec 2024	Proprietary	131,072	15 Apr 2025
Meta	LLaMA 4 Scout 17B	Up to Aug 2024	Open-Source	10,000,000	5 Apr 2025
	LLaMA 4 Maverick 400B	Up to Aug 2024	Open-Source	1,000,000	5 Apr 2025
	LLaMA 4 Behemoth 2T	Up to Aug 2024	Open-Source	TBA	TBA
	LLaMA 3.1 405B	Up to Dec 2023	Open-Source	128,000	23 Jul 2024
Mistral AI	Mistral Small 3.1	Up to Sep 2024	Open-Source	128,000	17 Mar 2025

Table 2.1: Model Features of Latest LLM model releases (LLaMA 4 Behemoth still in training as of May 2025, when this thesis was written. The GPT-4o model is constantly being updated by OpenAI)

forward pass, making them particularly valuable for production environments that prioritize inference efficiency [3]. In other words, architecture and system design in MoE play a major role in performance improvement, Ilya Sutskever (Co-founder and Chief Scientist of OpenAI) said in the 2024 NeurIPS talk, the age of pre-training and old models is done, and now we need better computing sources, better clusters, and better algorithms.

2.4 Retrieval-Augmented Generation (RAG)

2.4.1 Definition and importance

RAG is a method designed to mitigate the limitations of outdated or incomplete knowledge in pre-trained LLMs. It operates by integrating an external knowledge base that supplements

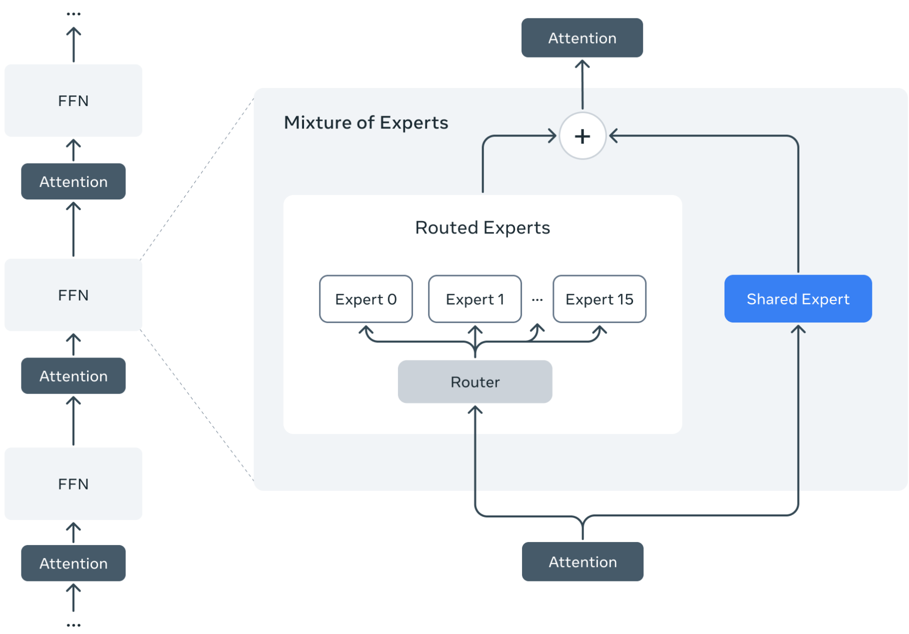


Figure 2.2: LLaMA 4 MoE Architecture [3]

the model's original training data, allowing it to retrieve and incorporate up-to-date information during inference. Importantly, the core language model remains unchanged; instead, the enhancement occurs through the retrieval mechanism, which improves response accuracy and factual grounding.

By 2024, RAG implementations were relatively simple, but the field has since evolved rapidly. Emerging variations such as GraphRAG (Han and 17 other authors [55]) and CorrectiveRAG (Yan et al. [56]) have introduced new strategies for reasoning, modularity, and iterative refinement. While a detailed discussion of these advanced architectures is beyond the scope of this thesis, we focus on the fundamental RAG design employed in this research, which is described below:

Instead of relying solely on the model's pre-trained parameters, RAG enables the system to retrieve relevant documents from an external dataset and use that information as additional context when generating outputs. This allows the model to operate with access to more current, task-specific, or domain-relevant data without requiring full retraining or fine-tuning. RAG is especially effective in dynamic fields such as finance, where real-time data is essential for informed decision-making.

The general RAG pipeline consists of several components and sequential steps (See Fig-

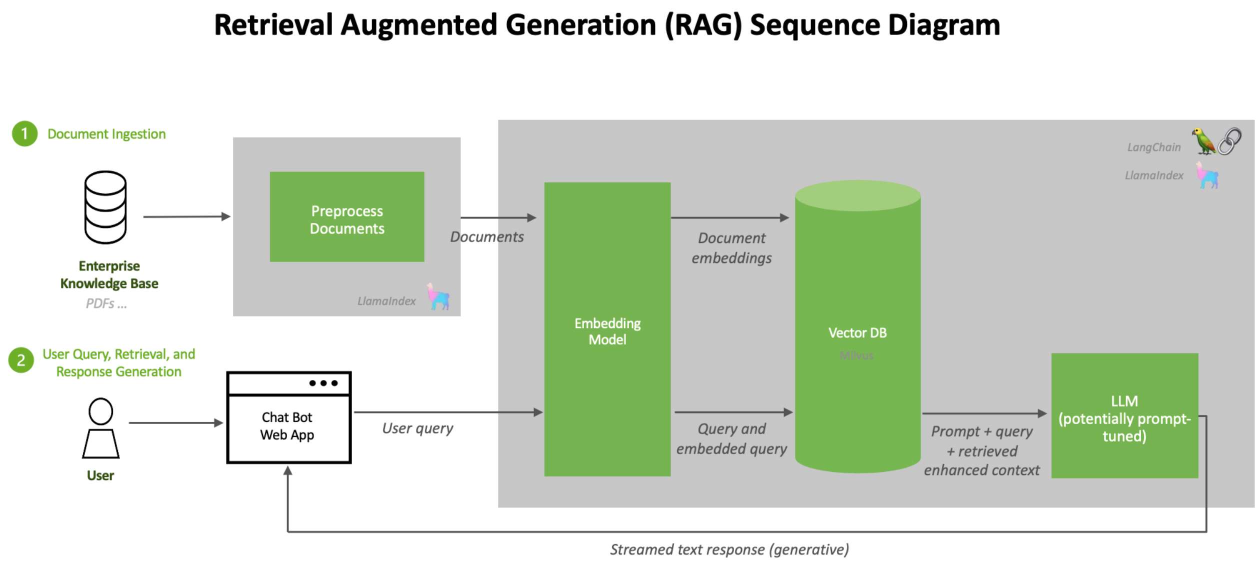


Figure 2.3: RAG Architecture [4]

ure 2.3):

1. **Document Preparation**: A domain-specific dataset is first collected and formatted, typically in JSON or text form. Each entry represents an individual unit of knowledge, such as a financial asset, report, or indicator.

2. **Chunking**: The documents are split into smaller overlapping segments or "chunks" to ensure compatibility with the input size limitations of the language model. This process preserves context across segments while enabling efficient processing.

3. **Embedding and Indexing**: Each text chunk is converted into a vector representation using a text embedding model, such as TF-IDF or a neural embedding method. These vectors are then stored in a searchable index.

4. **Retrieval**: When a user query is issued, the system encodes the query into a vector and searches the index to find the most relevant document chunks. Similarity is typically measured using cosine distance or a related metric.

5. **Context Injection**: The retrieved text segments are appended to the user's query and passed as input to the language model. This expanded context allows the model to produce more informed and grounded responses.

6. **Response Generation**: The model uses both the original query and the retrieved external information to generate a final answer. This step leverages the model's generative capabilities while grounding the output in verifiable data.

RAG provides a modular and scalable solution for improving LLM performance in scenarios where domain relevance and up-to-date information are critical. It allows for real-time adaptability without altering the internal structure of the model itself.

There are also many recent studies around utilizing this technique in finance. Zhang et al. [57] demonstrated that using RAGs can enhance sentiment analysis. Iaroshev et al [58], explore this technique for comprehensive financial statement analysis via LLMs and RAG and demonstrate how RAGs can enhance the accuracy. Results and analysis of RAG enhancements from this thesis study is reported recently in [28].

2.5 Multi-Agent Systems and Frameworks

2.5.1 Definition and key concepts of agent-based systems

Following the development of RAG architectures, another significant advancement in enhancing LLM capabilities emerged: the agentic approach, which involves the use of autonomous agents within a structured LLM workflow [59]. In this paradigm, LLMs are assigned specific roles as agents capable of independently reasoning, communicating, retrieving and analyzing data, summarizing content, and executing tasks. These agents operate within a defined framework, where they can exchange intermediate results, interact with external tools or APIs, and collaboratively work toward a final objective based on the system's architecture and user input. The agentic approach introduces modularity, adaptability, and scalability into LLM-based systems, enabling more complex and coordinated problem-solving processes [59].

In LLM-centered workflows, agents are often designed to perform specific sub-tasks, such as data retrieval, summarization, risk analysis, or optimization, and pass structured outputs to downstream agents. This design promotes modularity and interpretability while also improving the system's overall performance by dividing labor across specialized agents. By leveraging these

Figure 2.4: Some of the Major Agent Vendors

principles, agent-based systems are particularly well-suited for use in finance, where multi-step decision-making and data heterogeneity are commonplace. These frameworks are frequently integrated with orchestration tools, vector databases, and state management layers to facilitate seamless agent coordination and context preservation across the workflow.

Many organizations and tech companies are increasingly integrating agent-based architectures into their workflows to automate repetitive tasks and improve operational efficiency. A notable example is Uber's deployment of ”*QueryGPT*”, an agentic RAG-based system designed to translate user instructions into executable SQL queries [60]. This system has demonstrated significant gains in internal productivity and has been successfully adopted within Uber's workforce to streamline data querying tasks. Similarly, major AI companies such as Anthropic, OpenAI, and Google are actively developing agent-based systems capable of executing complex tasks, including deep research, real-time web browsing, and automated data extraction. These agents retrieve structured information from external sources, enabling LLMs to generate context-rich and goal-aligned responses with minimal user intervention.

Google recently introduced the ”*A2A (Agents to Agents)*” initiative to promote interoperability between autonomous agents across platforms and vendors. This framework aims to establish shared standards for agent communication, enabling cross-agent collaboration and tool usage across different ecosystems [61]. OpenAI also introduced ”*Operator*”, a platform designed to enable developers and enterprises to build, manage, and deploy LLM-powered agents that can interact with APIs, tools, and real-world systems. The service facilitates orchestration, reliability, and observability for complex agentic workflows, bringing scalable infrastructure to

AI agent deployment [20]. These and some other agent models are depicted in Figure 2.4.

2.5.2 Model Context Protocol (MCP) in Agent Interactions

MCP is a structured interface that facilitates communication between LLMs and external tools in agentic workflows introduced by Anthropic [19]. MCP provides a standardized way for exposing internal data, context, and system capabilities to AI agents, enabling them to take action, invoke tools, and perform multi-step reasoning tasks. Within LLM-centered multi-agent systems, MCP supports shared understanding by allowing agents to coordinate on task execution, pass intermediate outputs, and manage role-specific metadata.

This protocol is especially useful in complex workflows where multiple agents collaborate to accomplish a goal through actionable steps. For example, by integrating an MCP server into an agentic architecture and pairing it with an advanced LLM like Claude 3.7 Sonnet, you will have a software engineer that not only suggest edits or answer questions, but also to interact directly with an entire codebase, navigating documentation, applying code modifications, and executing predefined commands. MCP defines how agents interpret system context and take coordinated actions, thereby transforming LLMs from passive assistants into context-aware autonomous collaborators.

Chapter 3

Evaluating LLMs for Portfolio Construction

In this Chapter, we explore the capability of the LLMs for portfolio construction. We leveraged eight frontier LLMs with an advanced few-shot prompting technique to explore their capabilities for portfolio construction and compare with the S&P 500 index performance. For this phase of the work, we deployed Anthropic's Claude 3.5 Sonnet, Mistral Large 2 by Mistral AI, Meta's LLaMA 3.1-405B, Google's Gemma 2-9b, and all GPT models by OpenAI to this date (GPT3.5 Turbo-0125, GPT-4 Turbo, GPT-4o, GPT-4o-mini). We introduced a few-shot prompt across all eight models to assess their abilities for creating two (stock) portfolios: (i) high-growth portfolio, and (ii) a less risky but balanced return. After each experiment, we selected a winner portfolio based on the total number of occurrences of each ticker and compared the winner portfolios by financial metrics. We learned that LLMs are capable of creating stock portfolios and could outperform the S&P 500 index. Claude 3.5 Sonnet and LLaMA 3.1 405B beat the S&P 500 index by their portfolio selections. We explored how this cutting-edge technology can potentially lead to more robust asset allocation for adaptive portfolio construction strategies.

3.1 Objective and Approach

This Chapter explores the capabilities of LLMs in constructing financial portfolios. We deployed the few-shot learning technique to unlock the in-context learning capability of LLMs and used the latest state-of-the-art models from frontier companies in this area to evaluate their potential in portfolio construction.

Definitions:

Zero-shot prompting is using a particular model to perform a task it was not explicitly trained on, based solely on its general knowledge.

Few-shot learning is a method where we instruct a model with some examples and guide it to reach an optimal response [62].

We use few-shot learning to generate precise and well-constructed outputs. In the next sections, we thoroughly analyze a few-shot learning technique. We used few-shot learning instead of zero-shot because by zero-shot prompts, we would get simple and plain answers, and we cannot assess the model's capabilities and consistency in asset selection. This Chapter focuses on two specific goals:

1. Constructing a High-growth portfolio
2. Constructing a Low-risk portfolio

and compare their performance against the S&P 500. We deployed some of the flagship LLMs: Anthropic's Claude 3.5 Sonnet, Mistral Large 2 by Mistral AI, Meta's LLaMA 3.1-405B, Google's Gemma 2-9b, and all GPT models by OpenAI to this date (August 2024) (GPT3.5 Turbo-0125, GPT-4 Turbo, GPT-4o, GPT-4o-mini). Our key contribution through this research is to (a) demonstrate the capabilities of LLMs in portfolio management; (b) analyze their performance; (c) recommend state-of-the-art LLMs to construct portfolios in comparison S&P 500 index benchmark; and (d) bring out deficiencies of LLMs considered for the financial application being studied.

Recent studies have primarily concentrated on ChatGPT 3.5 and ChatGPT-4 [47][48][50], neglecting to investigate other state-of-the-art LLMs available in the market. Moreover, these studies have solely employed zero-shot learning to highlight the potential of LLMs in financial applications. These recent studies and their limitations motivated us to this research. This thesis distinguishes itself by leveraging few-shot learning and evaluating a comprehensive range of frontier models currently accessible, rather than exclusively depending on models developed by OpenAI, such as the ChatGPTs.

3.1.1 Overview of Few-shot Prompt Engineering and Chain-of-Thoughts

To obtain the optimal solution from frontier models, we had to leverage the few-shot prompting technique, which is an approach that is effectively better than zero-shot learning. In zero-shot prompting, we ask the model for a certain task in plain and simple language without any instruction, guidance, or examples. However, in few-shot prompting, we provide the model with some examples and level-by-level instructions to steer the model to reach our optimal solution. While many models perform well with zero-shot prompts, to enable in-context learning efficiently and obtain better performance, we guided the models for what portfolios should be and instructed them on how to create two diverse portfolios using few-shot learning, and studied the performance in comparison to the S&P500 index with given constraints.

We bounded the models by the following prompt to select only "20" stocks (We selected 20 stocks to force the models to create a diverse portfolio and use its all capacity and creativity to select assets for portfolios, and avoid over-concentration) from all stock tickers listed in the S&P500 index and put a constraint that each assigned weight should reach 1.0 (100%) (see Fig.3.1, for the full prompt, also see Appendix A for further details on prompt engineering). Most of the models failed to limit the assigned weights to 1.0 and by this result, we only selected 20 stocks from each portfolio with the most occurrences. We iterated the prompt in Fig.3.1 100 times and obtained 800 responses with 200 portfolios for each of the eight models, which adds up to 1600 portfolios and 32,000 tickers. However, since the models could not be fully consistent with their outputs due to the nature of API calls and prompting, we selected 50 clean responses from each model's responses that totals of 800 portfolios and 16000 tickers.

```
prompt = """
Instruction: Create two hypothetical funds using S&P 500 tickers and explain
    your reasoning.

Step 1: Select 20 tickers from the S&P 500 index for each portfolio.
Step 2: Allocate weights to each stock aiming for higher returns than the S&P
    500.
Step 3: Ensure the sum of total weights is 1 (100%), with different weights for
    each stock.
Step 4: Write your tickers in dictionaries (curly braces) with their respective
    weights. For example, {"Ticker": Weight}.

Output:
Tickers: {"Ticker 1": Weight 1,"Ticker 2": Weight 2, ...}
Rationale: I selected these companies because they are leaders in high-growth
    sectors like technology and healthcare. The weights are based on their
    potential to outperform the market...

Task: Create two hypothetical funds using S&P 500 tickers.
"""
```

Figure 3.1: Prompt for portfolio construction, we used and iterated this prompt to create financial portfolios from LLMs

A clean response is a response with two portfolios: one high-growth portfolio and one low-risk portfolio, each containing 20 tickers. The tickers and their corresponding weights are presented in a dictionary format. Additionally, a brief explanation is provided for each portfolio to offer insights into the investment strategy and potential risks associated with the selected stocks. We also put a constraint on every output to be in a dictionary and output tickers should be in double quotes with a comma separating each ticker and weight pair. Moreover, we asked the models to briefly explain the reasoning behind their choices as shown in Fig 3.5. Interestingly, the rationale provided by the models were quite similar across all eight models we tested.

In the final phase of our experiments, following the identification of the most effective models for portfolio construction, we selected two out of the eight models for weight re-optimization. This step was necessary as the initial optimization failed to produce portfolios with weights summing to 1.00 (100%). Claude 3.5 Sonnet and LLaMA 3.1 405B emerged as the superior models. We subsequently adjusted their weights from an equal allocation of 0.05 (5%) each to their optimal weights, determined through the re-optimization prompt (See Fig. 3.2; for the full prompt also see Appendix A for further details)

3.1.2 Eight frontier LLMs tested

In this section, we evaluate our methodology for this Chapter as it pertains to addressing the objectives stated above. We used the following LLM models from frontier companies in the generative AI field, such as OpenAI, Anthropic, Google, Mistral AI, and Meta. Table 3.1 shows the models with their properties. These models are the latest technology in this field, and they are evolving rapidly. The latest models (as of Aug.28, 2024) are LLaMA 3.1 405B by Meta, Mistral 2 Large by Mistral AI, and OpenAI's GPT-4o-Mini.

Company	Model	Training Data	Type Of Model	Context Window	Max Output Tokens	Release Date
OpenAI	GPT-3.5-Turbo-0125	Up To Sep 2021	Proprietary	16,385 Tokens	4,096 Tokens	Jan 25, 2024
	GPT-4-Turbo	Up To Dec 2023	Proprietary	128,000 Tokens	4,096 Tokens	Nov 6, 2023
	GPT-4o-Mini	Up To Oct 2023	Proprietary	128,000 Tokens	16,384 Tokens	July 18, 2024
	GPT-4o	Up To Oct 2023	Proprietary	128,000 Tokens	4,096 Tokens	May 13, 2024
Mistral AI	Mistral Large 2	Up To Aug 2023	Open-Source	128,000 Tokens	8,192 Tokens	July 24, 2024
Anthropic	Claude 3.5 Sonnet	Up To Apr 2024	Proprietary	200,000 Tokens	4,096 Tokens	June 20, 2024
Meta	LLaMA 3.1 405B	Up To Dec 2023	Open-Source	128,000 Tokens	2,048 Tokens	July 23, 2024
Google	Gemma 2 – 9b	-	Open-Source	8,000 Tokens	4,096 Tokens	June 27, 2024

Table 3.1: Large Language Models used in this phase

We utilized the most recent LLM models available at the time of experimentation to en-

```
prompt: """
Instruction: Assign weights to two given hypothetical funds of 20 stocks from
    S&P 500 index and explain your reasoning.
Role: You are an expert in financial analysis and portfolio management.
Task:
1. Your task is to distribute the weights of two given portfolios to maximize
    returns and outperform the S&P 500 index.
2. Mention the weight of each stock in the funds and the total weight should be
    1 (100%)
3. Ensure the total of all weights equals exactly 1.00 (100%)
4. Aim for higher returns than S&P 500 Index
Output:
Optimized Portfolio 1: {"Ticker 1": Weight 1,"Ticker 2": Weight 2,"Ticker 3":
    Weight 3,"Ticker 4": Weight 4, ...}
Optimized Portfolio 2: {"Ticker 1": Weight 1,"Ticker 2": Weight 2,"Ticker 3":
    Weight 3,"Ticker 4": Weight 4, ...}
"""
```

Figure 3.2: Prompt for optimizing the portfolios. We used this prompt to determine the best model after optimization and adjust the ticker weights.

sure that our results are robust and reflective of the latest capabilities of LLMs in financial applications. Using these powerful models, we captured the potential of LLMs in financial decision-making and portfolio construction.

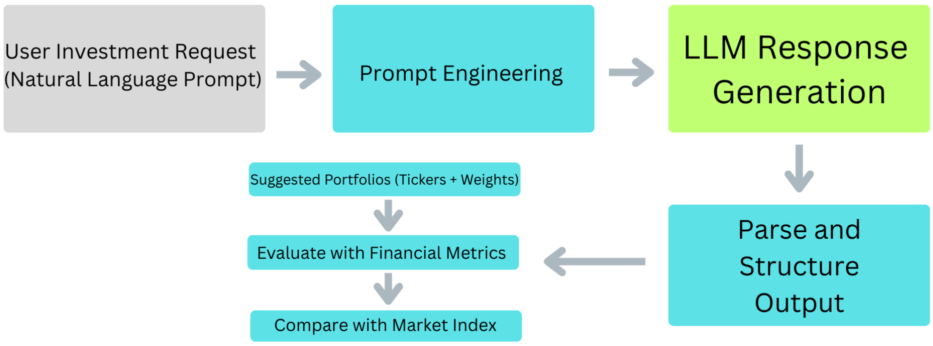


Figure 3.3: The Workflow of Our Methodology

3.2 Experimentation and Evaluation

3.2.1 Portfolio Construction

Given a universe of S&P 500 stocks $U = \{s_i : s_i \in \text{S\&P500}\}$, we construct two portfolios by selecting subsets $P_1, P_2 \subset U$ where:

$$|P_1| = |P_2| = 20 \quad \text{(20 stocks per portfolio)} \tag{3.1}$$

For each portfolio $k \in \{1, 2\}$, define the weight vector $\mathbf{w}_k = (w_{k1}, w_{k2}, ..., w_{k20})$ subject to:

$$\sum_{i=1}^{20} w_{ki} = 1 \quad \text{(Sum of weights = 100\%)} \tag{3.2}$$

$$0 < w_{ki} \leq 1 \quad \text{(Positive weights)} \tag{3.3}$$

Portfolio 1 (High Growth):

$$P_1 = \{(s_i, w_{1i}) : s_i \in \text{selected S\&P500 stocks}, i = 1, ..., 20\} \tag{3.4}$$

Portfolio 2 (Low Risk):

$$P_2 = \{(s_i, w_{2i}) : s_i \in \text{selected S\&P500 stocks}, i = 1, ..., 20\} \tag{3.5}$$

For N iterations of LLM-generated portfolios, let $f_k(s_i)$ represent the frequency of ticker s_i appearing in portfolio k. The total return for each portfolio over time period $[t_0, T]$ is:

Portfolio 1 (High-Growth) total return:

$$R_{P1} = \sum_{i=1}^{20} w_{1i} \cdot \frac{p_i(t) - p_i(t_0)}{p_i(t_0)} \quad \text{where } s_i \text{ selected based on } \max_{s_i \in U} f_1(s_i) \tag{3.6}$$

Portfolio 2 (Low-Risk) total return:

$$R_{P2} = \sum_{i=1}^{20} w_{2i} \cdot \frac{p_i(t) - p_i(t_0)}{p_i(t_0)} \quad \text{where } s_i \text{ selected based on } \max_{s_i \in U} f_2(s_i) \tag{3.7}$$

where $p_i(t)$ represents the price of stock i at time t, and the selection of stocks is based on their frequency of appearance in the N LLM-generated portfolios.

3.2.2 Experiments

For the experimental setup, we utilized the models through their official APIs except LLaMA 3.1 405B and Gemma 2 – 9B, for which we used the NVIDIA NIM platform's API. We iterated the prompt 100 times for each model to ensure that models were consistent with their responses, which was not very successful, and some of the responses were not what we expected. Analysis of the model's responses revealed significant inconsistencies in asset allocation recommendations. For example, while we expected comprehensive outputs with valid ticker symbols and appropriate weight distributions, many responses contained incomplete or non-existent tickers, along with mathematically incorrect weight assignments. Hence, we cleaned and selected only 50 responses from each API call to cover all models consistently. We gathered 100 portfolios (2 portfolios for each response) from each model. We asked the models for two different portfolios with one shared goal, which is the baseline for our evaluation. Portfolio 1 consists of more rewarding stocks and higher growth potential, and Portfolio 2 was constructed with less risk tolerance and balanced return.

To facilitate a fair experiment, we consider the parameters for creativity (temperature), diversity (top_p), and text generation length (max_tokens) to "temperature" = 0.7, "top_p" = 1, and "max_tokens" = 4096. We chose these parameters because we want the models to be consistent with their responses. If we increase the temperature, the model will be more creative, and on the other hand, if we decrease it, it will be more consistent with its response. "Top_p" controls diversity by selecting from the smallest set of tokens whose cumulative probability exceeds the specified value. The value of "top_p" is default to 1 so we left it unchanged for the highest diversity in asset selection. "Max_tokens" limits the length of the generated text by specifying the maximum number of tokens the model can produce in a response. Fig. 3.4 is a sample response from Claude 3.5 Sonnet.

As shown in Fig. 3.4, the first portfolio mostly consists of stocks from technology and communication services. After obtaining the results, through a preliminary analysis, we reached an initial conclusion that models cannot be bounded to the constraint of adding up weights to 1.0, and responses were not consistent. We analyzed stock selection frequency and portfolio diversification across all models to evaluate model performance. This approach enables us to identify the stocks deemed most significant by the models, providing insight into their decision-making processes and preferences in portfolio construction. For a fair and robust comparison we selected 20 of the most occurring stocks in both portfolios as an optimal portfolio out of 50 responses (100 portfolios) and assigned a 5% weight to each ticker, then by "yfinance" (Yahoo! Finance) python package we computed and compared results from the models by financial portfolio metrics such as Average Annual Return, Annualized Volatility, Sharpe Ratio, Maximum Drawdown, Alpha, Beta, value-at-risk (VaR) 95%, and VaR 99% (See Appendix A.1 for detailed explanation for metrics). The training data cutoff dates varied across models, necessitating both in-sample and out-of-sample testing for two distinct periods:

1. 2018-01-01 to 2021-12-31
2. 2022-12-31 to 2024-08-09

This approach was adopted to assess the profitability of model-generated portfolios using out-of-sample data. By evaluating performance beyond their training cutoff dates, we aimed to determine whether the models' stock selections, based on their in-sample knowledge, remained effective in future market conditions. For example, we want to see the stock selections by ChatGPT 3.5 if it is still profitable in 2024. The concluding stage of our analysis involved refining the portfolio construction process. Having identified the most promising models, we narrowed our focus to two out of the original eight for weight re-optimization. This step was

Portfolio 1: High Growth S&P 500 Outperformer

{"AAPL": 0.08, "MSFT": 0.07, "AMZN": 0.06, "TSLA": 0.05, "NVDA": 0.05, "META": 0.04, "GOOGL": 0.04, "AVGO": 0.04, "ADBE": 0.04, "CRM": 0.04, "PYPL": 0.04, "NFLX": 0.04, "ASML": 0.04, "COST": 0.04, "MRNA": 0.04, "REGN": 0.04, "ISRG": 0.04, "LRCX": 0.04, "KLAC": 0.04, "AMAT": 0.03}

Rationale: This portfolio is designed to capture the growth potential of leading technology, consumer discretionary, and healthcare companies. It includes dominant players in sectors such as software, e-commerce, electric vehicles, semiconductors, biotechnology, and digital entertainment. The weights are skewed towards the larger, more established companies while still allocating a significant portion to promising high-growth names. The goal is to outperform the S&P 500 by capitalizing on innovative companies with strong competitive advantages and long-term growth prospects.

Portfolio 2: Low-Risk S&P 500 Outperformer {"AAPL": 0.07, "MSFT": 0.07, "JNJ": 0.06, "JPM": 0.05, "XOM": 0.05, "UNH": 0.05, "PG": 0.05, "VZ": 0.05, "PFE": 0.04, "WMT": 0.04, "CVX": 0.04, "KO": 0.04, "MRK": 0.04, "PEP": 0.04, "CSCO": 0.04, "COST": 0.04, "DIS": 0.04, "HD": 0.04, "ABBV": 0.04, "INTC": 0.04}

Rationale: This portfolio focuses on large, established companies with stable earnings, strong balance sheets, and a track record of consistent dividend payments. It includes defensive sectors such as healthcare, consumer staples, utilities, and financials. The weights are distributed among industry leaders with durable competitive advantages and relatively low volatility. The goal is to provide steady returns with lower risk than the broader market, while still aiming to outperform the S&P 500 through a combination of dividend income and moderate capital appreciation.

Figure 3.4: Sample Response from Claude 3.5 Sonnet. It contains two portfolios; one high-growth portfolio and one low-risk portfolio

necessitated by the initial models' inability to generate portfolios with weights totaling precisely 1.00 (100%). Claude 3.5 Sonnet and LLaMA 3.1 405B demonstrated superior performance and were thus selected for further optimization. We initialized these models with equal weights of 0.05 (5%) and subsequently optimized their allocations by a single iteration to determine their ideal weights within the portfolio structure with prompt 2 (see Fig. 3.2).

3.3 Results and Analysis

Through extensive experiments, we come to an understanding that LLM models can process financial terms and can be helpful in financial decision-making. They are trained on vast amounts of data, which includes financial terms and expressions, and they can comprehend financial jargon. By this initial understanding, we illustrated that LLMs can choose various tickers from all tickers listed in the S&P500 index and state a rationale for their behavior. The logic behind their rationale was somehow similar with minor differences, and most of them mentioned a rationale like below in Fig. 3.5:

> "**For portfolio 1**, I selected high-growth tech companies and innovative firms in sectors like healthcare and fintech. These stocks have shown strong momentum and have the potential for significant future growth. The weights are allocated based on their market dominance and growth prospects.
> **For Portfolio 2**, I focused on established, blue-chip companies with strong balance sheets, consistent dividends, and defensive characteristics. These companies tend to be more stable during market downturns and provide steady returns. The weights are distributed to balance stability with moderate growth potential."

Figure 3.5: Sample Rationale from Claude 3.5 Sonnet. By this rationale, you can determine the mindmap of the LLM for choosing tickers

The number of occurrences for every stock in portfolios was similar to each other and most of them mainly selected NVIDIA, Amazon, Apple, Johnson & Johnson, Walmart, Meta, Procter and Gamble, Coca-Cola, and many more but the similarities were very high (See Fig.3.6 and Fig.3.7), also see Appendix A for all stock occurrences from different models.

GPT-4o stock occurrences for portfolio 1 and 2

Portfolio 1	AAPL	ZM	GOOGL	AMZN	NVDA	TSLA	META	ADBE	NFLX	CRM	MSFT	SQ	DOCU	SHOP	PYPL	ROKU	TWLO	AMD	INTU	SNOW
Occurance	50	50	50	50	50	50	50	50	50	50	50	50	49	48	47	44	42	41	20	20
Portfolio 2	JNJ	KO	PEP	VZ	MRK	PG	WMT	T	PFE	MCD	CL	KMB	GIS	HD	COST	MDT	LLY	XOM	ABBV	CVX
Occurance	50	50	50	50	50	50	49	49	49	41	39	37	37	34	34	31	27	22	21	19

Table 3.2: GPT-4o stock occurrences for portfolio 1 and 2

Furthermore, we selected a winner portfolio from each response by the number of occurrences and did a comprehensive in-date and out-date testing to demonstrate the results.

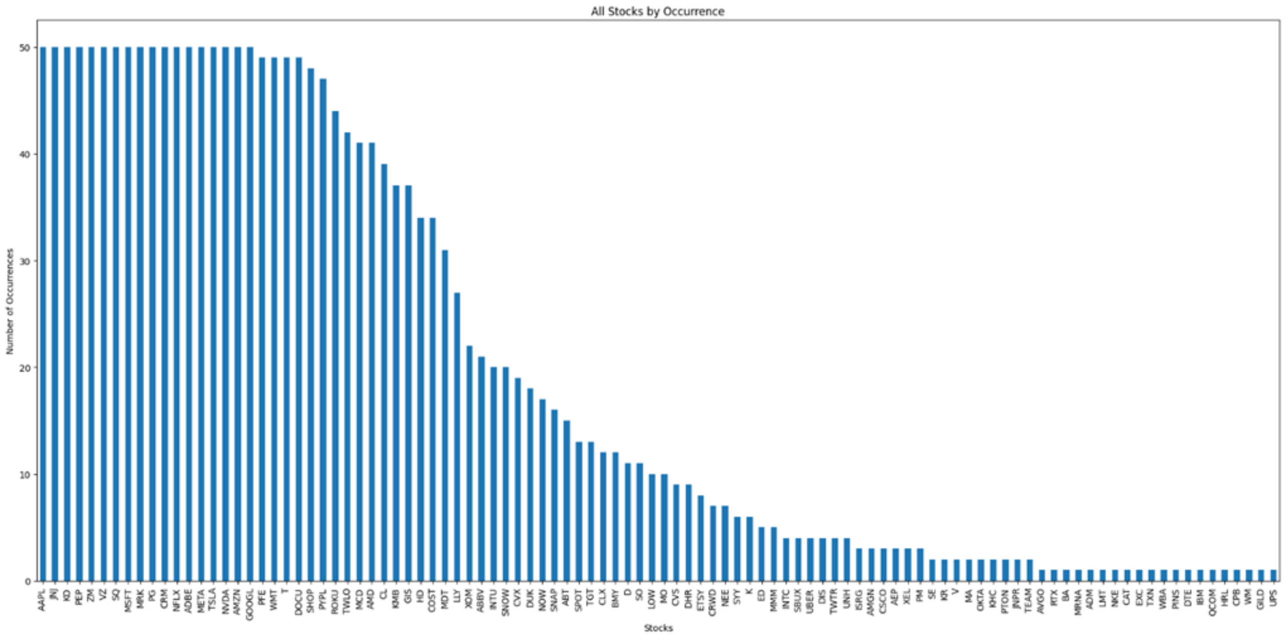
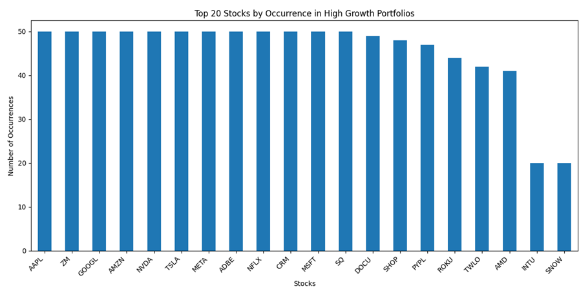
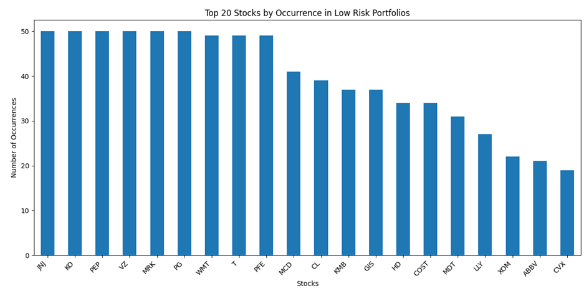


Figure 3.6: Total occurrence of stocks in GPT-4o



(a) Top 20 stocks in high-growth portfolios for GPT-4o



(b) Top 20 stocks in low-risk portfolios for GPT-4o

Figure 3.7: Stock occurrences for GPT-4o

In Table 3.2 we can see that the ChatGPT-4o can create diversified financial portfolios and outperform the S&P500 in some scenarios with its asset selection. Based on the results demonstrated in Tables 3.3 - 3.6, they did not perform well in creating low-risk portfolios, but they achieved promising results in curating high-growth portfolios. While LLMs and GPTs fundamentally rely on transformer architectures, their precise decision-making processes remain vague, and their architectural details often differ substantially. Many of these models, particularly those developed by private companies, have limited public documentation about their internal mechanisms and training methodologies. Knowing that LLMs are black boxes, we cannot firmly determine the reason why they could not achieve better performance over the

S&P 500 index for low-risk portfolios, however a few potential reasons could be market conditions, the risk-return trade-off, or the model's bias toward growth stocks. The stocks these models chose for the low-risk portfolio are prominent blue-chip stocks with high dividends and great statements, but the models seem to somehow compromise the return to ensure that the response is "low" in risk and obeying this one particular constraint. LLM models' success in high-growth portfolios can be attributed to their focus on sectors with strong innovation and growth potential, such as technology, communication services, consumer cyclical, and financial services. These selections indicate strong profitability and investor confidence in future growth, while often paying lower dividends. Despite the inherent volatility in these sectors, the LLMs chosen portfolios demonstrated higher return-to-risk ratios, suggesting the models' proficiency in identifying companies with robust growth prospects relative to their risk profiles (See Table 3.3). This pattern reveals the LLM's bias towards innovative, momentum-driven companies and their ability to interpret complex market data, though it also highlights the need for careful consideration of overall portfolio balance and risk management. Table 3.3 through Table 3.6 demonstrate the analysis of the results for all eight LLM models, presenting key financial metrics.

Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
LLaMA 3.1 405B	57%	30.79%	1.8235	54.16%	-2.98%	-4.52%	0.9854	0.3239
GPT-4o mini	55.93%	31.32%	1.7541	52.44%	-3.16%	-4.75%	1.0674	0.3082
Gemma 2 - 9b	54.62%	28.06%	1.91112	47.94%	-2.76%	-4.60%	1.0748	0.3258
GPT-4 Turbo	51.88%	30.49%	1.6691	49.99%	-3.00%	-4.99%	1.1343	0.2458
Mistral 2 Large	45.89%	29.22%	1.5365	42.20%	-2.94%	-4.70%	1.0931	0.2019
GPT-3.5 Turbo	36.31%	26.58%	1.3284	59.32%	-2.46%	-4.68%	1.1627	0.1854
GPT-4o	33.28%	28.35%	1.1386	27.24%	-3.09%	-4.82%	1.2865	-0.3832
Claude 3.5 Sonnet	30.59%	26.49%	1.1172	21.73%	-3.16%	-4.38%	1.4335	-0.727
S&P 500 Index	15.40%	21.25%	0.68	33.92%	-1.96%	-3.74%	1	-

Table 3.3: Portfolio 1 (High-Growth) - In-date testing (2018-2021)

It is obvious that LLMs were mostly successful in creating portfolios for high-growth selection and often failed to curate an effective low-risk portfolio. Fig.3.8 plots the results for both portfolios in out-date and in-date testing.

For in-date testing, which encompasses the period within most models' data cutoffs, LLaMA 3.1 405B and Claude 3.5 Sonnet demonstrated superior performance compared to other models.

Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
Claude 3.5 Sonnet	52.75%	26.15%	1.8257	40.27%	-2.78%	-3.95%	1.6984	0.171
GPT-4 Turbo	52.59%	22.81%	2.0867	36.00%	-2.29%	-3.24%	1.5605	0.1943
Mistral 2 Large	47.03%	21.63%	1.943	34.21%	-2.11%	-3.18%	1.4913	0.1511
GPT-4o	44.97%	25.47%	1.5692	31.38%	-2.70%	-3.65%	1.6631	0.0996
GPT-4o mini	43.39%	23.84%	1.61	31.42%	-2.53%	-3.62%	1.6071	0.0939
LLaMA 3.1 405B	41.45%	22.86%	1.5941	29.40%	-2.31%	-3.47%	1.5281	0.0887
GPT-3.5 Turbo	41.45%	17.99%	2.0256	21.95%	-1.81%	-2.53%	1.2818	0.1331
Gemma 2 - 9b	40.56%	20.41%	1.7418	26.27%	-1.99%	-3.13%	1.4346	0.0967
S&P 500 Index	22.79%	12.91%	1.38	10.28%	-1.37%	-1.84%	1	-

Table 3.4: Portfolio 1 (High-Growth) - Out-date testing (2022-2024)

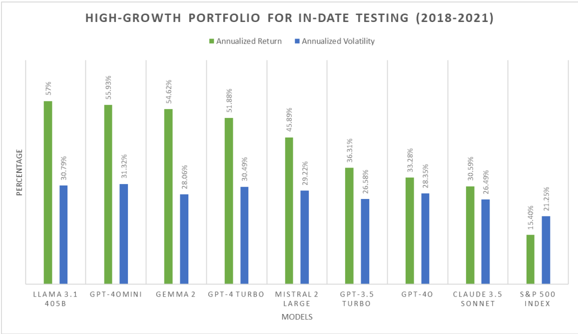
Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
Claude 3.5 Sonnet	18.60%	18.70%	0.9448	42.55%	-1.62%	-3.47%	0.8133	0.0512
S&P 500 Index	15.40%	21.25%	0.68	33.92%	-1.96%	-3.74%	1	-
GPT-3.5 Turbo	14.73%	18.15%	0.7565	38.68%	-1.56%	-3.42%	0.7642	0.027
GPT-4o	13.98%	17.39%	0.7467	34.11%	-1.57%	-3.09%	0.7194	0.0261
LLaMA 3.1 405B	13.91%	17.97%	0.7186	33.09%	-1.59%	-2.89%	0.7574	0.0199
GPT-4o mini	13.47%	18.16%	0.6865	37.01%	-1.60%	-3.13%	0.7784	0.0124
GPT-4 Turbo	13.39%	17.57%	0.7052	35.01%	-1.42%	-3.44%	0.6522	0.0299
Gemma 2 - 9b	11.99%	20.47%	0.5366	39.65%	-1.84%	-3.20%	0.8903	-0.0185
Mistral 2 Large	10.23%	19.31%	0.478	39.37%	-1.66%	-3.17%	0.8241	-0.0266

Table 3.5: Portfolio 2 (Low-Risk) - In-date testing (2018-2021)

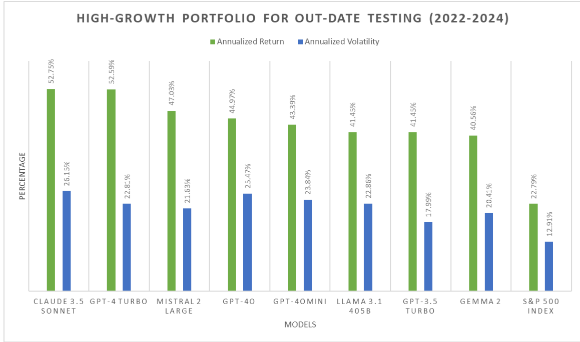
Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
S&P 500 Index	22.79%	12.91%	1.38	10.28%	-1.37%	-1.84%	1	-
GPT-3.5 Turbo	13.33%	10.68%	0.78	9.03%	-1.03%	-1.59%	0.4302	0.0057
Claude 3.5 Sonnet	13.02%	9.89%	0.8111	8.47%	-0.97%	-1.41%	0.4647	-0.037
Gemma 2 - 9b	11.75%	11%	0.6138	9.05%	-1.06%	-1.63%	0.5626	-0.034
GPT-4o	10.76%	9.80%	0.5874	8.54%	-0.95%	-1.53%	0.3665	-0.0086
Mistral 2 Large	8.75%	10.33%	0.3632	10.19%	-0.97%	-1.56%	0.4293	-0.00399
LLaMA 3.1 405B	8.71%	10.59%	0.3502	8.60%	-1.03%	-1.54%	0.4705	-0.0478
GPT-4o mini	5.39%	10.47%	0.0372	11.49%	-1.05%	-1.56%	0.4377	-0.0751
GPT-4 Turbo	2.71%	11.62%	-0.1968	16.37%	-1.18%	-1.66%	0.3108	-0.0789

Table 3.6: Portfolio 2 (Low-Risk) - Out-date testing (2022-2024)

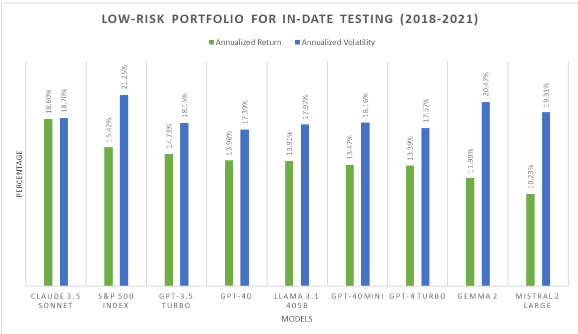
For out-date testing, which covers outside the training data cutoff, only Claude 3.5 Sonnet could beat the S&P500 index in high growth portfolio and other models failed to beat the S&P 500 Index on low-risk portfolio construction. As shown in Table 3.3 - 3.6, the beta is a valid metric for our analysis and demonstrates that most of the portfolios for high-growth selection got a beta greater than 1 and on the other hand for low-risk selection, the beta is less and near 1. Beta measures a stock or portfolio's volatility or systematic risk relative to the overall market. The Sharpe ratio is also higher for high-growth portfolios and lower for low-risk portfolios, which is evident due to the fundamental composition of the Sharpe ratio. This metric inherently favors



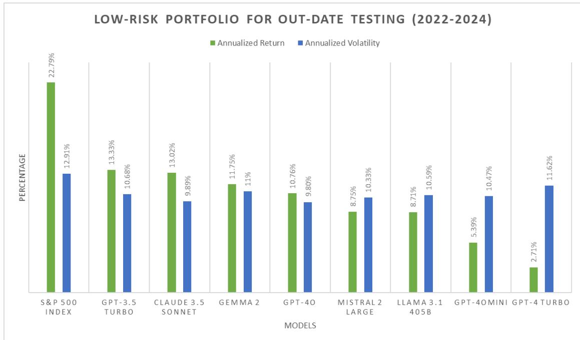
(a) High-Growth Portfolio for in-date testing (2018-2021)



(b) High-Growth Portfolio for out-date testing (2022-2024)



(c) Low-Risk Portfolio for In-date testing (2018-2021)



(d) Low-Risk Portfolio for out-date testing (2022-2024)

Figure 3.8: Comparison of High-Growth and Low-Risk Portfolios for different periods

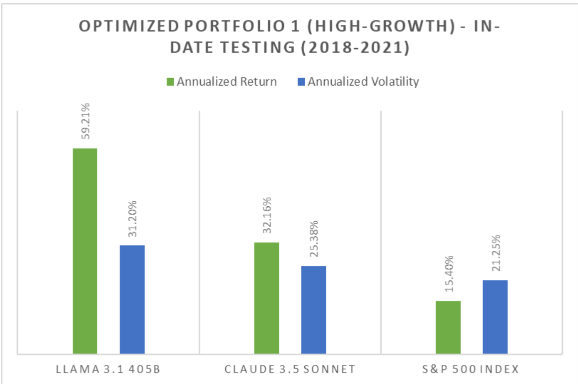
higher returns relative to risk, aligning with the characteristics of high-growth portfolios. This relationship underscores the trade-off between risk and return, highlighting that while high-growth portfolios may offer potentially higher returns, they also come with increased volatility, whereas low-risk portfolios prioritize stability at the expense of potentially lower returns. After determining the best models for portfolio construction, we used them to re-optimize the weights. In our analysis, it is important to mention that Claude 3.5 Sonnet successfully re-optimized its winning portfolio, achieving an exact 1.00 (100%) weight distribution. In contrast, LLaMA 3.1 405B encountered persistent challenges, getting substandard results in the re-optimization process. Specifically, LLaMA 3.1 405B produced weight distributions of 103% for Portfolio 1 (High-growth portfolio) and 98% for Portfolio 2 (Low-risk portfolio), deviating from the desired 100% allocation. Table 3.7 to Table 3.10 illustrates the results for optimized portfolios.

Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
LLaMA 3.1 405B	59.21%	31.20%	1.866	4.70%	-2.95%	-4.65%	1.0405	0.3313
Claude 3.5 Sonnet	32.16%	25.38%	1.2278	21.36%	-2.90%	-4.21%	1.372	-0.6673
S&P 500 Index	15.40%	21.25%	0.68	33.92%	-1.96%	-3.74%	1	-

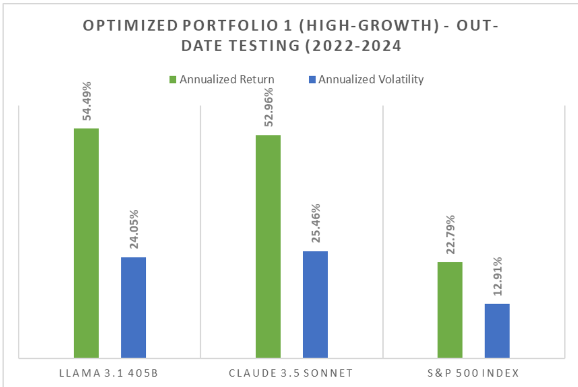
Table 3.7: Optimized Portfolio 1 (High-Growth) - In-date testing (2018-2021)

Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
Claude 3.5 Sonnet	20.66%	18.95%	1.0376	44.66%	-1.59%	-3.61%	0.8279	0.0772
S&P 500 Index	15.40%	21.25%	0.68	33.92%	-1.96%	-3.74%	1	-
LLaMA 3.1 405B	12.97%	18.06%	0.6628	3.08%	-1.61%	-3.02%	0.7261	0.015

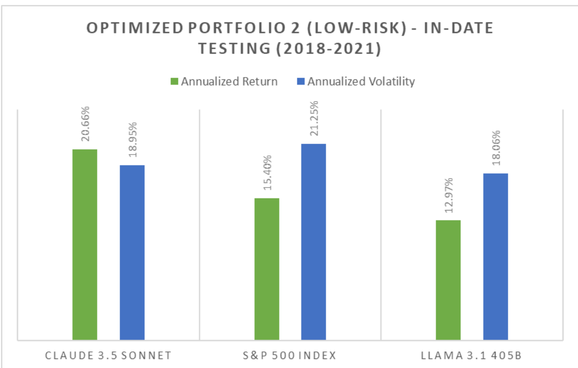
Table 3.8: Optimized Portfolio 2 (Low-Risk) - In-date testing (2018-2021)



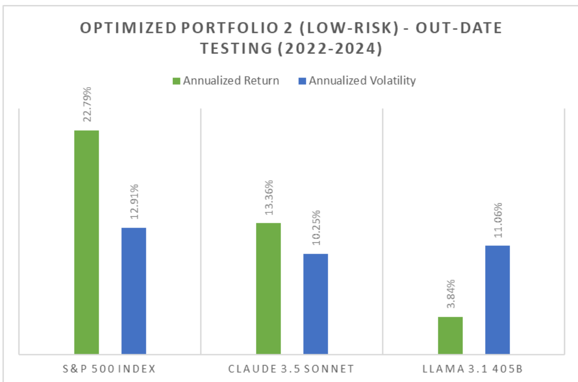
(a) Optimized Portfolio 1 (High-Growth) - In-date testing (2018-2021)



(b) Optimized Portfolio 1 (High-Growth) - Out-date testing (2022-2024)



(c) Optimized Portfolio 2 (Low-Risk) - In-date testing (2018-2021)



(d) Optimized Portfolio 2 (Low-Risk) - Out-date testing (2022-2024)

Figure 3.9: Comparison of Optimized Portfolios for different periods

After normalizing the weights for LLaMA 3.1 405B, our analysis revealed that LLaMA 3.1 405B outperformed Claude 3.5 Sonnet. Despite its initial deviation from the target 100% allocation, LLaMA 3.1 405B demonstrated superior optimization capabilities, yielding a more effective

Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
LLaMA 3.1 405B	54.49%	24.05%	2.0578	45.23%	-2.40%	-3.56%	1.5798	0.2098
Claude 3.5 Sonnet	52.96%	25.46%	1.8841	41.11%	-2.70%	-3.61%	1.6556	0.1808
S&P 500 Index	22.79%	12.91%	1.38	10.28%	-1.37%	-1.84%	1	-

Table 3.9: Optimized Portfolio 1 (High-Growth) - Out-date testing (2022-2024)

Models	Annualized Return	Annualized Volatility	Sharpe ratio	Maximum drawdown	Value at Risk (95%)	Value at Risk (99%)	Beta	Alpha
S&P 500 Index	22.79%	12.91%	1.38	10.28%	-1.37%	-1.84%	1	-
Claude 3.5 Sonnet	13.36%	10.25%	0.8161	1.03%	-1.01%	-1.38%	0.5097	-0.0084
LLaMA 3.1 405B	3.84%	11.06%	-0.1049	10.88%	-1.04%	-1.74%	0.3721	-0.0787

Table 3.10: Optimized Portfolio 2 (Low-Risk) - Out-date testing (2022-2024)

weight distribution upon normalization. Fig.3.9 plots the results from the re-optimization process showing LLaMA 3.1 405B has superior performance to Claude 3.5 Sonnet in High-growth portfolios, and Claude 3.5 Sonnet is still performing better than the S&P500 index in low-risk portfolio with slightly better performance after optimization.

3.4 Limitations

3.4.1 Weight optimization challenges

The majority of the eight models failed to achieve optimal weight distribution constraints. In numerous instances, we were forced to normalize the weight distributions to facilitate our analysis, as the models' outputs frequently deviated from the desired allocations. Since most of the LLMs are acting like a black box and we do not know the true rationale behind their statements, we can only improve their efficiency by utilizing augmentation techniques and external tools. We can connect the LLM models to a RAG with up-to-date statements and historical data of stocks to enhance the portfolio construction. In the next Chapters, we mitigate all the mentioned risks and limitations first with RAGs and then with a multi-agent framework.

Chapter 4

Enhancing Portfolio Construction with RAG

This Chapter assesses the LLMs in creating investment portfolios. We implement a few-shot learning technique, followed by retrieval augmented generation (RAG) enhanced with comprehensive up-to-date financial data, using Meta's latest LLM, Llama 3.1-8b. At first, we assess the models' efficacy using key financial indicators, including total returns, annualized volatility, risk-adjusted performance (Sharpe ratio), potential loss estimates (value-at-risk), and their pre-training knowledge with the S&P 500 Index performance baseline. Moreover, we enhance the LLM's knowledge base by RAG and the latest historical and statistical metrics (such as earnings per share (EPS), dividends per share (DPS), profit margin, and many more) for each asset from different classes. This thesis focus on model inputs to specific sets of financial assets, such as equities, exchange-traded funds (ETFs), commodities, cryptocurrencies, and bonds, among many other investment instruments such as derivatives and T-bills. To evaluate model performance and adaptability, we analyzed across two distinct time frames: (1) within the models' training data cutoff, and (2) from the cutoff date to the present (at the time of experiments for this Chapter). This approach enables the assessment of model generalization to past and present market conditions. The research quantifies LLMs' capabilities in financial asset allocation, comparing baseline performance against RAG-augmented strategies. Our results demonstrate that RAG-enhanced LLM significantly outperforms vanilla LLM in portfolio construction across various asset classes. We contemplate that these results could influence AI-driven financial decision-making processes such as automated trading, real-time sentiment analysis, and investment management.

4.1 Objective and Methodology

This Chapter aims to assess the impact of generative AI and LLMs on asset allocation and portfolio construction. Our study explores both the advantages and constraints of applying LLMs to portfolio construction, aiming to shed light on the future landscape of AI-enhanced financial strategies. The results of this research can help portfolio managers and people in the finance domain to make more accurate financial decisions. The utilization of LLMs in portfolio construction is a promising approach, given their ability to process and extract information from an immense database of structured and unstructured data. LLMs, such as the one employed in this research (Llama 3.1 from Llama Team, AI @ Meta [63]), have been trained on extensive corpora of texts, including Wikipedia pages, which equips them with a broad knowledge base in various fields, including finance. This pre-existing knowledge allows LLMs to understand and explain basic financial concepts, such as earnings per share (EPS) or the components of a balance sheet, with ease.

The transformer architecture, which forms the core of LLMs, plays a crucial role in their ability to tokenize and digest information efficiently. Transformer-based models use an attention mechanism to pinpoint relevant information in input data, a feature that makes them particularly adept at tasks like building portfolios and distributing assets.

However, one of the main challenges in utilizing LLMs for real-time financial decision-making is their knowledge data cutoff. In the case of Llama 3.1, the training data cutoff is December 31, 2023, meaning that the model's knowledge and answers are based on information up to this date. Given that LLMs do not have direct access to the Internet, they cannot provide real-time information, which is essential for making informed decisions in the rapidly evolving financial markets (At the time of writing this section, LLMs were primarily pre-trained models with static knowledge bases; however, by 2024 and into 2025, some models, such as ChatGPT, Gemini, and Claude, began incorporating full internet access or retrieval-augmented capabilities). Our key objectives in this research are to address these limitations and deficiencies through the following steps:

1. This research is divided into two phases. In the first phase, the efficacy of the LLM in asset allocation is evaluated through advanced prompt engineering and few-shot learning techniques. This approach aims to optimize the model's performance based on its existing knowledge and assess its potential for portfolio construction tasks.

2. In the second phase, we introduce a RAG framework to provide the model with up-to-date, large-scale financial data. This dataset contains 90,000 rows of information from top asset classes, ensuring that the model has access to the most recent and relevant data for making informed decisions. Zhang et al.[57] showed that RAG can enhance the models' understanding and accuracy.

By comparing the model's performance with and without RAG, our work aims to demonstrate the impact of up-to-date information on the quality of the LLM's answers and its ability to construct effective portfolios. The integration of RAG in the second phase of this research is a critical step in bridging the gap between the LLM's pre-existing knowledge and the real-time information required for optimal portfolio construction. By providing the model with a comprehensive and up-to-date dataset, researchers aim to enhance the LLM's decision-making capabilities and assess its potential for real-world applications in the finance domain.

4.1.1 Prerequisites, System Architecture, and Model Selection

As mentioned earlier, for this research, we use Meta's Llama 3.1, an open-weight (open-source) LLM, which sets it apart from other frontier LLMs. The open-source nature of the Llama family allows for greater transparency and accessibility, enabling researchers and practitioners to better understand and adapt the model to their specific needs. By utilizing Llama 3.1, we aim to explore the potential of this LLM in the domain of portfolio construction and asset allocation. Meta's LLaMA 3.1 model, featuring 8 to 405 billion parameters, is trained on data up to December 2023 with a context window of 128,000 tokens, enabling extensive context retention, with a maximum output of 4,096 tokens. The model was officially released on July 23, 2024.

While LLMs possess extensive knowledge, they may show deficiencies and generate hallucinations when queried. These issues primarily come from inadequate prompting techniques, limitations in their pre-training knowledge base, and lack of access to real-time data [52]. To address these challenges, this research contains two parts;

First, we begin by optimizing the interaction with the model through an advanced prompting technique. Prompt engineering and few-shot learning are two key techniques for guiding LLM's behavior [62]. The first one involves crafting specific input instructions to steer model outputs, while the second one uses exemplar cases to demonstrate desired task performance. This approach facilitates task-specific adaptation, improving the model's capacity to produce accurate financial analyses without additional training. We instruct the model with a prompt and a limitation on asset selection.

During the second phase, we enhance the results from the previous prompt by augmenting the model with the latest financial data. A large-scale dataset is gathered for this data, which is about 90,000 rows through the Yahoo! Finance[1] database and contains financial information such as statistical measures, financial metrics, balance sheets, historical data, income statements, and cash flows. We allowed the model to select assets from a certain pool of asset classes listed in (Table 4.1).

Lastly, we compare the gathered portfolios from both phases in two distinct time frames. Since the LLMs are trained on a vast corpora of texts and datasets, they have a training data cutoff. For the Llama 3.1, the training data cutoff is December 31, 2023. For this reason, we split the comparison into two time frames;

1. '2022-01-01' to '2023-12-31'

2. '2024-01-01' to '2024-09-25'

[1] https://ca.finance.yahoo.com/

Sector	Tickers
Basic Materials	LIN, BHP, RIO, SHW, SCCO, ECL, FCX, CTA-PB, APD, CRH
Financial Services	BRK-B, JPM, V, BAC, MA, WFC, GS, MS, AXP, BX
Consumer Defensive	WMT, PG, COST, KO, PEP, FMX, PM, UL, BUD, MDLZ
Utilities	NEE, SO, DUK, NGG, CEG, AEP, SRE, PCG, GEV, D
Energy	XOM, CVX, SHEL, TTE, COP, BP, PBR, ENB, CNQ, EOG
Technology	AAPL, MSFT, NVDA, TSM, AVGO, ORCL, ASML, AMD, ADBE, CRM
Consumer Cyclical	AMZN, TSLA, HD, TM, MCD, PDD, BABA, LOW, BKNG, TJX
Real Estate	PLD, AMT, EQIX, WELL, SPG, PSA, SPG-PJ, DLR, O, CCI
Healthcare	LLY, NVO, UNH, JNJ, MRK, ABBV, AZN, NVS, TMO, DHR
Communication Services	GOOG, META, NFLX, TMUS, VZ, DIS, CMCSA, T, DTEGF, NTES
Industrials	GE, CAT, UNP, HON, RTX, ETN, UPS, LMT, BA, DE
ETFs	SPY, IVV, VOO, QQQ, VTI, IWM, VWO, EEM, VEA, EFA
Commodities	GLD, SLV, USO, DBC, GSG, CORN, GDX, JJC, DBA, GLTR
Cryptocurrencies	BTC, ETH, ADA, XRP, SOL
Bonds	LQD, BND, TLT, JNK, HYG, GOVT, SCHZ, VCIT, IEF, BOND

Table 4.1: Asset classes

This action (of delimitation) allows us to analyze the LLMs in the past and present market conditions and assess the model's approach to the market's dynamics. For comparison, we analyze the portfolios with financial metrics that are mostly used for portfolio analysis, such as total return, Sharpe ratio, value-at-risk (VaR), alpha, beta, and a few more. We also compared the portfolios generated by the RAG-enhanced Llama with normally generated portfolios in terms of their diversification. One of the most important aspects of creating a financial portfolio is to keep the asset classes well-diversified. By diversification, one can mitigate the overall risk of the constructed portfolio.

4.1.2 Prompt Engineering

Prompt engineering is a vast sub-domain in natural language processing (NLP) area. It's the way one interacts with the language models (LMs) and how one can fetch better responses and results from the models. It guides the model to get accurate and relevant responses. The prompt engineering is divided into three main sub-classes, as we mentioned earlier in the previous Chapter (See Chapter 2):

1. **Zero-shot learning**: By this technique, you can utilize the model's ability to perform certain tasks on unseen data or without any prior examples. This method only relied on the model's general knowledge.

2. **Few-shot learning**: In this approach, we provide the model with a couple of examples of how the structure of the response should be. This action allows the LLM to think about its steps and respond better.

3. **Chain-of-Thoughts (CoT)**: In CoT, we steer the model's thinking capability with step-by-step reasoning before getting the final response. CoT often leads to more accurate results, but it is mostly used for complex reasoning tasks.

In scenarios where zero-shot prompting fails to reach optimal results, it is advisable to employ few-shot learning or CoT techniques. Should these methods prove insufficient, more advanced approaches such as model fine-tuning, RAG, or agents may be necessary to achieve satisfactory performance. In this study, we initially evaluate the language model's capacity to generate financial portfolios through few-shot prompting techniques. Subsequently, we augment and assess the model's performance using a RAG framework, which incorporates the most recent financial data. This approach lets us compare the model's baseline portfolio construction abilities with its enhanced performance when provided with up-to-date market information. In this research, we used the prompt shown in Figure 4.1 to assess the model's potential for asset allocation. We iterated this prompt 100 times to ensure that the model is consistent with its answer and can follow the instructions of the given structure. For iterations, we set the model's hyperparameters to "temperature=0.7", "top_p=1", and "max_tokens" to 4096. "Max_tokens" limits the number of tokens that the model can generate. As mentioned in previous Chapter "Temperature" controls the randomness in the output, and "Top_p" controls the probability distribution when sampling tokens. After a few experiments we determinded that the model works best on these parameteric values.

High-Reward Portfolio Construction Prompt

Role: You are a financial advisor specializing in constructing investment portfolios. **Task:** Using the provided list of stocks, ETFs, bonds, and commodities, create a hypothetical high-reward portfolio for an investor with high risk tolerance. The portfolio should have assets across different sectors and asset classes. Follow the example format below, but choose different assets for each portfolio.

Instructions:

- Mention the weight for each selection in the portfolio, and the total weight should be 100.00 (100%)
- Ensure the total of all weights equals exactly 100.00 (100%)
- Aim for higher returns than S&P 500 Index

Example Portfolio Format:

Portfolio : Aggressive Growth { "AAPL": 10.00, "NVDA": 8.00, ... "DBC": 2.00 }

Assets List: [List of assets]

- Write your tickers in dictionaries (curly braces) with their respective weights. For example, {"Ticker": Weight}.
- Please ensure the output is in plain text without any code formatting markers like ```json or ```python and output the tickers in double quotes with a comma separating each ticker and weight pair.
- The tickers should be formatted exactly in a dictionary like this example: {"AAPL": 10.00, "MSFT": 7.00, ...}.
- Make sure that the total of all weights equals exactly 100.00 (100%)

Figure 4.1: High-Reward Portfolio Construction Prompt

4.1.3 Implementing RAG With a Comprehensive Financial Dataset

By this framework, we enhance the models' outputs by incorporating relevant pieces of information from an external knowledge base. In this research, by creating a RAG we can allow the model to access up-to-date financial information without any fine-tuning or training from scratch. By acquiring relevant financial data before generating responses, the model can provide more accurate and specific information about the assets. This action reduces the hallucination and minimizes the risk of generating unreliable financial decision bsed on outdated information. The RAG technique offers a cost-effective and scalable alternative to expensive LLM fine-tuning or training from scratch, facilitating efficient deployment in dynamic financial environments.

As we mentioned before, in this research, we deploy Llama's RAG framework upon Groq's cloud service[2], LangChain[3], and NVIDIA NIM[4] for normal Llama usage through their API. Groq allows us to interact with models with much more speed than other services. LangChain also provides the necessary tools for document loading and chaining the process to streamline the RAG architecture. Our RAG framework contains several steps:

1. **Data loading and processing**: We load the financial data from a structured JSON file, converting it into a list of document objects, each document represents an asset ticker (for example Apple: "AAPL") and contains detailed financial information.

2. **Document Splitting**: In this step, we divide the documents into smaller chunks, ensuring that each piece of the document is manageable for the model. In our case, the chunk level is "4500" with a chunk overlap of "200".

3. **Vectorization**: A "TfidVectorizer" is deployed to transform the document chunks into numerical vectors.

4. **Retriever**: A custom retriever is utilized to find the relevant piece of documents based on cosine similarity between the query and the document vectors.

5. **Chaining the process**: We used LangChain's python package to combine the retriever and LLM's integration from Groq's API. After chaining the process, when a query is received from the user, it uses the "TfidRetriever" class to obtain a relevant document

[2]https://groq.com/
[3]https://www.langchain.com/
[4]https://ai.nvidia.com/

chunk and passes that document along with the query to the LLM to produce a valid response from retrieved information combined with the model's original knowledge base.

6. **Query processing**: In the last step, when a user submits the query, the system retrieves the relevant financial data, and allows the model to provide informed responses about the assets from the provided dataset.

4.1.4 Portfolio Selection

After completing both the few-shot and RAG-enhanced phases and collecting all generated portfolios, we determine a winner portfolio based on two key factors: the frequency of asset selection and the distribution of weights. This approach allows us to identify the most favored portfolio composition, focusing on the top 20 assets that consistently appear across the model's recommendations. By these results, we can construct a representative high-reward portfolio that reflects the model's most confident selections and weight allocations, potentially offering superior performance compared to individual portfolio suggestions. We are aware that risks for various classes of assets are different. However, through our prompts, we try to construct portfolios for high-reward or low-risk from these classes of assets by selecting only major tickers from S&P500 assets so that the effect of variation in their risk level could be mitigated to some extent.

4.2 Results and Analysis: Vanilla LLM vs. RAG-Enhanced LLM

After gathering the model's portfolio recommendations from both phases, we analyze the frequency of each ticker's appearance in the responses (Table 4.2). This comparison provides insight into the model's asset selection preferences.

1. We count how often each ticker is chosen across all portfolios in each phase.

2. By comparing these frequencies between the few-shot and RAG-enhanced phases, we can identify consistently popular assets and detect shifts in asset preferences when using RAG,

3. Lastly, we can assess the diversity of asset selection in each approach.

Table 4.2: Comparison of Asset Selection between Normal and RAG-LLaMA

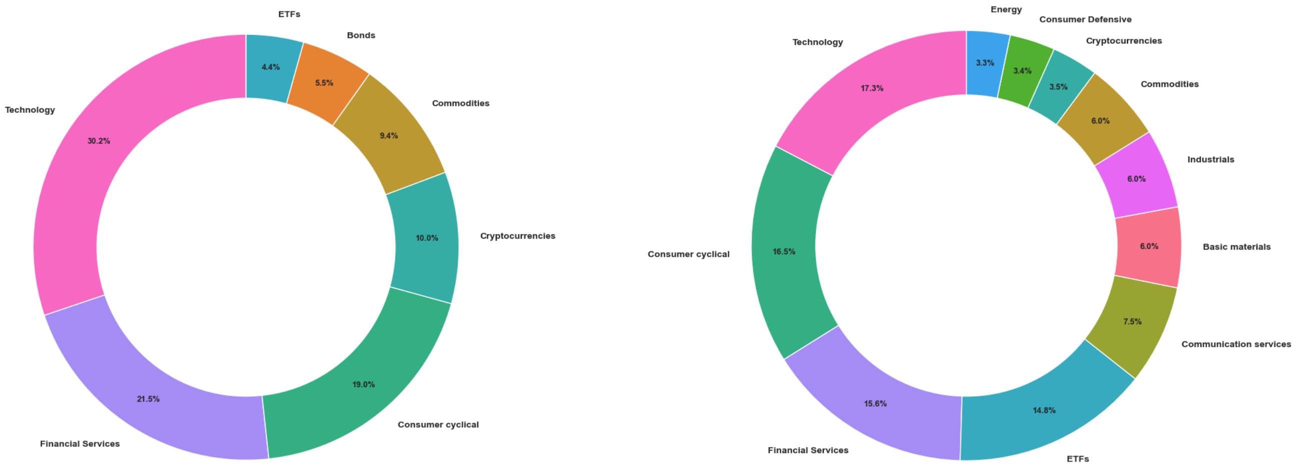
Normal LLaMA 3.1			**RAG LLaMA 3.1**		
Ticker	**Occur.**	**Avg. Weight**	**Ticker**	**Occur.**	**Avg. Weight**
AAPL	100	12.00	SPY	83	8.63
NVDA	100	9.98	DIS	71	7.71
AMZN	100	8.98	NVDA	101	7.17
TSLA	100	7.99	QQQ	96	6.63
BRK-B	100	5.03	TSLA	79	6.24
MSFT	99	5.01	SHW	91	6.22
GLD	100	4.97	RTX	62	6.15
JPM	98	3.98	AMZN	92	6.09
BTC-USD	100	3.97	AAPL	92	5.66
BAC	99	3.97	MSFT	53	5.03
WFC	98	3.48	HD	58	4.70
DBC	100	3.45	BRK-B	58	4.60
MS	97	3.00	JPM	45	4.14
LQD	98	2.99	GLD	64	3.74
ETH-USD	100	2.50	V	62	3.68
TLT	98	1.99	GS	48	3.59
EEM	98	1.98	PG	60	3.51
EFA	98	1.98	BTC-USD	60	3.40
SOL-USD	100	1.50	XOM	86	3.37
XRP-USD	100	1.00	DBC	72	2.40

Table 4.3 demonstrates the statistical metrics for these two comparisons.

With these details and information, we infer how the vanilla model worked versus the RAG-enhanced model as presented in Figure 4.2 and discussed next.

1. RAG-Llama created a more diversified portfolio with asset classes spread across more categories. The normal Llama was heavily concentrated on the Technology sector with 30.2% and Financial Services with 21.5%, while RAG-Llama had a more balanced distribution.

2. RAG-Llama showed better risk management with a lower standard deviation (1.59 vs.



(a) LLaMA 3.1 Diversification

(b) RAG-LLaMA 3.1 Diversification

Figure 4.2: Portfolio Diversification Comparison by Asset Category

Table 4.3: Statistical Metrics Comparison

Metric	**RAG-Llama 3.1**	**Normal Llama 3.1**
Total Value	100.07	100.00
Number of Assets	20.00	20.00
Average Weight	5.00	5.00
Maximum Weight	8.39	13.45
Minimum Weight	2.34	1.11
Weight Range	6.05	12.34
Standard Deviation	1.59	3.29

3.29), indicating less volatility in asset weights. The weight range for RAG-Llama is much smaller (6.05 vs. 12.34), suggesting more balanced allocations.

3. RAG-Llama maximum weight (8.39%) is significantly lower than normal Llama's (13.45%), avoiding over-concentration in any single asset.

4. RAG-Llama includes sectors like Energy, Consumer Defensive, and Industrials, which are absent in normal Llama.

The RAG-Llama model has constructed a more diversified, balanced, and potentially lower-risk portfolio with maintaining an aggressive strategy, compared to the normal Llama model. The

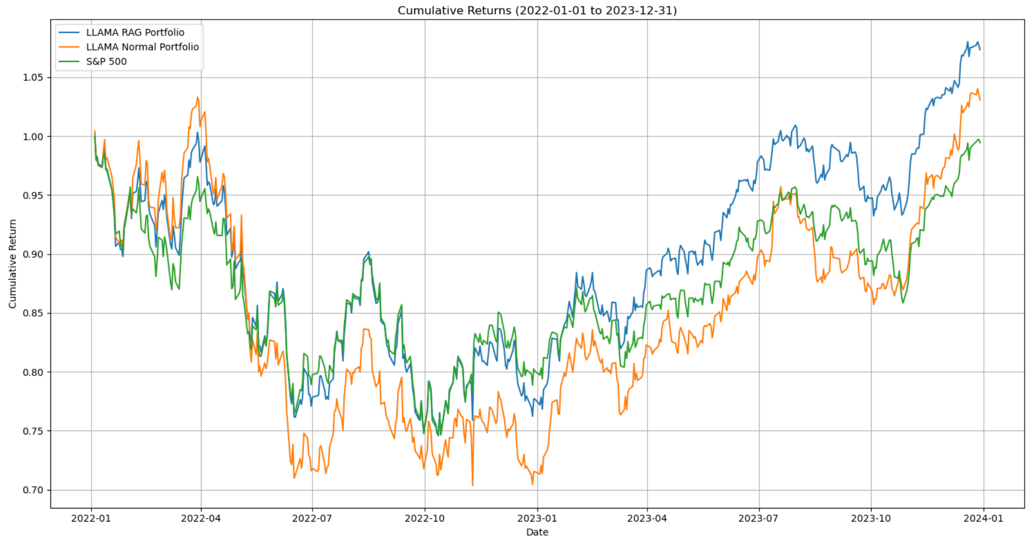
RAG approach seems to have led to a more sophisticated asset allocation strategy, considering a broader range of sectors and maintaining more moderate position sizes. This suggests that the RAG method may have incorporated more comprehensive market information and risk management principles in its portfolio construction process.

Now, if we compare the models with financial metrics, we can better understand how models work and assess their efficiency.

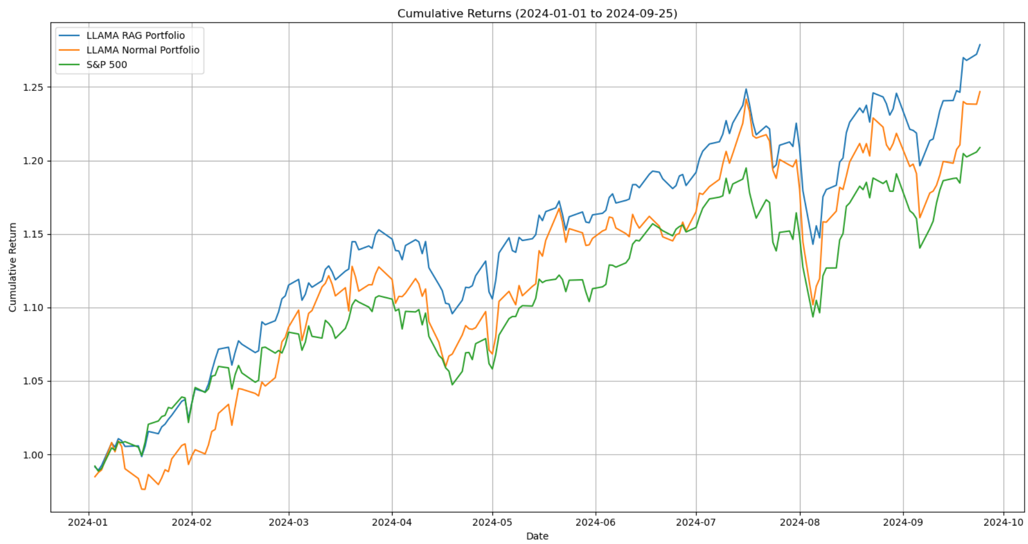
Table 4.4: Portfolio Performance Metrics Comparison for Two Time-frames

	2022-2023			**2024 (YTD)**		
Metric	**RAG**	**Normal**	**S&P**	**RAG**	**Normal**	**S&P**
Total Ret.	7.33%	3.08%	-0.56%	27.87%	24.68%	20.88%
Ann. Ret.	3.63%	1.54%	-0.28%	40.29%	35.49%	29.83%
Ann. Vol.	20.51%	22.75%	19.49%	12.81%	15.84%	12.82%
Beta	1.0177	1.0301	1.0020	0.9351	0.9577	1.0055
Alpha	2.00%	-0.09%	-1.90%	15.04%	9.65%	2.75%
Sharpe	0.1283	0.0237	-0.0657	3.0678	2.1767	2.2485
Max DD	25.68%	32.96%	25.36%	10.57%	14.00%	10.14%
VaR	-2.09%	-2.16%	-1.90%	-1.25%	-1.52%	-1.37%

Based on the financial metric from Table 4.4, in the 2022-2023 period, the RAG-Llama portfolio significantly outperformed both the normal Llama's portfolio and the S&P 500 benchmark during a challenging market environment. The RAG-Llama portfolio achieved a positive total return of 7.33% and an annualized return of 3.63% (138% improvement). The RAG-Llama portfolio also demonstrated superior risk-adjusted performance with the highest Sharpe ratio (0.1283) and alpha (2.00%), indicating better returns per unit of risk. Although its volatility was slightly higher than the S&P 500, it managed downside risk better with a lower maximum drawdown than the normal Llama's portfolio. The 2024-YTD (present market situation) period shows a marked improvement in overall market conditions, with all portfolios delivering strong positive returns. The RAG portfolio continues to outperform, with the highest total return (27.87%) and annualized return (40.29%). Figure 4.3a and Figure 4.3b demonstrate the total return for both periods. RAG-Llama also maintains its edge in risk-adjusted performance, keeping the highest Sharpe ratio (3.06) and alpha (15.04%). Importantly, the RAG framework achieves this with the lowest annualized volatility (12.81%) and beta (0.93) among the three, suggesting excellent risk management. The consistent outperformance of the RAG-Llama portfolio across both periods, especially its ability to generate positive returns in the challenging 2022-2023 market, indicates that the RAG approach may offer a robust and adaptable strategy for portfolio construction and management across varying market conditions.



(a) Total Return 2022-2023



(b) Total Return 2024-YTD

Figure 4.3: Total Return For Both Portfolios

4.3 Results and Insights

We can conclude that RAG demonstrates superior performance in portfolio management and asset allocation. By integrating an external knowledge base to keep the model updated with the latest financial data, we can significantly enhance our efficiency in these domains. As previously discussed, a primary limitation of most LLMs in providing accurate financial responses is their restricted knowledge base, despite their proficiency in financial terminology. This research illustrates that equipping an LLM with the latest and relevant data can lead to improved performance.

The research reveals the potential of combining the capabilities of LLMs with real-time financial data, creating a more dynamic and responsive system for portfolio construction. The RAG approach effectively bridges the gap between the model's vanilla knowledge and the rapidly changing financial market, resulting in more informed and timely investment decisions.

Still, there are certain limitations to this approach. Notably, during the final stage of the research, we had to normalize the weights for both portfolios as the LLMs were unable to consistently maintain a sum of 100% for the assets weights. This adjustment was necessary to ensure portfolio feasibility and comparability. This issue is addressed in the multi-agent workflow in the next Chapter and our third and final phase of this research (See Chapter 5).

Furthermore, while the RAG framework enhances performance, it does not fully resolve the 'black box' nature of LLMs. Even with the integration of external knowledge bases, the internal decision-making processes of these models remain vague and unexplainable. This lack of transparency in the model's reasoning and decision pathways continues to be a challenge, particularly in the context of financial decision-making, where explainability is often crucial.

These findings suggest that, while the RAG approach represents a significant advancement in applying LLMs to financial portfolio management and asset allocation, there is still room for improvement in areas such as weight consistency and model interpretability.

Chapter 5

Agentic AI Framework For Portfolio Management

In the third and final phase of our research, which represents the most comprehensive segment of the study, we implemented an agent-based methodology to significantly enhance the capabilities of LLMs in finance and portfolio management. Earlier phases of our work primarily leveraged LLMs in isolation or by RAGs, relying exclusively on their pretrained internal knowledge or an augmented dataset. In contrast, this phase integrates LLMs within a coordinated system of agents designed to execute complex workflows through interaction with one another and with external tools such as APIs, functions, or other LLMs.

5.1 Introduction to Multi-Agent Systems

Each agent within this system is tailored to perform a specific role and equipped with a defined objective and specialized functionality. These agents collaborate by sharing intermediate outputs, validating information, and routing tasks based on their expertise, thereby enabling a more structured and adaptive computational process.

5.1.1 Definition and Relevance in Finance

Our research centers on a singular, well-defined application: portfolio management. Given the intricacies of financial markets, including high data dimensionality, volatile conditions, and the necessity for rapid, evidence-based decisions, LLMs alone are often insufficient. By embedding them in an "Agentic AI" framework, we can better manage financial complexity, improve decision accuracy, and introduce modular workflows that could support more effective asset allocation and risk management.

This structured integration of agents marks a critical advancement in the deployment of LLMs for financial intelligence and autonomous decision-making. Previous research, including our own earlier investigations, have shown that relying exclusively on LLMs with static, outdated training data, without access to external reasoning mechanisms, can produce only reasonably good results in financial tasks. In other words, these outcomes often fall short of the accuracy and adaptability required for high-stakes decision-making. By adopting an agentic approach that incorporates multiple stages of reasoning, data retrieval, and quantitative analysis prior to arriving at a conclusion, we could achieve significantly more robust and precise financial decisions. This multi-step architecture allows agents to validate assumptions, compute relevant metrics, and iteratively refine their outputs, ultimately enhancing both the reliability and performance of the system.

Traditional approaches to portfolio management typically relied on a single, centralized source of information, requiring analysts to manually collect, compute, and verify financial data. These conventional systems, though methodical, were often limited in scope, inflexible to rapidly changing market dynamics, and prone to delays in decision-making. With the advent of LLMs, advancements in NLP, and the integration of external agents capable of real-time web browsing and data extraction, portfolio management has become significantly more adaptive. These tools enable continuous access to up-to-date financial news and broader market insights, facilitating more informed and timely investment decisions.

5.1.2 Agent Roles and Interactions

In this Chapter, we provide an overview of each node, whether they are AI-agent nodes (LLM-based) or function-based nodes. In the proposed framework (see Figure 5.4) for agentic portfolio construction using LangGraph, we implemented a set of internal and external agents, each with defined responsibilities, to facilitate and ultimately achieve our goal of intelligent portfolio management. At the core of our approach, we utilized ChatGPT-4o by OpenAI as the primary LLM, serving as the main engine behind the system's reasoning and decision-making capabilities. To supplement this, we integrated Tavily for real-time Internet browsing and web scraping in order to gather the latest financial news. Tavily Search API is a specialized search engine designed for LLMs and AI agents. It provides real-time, accurate, and unbiased information, enabling AI applications to retrieve and process data efficiently [1].

Client Profiler Agent (LLM-based):

This agent utilizes GPT-4o to parse natural language investment requests submitted by the user. It extracts structured investment attributes, including goals (e.g., maximizing return, saving for retirement), risk tolerance (e.g., conservative, moderate, aggressive), time horizon, and initial capital. You can see the prompt we used to fetch the initial data and form the client profile on Figure 5.1.

Additionally, it interprets user-defined preferences such as personal considerations (e.g., avoid fossil fuel industries and focus on renewable energy industries such as solar, wind, or tidal power) or asset type constraints. If the user provides fewer than five asset tickers, or none at all, the agent generates a diverse and contextually appropriate list of approximately 15–20 suggested assets. The output is a structured user profile and a curated asset universe that initiates the downstream decision-making pipeline. An important design philosophy of the Client Profiler Agent is its ability to incorporate investment preferences that align with broader principles of equity, diversity, and inclusion (EDI). Investors can choose to support businesses that reflect their cultural, religious, or social values. For instance, they may prefer companies that follow

[1] https://www.tavily.com/

halal financial principles, organizations that actively support Indigenous communities, firms that promote racial equity and minority empowerment, those with strong commitments to LGBTQ+ inclusion and rights, or businesses that avoid unethical labor practices or involvement in military manufacturing.

These considerations are not hard-coded; instead, they are derived directly from the user's request or prompted through initial user input. This flexibility ensures that the portfolio construction process is inclusive, adaptable, and ethically aware, avoiding the imposition of a predefined worldview. The system's architecture is designed to embrace diverse perspectives and individual investment values without judgment or restriction.

Market Data Analyst Agent (External-Agent):

This is a function-based component responsible for aggregating real-time market data and contextual financial news. It fetches historical asset prices using the API from Yahoo! Finance and retrieves relevant market headlines through the Tavily API (TavilySearchResults tool). Tavily is responsible for browsing the internet and scrap the data that is necessary to facilitate our pipeline. The data collected serves as the empirical foundation for financial analysis and is aligned with the user's investment objectives and the identified asset universe.

Quantitative Analyst Node:

This node computes a comprehensive set of financial metrics for both individual assets and the proposed portfolio. Key outputs include annualized returns, volatility, Sharpe ratio, maximum drawdown, beta (manually computed when needed, because of the limitations from Yahoo! Finance API, it doesn't return "beta" metric for ETFs or some asset classes, so we had to implement a code to compute beta manually.), CAPM-based expected returns, and technical indicators such as SMA 50 and SMA 200. These metrics are essential for evaluating asset quality and are used by the Portfolio Manager Agent to inform allocation decisions.

Portfolio Manager Agent (LLM-based):

This agent, which is GPT-4o, synthesizes the structured user profile, quantitative financial metrics, and real-time market context to propose an optimized portfolio allocation. It uses an LLM to reason about expected returns, risk exposure, momentum indicators, and user-defined constraints. The output is a dictionary mapping tickers to allocation weights, ensuring the portfolio is diversified, balanced, and aligned with the investor's profile. It also generates a concise natural language explanation of the allocation rationale. On Figure 5.2, you can see the detailed prompt we used to curate the portfolios by the LLM-Agent.

Validation Node:

This function-based node validates the portfolio allocation by performing logical and mathematical checks. It verifies that weights sum to one, checks for invalid or missing metrics, and ensures no structural inconsistencies exist in the portfolio. Although not LLM-based, this step is essential for enforcing investment constraints and detecting flawed allocations before final reporting.

Reporting and Commentary Agent (LLM-based):

This agent employs the LLM to generate a detailed, human-readable summary of the final portfolio. The report includes a recap of the user profile, the proposed allocation, key financial metrics, validation results, and interpretative commentary. It translates structured outputs into accessible insights, making the system's recommendations understandable to a non-technical user or stakeholder. In Figure 5.3, you can find the prompt that the agent is using to write a comprehensive report for the gathered portfolios.

Error Handling Node:

This function-based node is activated when any part of the workflow fails. It logs the error, formats a markdown-compatible error report, and optionally attaches relevant context such as the user profile and last known portfolio state. This ensures the system can gracefully recover from unexpected interruptions and provide meaningful feedback to the user or developer.

Within our design, LLM agents directly leverage GPT-4o to interpret user input, perform reasoning over structured and unstructured data, and generate decisions. These agents invoke the LLM to carry out tasks that require context understanding and natural language-based judgment.

Client Profiler Agent Prompt

Role: You are an expert financial analyst assistant.

Task: Parse the user's request to understand their investment profile. Extract the goal, risk tolerance, time horizon, initial capital, and any specific preferences mentioned (store simple preferences as a string in `'specific_preferences'`). Identify any specific assets the user suggested.

Asset Generation Rules:

- If the user explicitly requests a specific number of tickers (e.g., "select 20 tickers", "give me 10 stocks"), prioritize fulfilling that exact number. Generate a diverse list matching the profile to meet the requested count.
- If the user suggests 5 or more specific assets, use those primarily, potentially adding more diverse assets to reach a count of around 15 if the user didn't specify a number.
- If the user suggests fewer than 5 assets AND does not specify a number, generate a diverse list of approximately 20 suitable assets (considering stocks, bonds, ETFs relevant to the profile).

Output Requirements:

- Populate the `'suggested_assets'` field with the final list of tickers.
- Ensure the list contains **only valid tickers**.
- Output **only the JSON object** matching the required schema.
- **Do not include any comments** (like `//`) inside the JSON output.
- Today's Date: *`Today's date Code`*

User Input Instruction:

Here is the user request: `{request}`

Output ONLY the JSON object matching the required schema. Ensure NO comments are included in the JSON.

Figure 5.1: Client Profiler Agent Prompt used to extract structured investment profiles from user input

Portfolio Manager Agent Prompt

Role: You are an expert portfolio manager.

Task: Your task is to propose a portfolio allocation based on the user's profile, available asset data metrics (including historical performance, CAPM expected return, and SMA indicators), and recent market news.

Inputs Provided to the Agent:

- **User Profile:** structured fields such as goal, risk tolerance, time horizon, and preferences
- **Available Assets with Data:** a list of validated tickers
- **Asset Metrics Summary:** historical return, volatility, Sharpe ratio, drawdown, beta, CAPM expected return, SMA 50/200, and portfolio momentum outlook
- **Recent Market News Context:** headline summaries retrieved via external API

Constraints:

- Allocate **only** among the listed available assets
- Proposed weights **must sum to 1.0** (or very close)
- Consider the user's investment goal and risk tolerance as primary drivers
- Incorporate CAPM expected return and SMA-based momentum signals
- Provide brief reasoning with the portfolio

Output Requirements:

- Output ONLY the JSON object matching the required schema
- Ensure ticker symbols match the asset list exactly

User Input Instruction:

Based on the provided information, propose a suitable portfolio allocation and provide reasoning, considering historical metrics, expected returns, and momentum.

Figure 5.2: LLM prompt used by the Portfolio Manager Agent to generate allocation strategy

Reporting Agent Prompt

Role: You are a financial advisor AI.

Task: Generate a clear and concise commentary explaining the proposed portfolio allocation, its key metrics (including historical performance, CAPM expected return, and momentum outlook), and the validation results. Incorporate reasoning if it was provided by the allocation model, and clearly report any issues if validation failed.

Context Provided:

- **User Profile** – risk, goals, capital, time horizon, and preferences
- **Proposed Portfolio** – dictionary of assets and weights
- **Portfolio and Asset Metrics** – historical return, volatility, Sharpe ratio, drawdown, CAPM expected return, SMA-based momentum
- **Validation Result** – pass/fail status and list of issues (if any)
- **Market News Context** – recent relevant news headlines
- **Initial Allocation Reasoning** – rationale from the portfolio proposal (if provided)

Instructions to the Model:

- Explain the portfolio's reasoning based on user profile, metrics, and news
- Interpret the most important metrics clearly
- If validation failed, summarize the reasons
- Keep the tone informative and objective
- **Include a disclaimer that this is not financial advice**
- Output only the commentary text

User Input Instruction:

Please provide the commentary for the generated portfolio report based on the context, including interpretation of the new CAPM and momentum metrics.

Figure 5.3: LLM prompt used by the Reporting Agent to generate final portfolio commentary

5.2 System Architecture and Design

In this Chapter, we review the overall system architecture and describe the design and implementation of agents and internal function nodes that collaboratively construct an optimal financial portfolio based on user-defined preferences and investment goals. The architecture leverages both language model agents and internal function units, organized within a LangGraph-based workflow. We begin by providing an overview of the LangChain and LangGraph libraries, which enable modular LLM integration and stateful agent orchestration. We then discuss the role of centralized state management in controlling the flow of data and decisions throughout the graph. Finally, we detail the construction of the workflow graph itself, including the conditional routing logic that governs transitions between nodes and handles fault tolerance within the system.

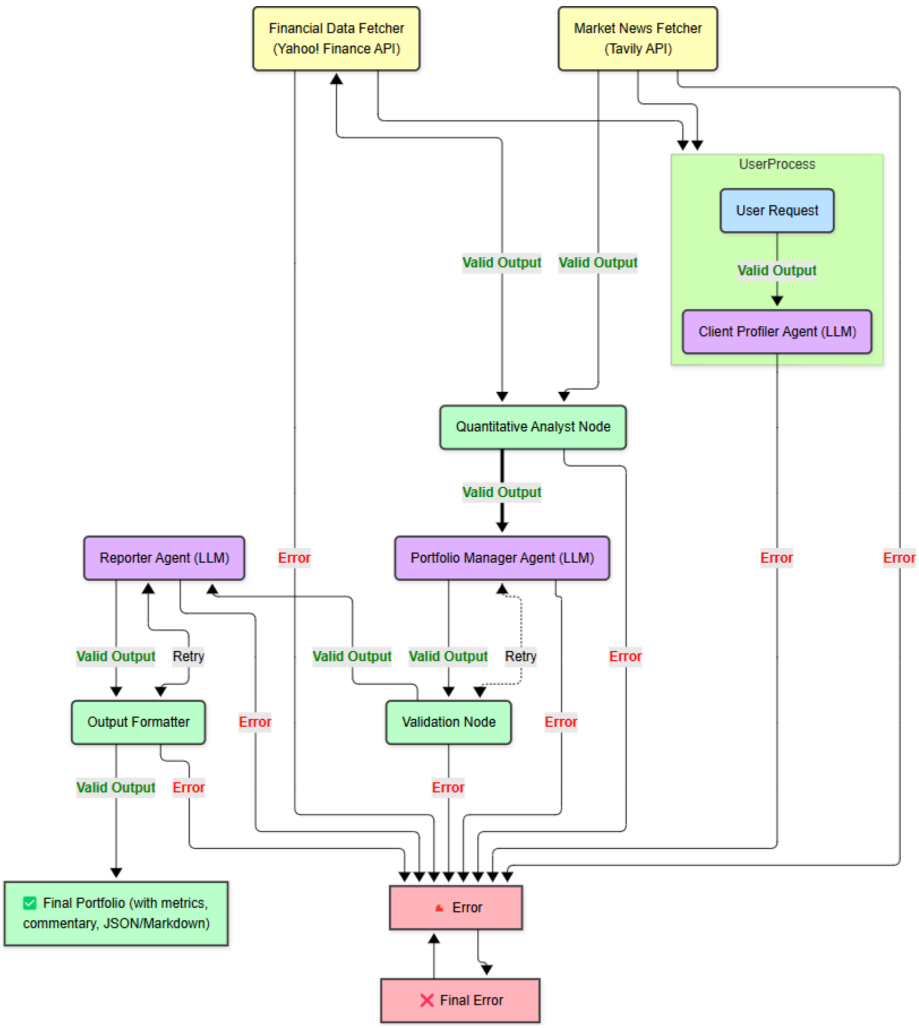


Figure 5.4: The Multi-agent architecture for Portfolio Construction

5.2.1 LangChain and LangGraph Overview

The framework employs LangChain and LangGraph as the primary orchestrators for interacting with LLMs, enabling structured NLP, response generation, and schema-constrained parsing. LangChain allows each agent to invoke LLMs through prompts and output parsers, ensuring that responses are both syntactically valid and semantically aligned with downstream tasks.

To coordinate the interaction among these agents, the system uses LangGraph, a stateful orchestration engine designed for agentic workflows. LangGraph facilitates modular and deterministic routing of information between nodes based on the evolving internal state. This state-driven architecture enables the construction of a clear, scalable, and fault-tolerant workflow where agents can reason, communicate, and operate autonomously within a tightly managed execution graph.

5.2.2 State Management

The state manager in LangGraph serves as the central data structure and a memory that maintains and propagates relevant information across all stages of the portfolio construction workflow. It encapsulates the initial user request, along with the parsed investment profile, comprising the user's objectives, risk tolerance, time horizon, and preferences. As the workflow progresses, this state is incrementally updated to include the selected assets for analysis (i.e., the asset universe), retrieve market data such as historical price series and financial news, and compute financial metrics including returns, volatility, Sharpe ratios, and CAPM-based estimates. It also stores the proposed portfolio allocation, the results of validation checks, LLM-generated commentary, and any encountered error messages. This unified state ensures consistent data access, facilitates decision-making across agents, and supports robust error recovery throughout the system.

5.2.3 Workflow Graph and Conditional Routing Overview

The system architecture is built upon a state graph, which defines explicit pathways between agent nodes, allowing for transparent and controlled execution of the multi-agent workflow. This graph-based structure models the sequential and conditional logic governing the portfolio generation process. Each node in the graph represents a discrete agent or functional component, and transitions between nodes are governed by conditional routing logic that evaluates the current state.

Under normal execution, the workflow progresses in a linear and modular fashion, beginning with user profiling, followed by data acquisition, metric computation, portfolio proposal, validation, commentary generation, and final reporting. At each stage, the system performs integrity checks on the evolving state. If inconsistencies, missing data, or errors are detected, control is dynamically rerouted to a dedicated error-handling node. This mechanism ensures both robustness and fault tolerance, allowing the framework to gracefully handle unexpected conditions while preserving traceability and state continuity.

5.3 Implementation Details

5.3.1 Data Collection: Integration with Yahoo! Finance and Tavily Search, Processing Natural Language Investment Requests

In addition to internal functions and LLM agents, our framework includes external agents that directly interface with real-world systems and data sources. These agents are essential for bringing in timely and relevant information that the LLM-based agents use for decision-making.

- Fetching Market News Node

Leverages Tavily API to gather external market news based on the user's investment goals and the identified asset universe.

- Fetching Financial Data Node

Uses the Yahoo! Finance API to retrieve historical financial data, including price histories and relevant asset fundamentals. These components serve as crucial integration points with external systems, ensuring that our framework remains dynamic and up-to-date.

5.3.2 Quantitative Analyst Node: Key Metrics Calculated and Portfolio Validation Overview

There are also non-agentic components within the system. These are utility functions or deterministic nodes that, while vital, do not possess autonomous reasoning capabilities. For this phase we have two nodes: the first is a node to compute financial metrics, and the other is a node to validate the computations and weights.

Although these functions perform essential computations, such as evaluating returns, volatility, and portfolio validation, they operate deterministically and do not interact with the LLM model outside of the workflow or external APIs. Therefore, they are best viewed as supportive modules that enable the agents to function effectively rather than as agents themselves. The metrics and computations from these nodes and other internal nodes are passed into the graph and Agents (LLM-nodes) to facilitate their reasoning and operations.

The computation and validation nodes are responsible for computing a comprehensive set of financial metrics essential for evaluating both individual assets and the overall portfolio. These include annualized returns, which standardize performance across time; volatility, as a measure of price dispersion and investment risk; and the Sharpe ratio, which quantifies risk-adjusted returns. It also calculates maximum drawdown to capture the extent of peak-to-trough losses. Additionally, the node derives Capital Asset Pricing Model (CAPM) metrics, including expected returns and Beta coefficients, the latter of which are externally computed when sufficient benchmark data is available.

To assess technical momentum, this internal node computes simple moving averages (SMA), specifically the 50-day and 200-day indicators, which are two of the industry standard indicators for technical analysis. After metrics computation, a list of metrics with comprehensive data will be passed to a LLM Agent node to construct a portfolio based on the gathered data.

5.3.3 AI Decision Making: Parsing Investment Requests, Generating and Validating Portfolio Allocations, and Commentary Generation

In the proposed framework, AI-driven decision-making is distributed across three key stages, each delegated to a dedicated LLM-based agent. These stages include interpreting user intent, generating a portfolio allocation strategy, and composing a human-readable explanation of the final recommendations. This sequence ensures the system maintains a high level of autonomy, transparency, and adaptability across the entire portfolio construction process.

1. **Parsing Investment Requests:**

   The first LLM-Agent decision-making task is performed by the Client Profiler Agent, which leverages GPT-4o to parse natural language input provided by the user. The input may typically include an initial capital amount, investment goals (e.g., growth, retirement), time horizon, risk tolerance, and any specific preferences or constraints (such as ESG compliance or sector focus). The LLM processes this unstructured text using a predefined prompt template, extracting structured fields to create a coherent investment profile. If the user does not explicitly specify a list of assets to include, the model is instructed via the prompt to generate a diverse list of suitable tickers (stocks, ETFs, or bonds) aligned with the user's profile. This output is parsed using a strict JSON schema, ensuring data consistency and validation before it proceeds to the next stages. This LLM agent thus functions as an intelligent interpreter, bridging the gap between human intent and machine-readable state data.

2. **Portfolio Generation and Validations**:

   When the user profile, market data, and financial metrics are in place and computed, it will be passed to the portfolio manager agent, which is another LLM-Agent to reason over the combined context and propose a portfolio allocation that is both quantitatively sound and tailored to the user's constraints. The model is explicitly prompted to consider risk-reward trade-offs, technical signals (e.g., SMA crossover), and expected market performance, and it is constrained to output weights summing to 1.0 (100%). The output is a dictionary, mapping asset tickers to allocation weights, along with a short natural language explanation. The system also normalizes weights if necessary and filters out assets lacking sufficient data. After processing the metrics and portfolio creation from previous nodes, the validation node performs rigorous validation procedures to ensure portfolio integrity. These checks verify that asset weights sum to one, detect and flag the presence of negative weights, and confirm the presence and correctness of all required portfolio-level computations. This combined process ensures that proposed allocations are both quantitatively sound and structurally valid. This validation step adds a critical safeguard, preventing flawed or non-compliant portfolios from moving forward in the workflow.

3. **Commentary Generation:**

   Lastly, the reporting agent, which is a LLM-based agent, is responsible for a comprehensive commentary that summarizes the entire portfolio construction process. This agent employs the LLM to analyze the user profile, the proposed portfolio, computed metrics, validation results, and market news, and transform them into a clear, human-readable narrative. The commentary addresses key aspects such as allocation rationale, asset performance, risk-adjusted returns, CAPM-based expectations, and portfolio momentum trends. This agent ensures that end users (especially non-technical stakeholders) can understand the rationale behind the AI's decisions. The commentary includes an automatic disclaimer clarifying that the output is not financial advice, reinforcing responsible usage of the generated insights.

5.4 System Workflow Overview

The portfolio generation system's architecture is centered on a state-driven execution graph, which regulates both data flow and decision-making processes among agents and functional nodes. LangGraph is used to coordinate this modular workflow, allowing for conditional routing that adapts based on runtime state evaluations, thus ensuring robust processing from initial user input to the final output.

To explain the whole architecture in summary, the workflow begins with a natural language investment request, which is parsed by an LLM-based Agent. The output, which is a structured user profile and a list of relevant assets, triggers subsequent steps in the pipeline. The system then invokes an external agent to retrieve recent financial news using the Tavily API, followed by another node that collects historical financial data via the Yahoo! Finance API. Once this data is retrieved, the Quantitative Analyst Node computes essential financial metrics, including CAPM-based expected returns, volatility, Sharpe ratios, and momentum indicators. Utilizing this data, a Portfolio Manager agent powered by GPT-4o suggests an optimized portfolio, which the validation node subsequently verifies. The portfolio must meet all structural and metric-based criteria, and afterward, a reporting Agent provides a commentary in natural language outlining the outcomes, culminating in a formatted report.

Chapter 6

Analysis and Discussion

In this Chapter, we present an analysis of the results from the previous Chapter(s), evaluating the performance and effectiveness of the proposed multi-agent framework in relation to the two earlier approaches: few-shot LLM prompting and RAG. We demonstrate the relative strengths and limitations of each method and highlight the scenarios in which agent-based systems offer distinct advantages. We also evaluate our framework againts a real-world portfolio manager to illustrate the precision and accuracy of the system in creating diverse portfolios based on different client profiles.

In the preceding Chapter(s) and Chapter 3, we initially evaluated vanilla pre-trained LLMs (See Table 3.1) using advanced prompt engineering techniques to assess their ability to handle asset allocation tasks. Specifically, we examined how effectively these models could curate and maintain a portfolio based on a structured few-shot prompting instruction. Our analysis revealed that while LLMs could select reasonable assets from different classes (See Table 3.4), they exhibited deficiencies in assigning optimal weight distributions, necessitating post-hoc normalization of the generated weights to ensure a rigorous and fair evaluation. Additionally, we identified both strengths and limitations inherent to this approach; however, the overall results were not sufficiently robust for practical applications. Consequently, this motivated the second phase of our research, in which we implemented a RAG framework to enhance the models' performance and mitigate the risks and issues observed in the initial vanilla prompting methodology.

In the second phase of our research and Chapter 4, utilizing RAG, we demonstrated that LLMs enhanced with external knowledge sources exhibited improved reasoning capabilities and achieved greater accuracy in asset allocation tasks. Specifically, we showed that the RAG-LLaMA model was able to construct more diversified (See Figure 4.2a), lower-risk, and higher-return portfolios compared to the baseline vanilla LLaMA model (See Table 4.4). Diversification, a critical principle in portfolio management for mitigating portfolio-specific risks, was significantly better achieved under the RAG framework. Furthermore, the portfolios generated by the RAG-LLaMA exhibited superior Sharpe ratios and higher annualized returns relative to the non-augmented model. Despite these improvements, the RAG-enhanced approach still exhibited certain limitations, particularly in optimal weight assignment, which prevented it from fully meeting our objectives. This motivated the third phase of our research, focused on developing and implementing a multi-agent framework for portfolio management to further enhance system performance and decision quality.

In the third and final phase of our research (See Chapter 5), we aimed to address the limitations and risks associated with previous approaches by introducing a novel multi-agent framework. This system is capable of reasoning, performing calculations, maintaining portfolio structure, optimizing allocations, and delivering comprehensive asset management solutions based on user-defined instructions. Following the introduction of the multi-agent framework in the previous Chapter, we discuss its performance against the vanilla-LLMs and RAG through key metrics and performance. This approach successfully mitigates critical issues related to weight assignment, improves portfolio diversification, and ensures adherence to ethical standards and user preference constraints during the asset allocation process.

6.1 Findings and Analysis

In this subChapter, we illustrate the effectiveness and performance of the multi-agent framework in generating diverse financial portfolios tailored to various user profiles and instruction sets. We begin by comparing the framework with two baseline methods from earlier phases of this research: prompt engineering with vanilla LLMs and RAG-enhanced LLMs. Following this, we demonstrate how the multi-agent system responds to different investment instructions and asset preferences, showcasing its adaptability and reasoning capabilities. To provide a practical benchmark, we use Wealthsimple's portfolio classifications and public insights as a reference point, replicating their asset class structures where appropriate. It is important to note that the performance results presented in the third subChapter are hypothetical and intended solely for research purposes; this limitation is discussed in detail in the corresponding section.

6.1.1 Performance Comparison (LLM, RAG-enhanced LLM, multi-agent system)

Multi-Agent System Vs. Vanilla LLMs with Prompt Engineering (Phase 1)

We first demonstrate the performance of the multi-agent framework in comparison to the two previous approaches by utilizing the same user instructions and prompting structure. In the initial experimental setup in Chapter 3, the models were tasked with constructing two distinct portfolios based on the given prompt: (1) a High-Growth S&P 500 Outperformer Portfolio, and (2) a Low-Risk S&P 500 Outperformer Portfolio. In this baseline evaluation, the LLaMA 3.1 model and Claude 3.5 Sonnet achieved the best results among the tested LLMs, delivering more effective asset allocations in alignment with the target objectives.

In our multi-agent framework, the LLM agents are instructed with nearly the same prompt as used in earlier phases, but now embedded within dedicated *system prompts* (See Figure 5.1). We employ the same core instructions to guide the models in constructing portfolios, ensuring consistency across all experimental phases while leveraging the enhanced reasoning and modular capabilities of the agentic workflow.

Now, we attempt to utilize our multi-agent framework with the same underlying task; however, this time, the input is provided as a user instruction rather than an engineered prompt. The instruction conveys the portfolio objectives directly, allowing the agents to parse, reason, and generate the portfolios autonomously within the structured system workflow:

1. **Create a high-growth portfolio from S&P 500 companies that aims to outperform the index in returns. Select around 20 tickers focusing on aggressive and growth potential.**

2. **Create a low-risk portfolio from S&P 500 companies that aims to outperform the index with minimal volatility. Select around 20 stable, defensive tickers.**

Through our initial instructions to the multi-agent framework, we configure the following settings (see Figure 6.1):

1. We assign a hypothetical initial investment amount of $100,000.

2. The investment time horizon is set to two years. Given the framework's access to real-time financial data, there is no need to separately evaluate the model under distinct in-sample and out-of-sample periods, as was necessary in earlier phases.

3. For portfolio-specific risk preferences, we assign a ”high-growth” portfolio with an ”Aggressive” risk tolerance level, and a ”low-risk” portfolio with a ”Low” risk tolerance level.

4. For specific investment instructions, we input only the task-oriented directives as previously outlined (i.e., focus on high growth or low risk within the S&P 500 universe).

```
Please provide your investment details:
1. Initial investment amount (e.g., 10000): 100000
2. Investment time horizon (e.g., '5 years', '10-15 years', 'long-term'): 2 years
3. Risk tolerance (e.g., 'low', 'medium', 'high', 'conservative', 'aggressive'): aggressive
4. Any specific preferences? (e.g., 'focus on tech', 'avoid fossil fuels', 'include GOOGL', or leave blank): Create a high-growth portfolio from S&P 500 companies t
hat aims to outperform the index in returns. Select around 20 tickers focusing on aggressive and growth potential.
```

Figure 6.1: Custom Investment Details for First High-Growth Portfolio

Table 6.1: Performance Comparison - High-Growth Portfolio (Multi-agents Vs. Prompt Engineering)

Approach	Annualized Return	Annualized Volatility	Sharpe Ratio	Maximum Drawdown	VaR (95%)	VaR (99%)	Beta	Alpha
Multi-Agent Framework	66.79%	33.98%	1.97	-29.86%	-6.70%	-9.62%	1.32	0.345
LLaMA 3.1 405B	59.21%	31.20%	1.866	4.70%	-2.95%	-4.65%	1.0405	0.3313
Claude 3.5 Sonnet	32.16%	25.38%	1.2278	21.36%	-2.90%	-4.21%	1.372	-0.6673
S&P 500 Index	15.40%	21.25%	0.68	33.92%	-1.96%	-3.74%	1	-

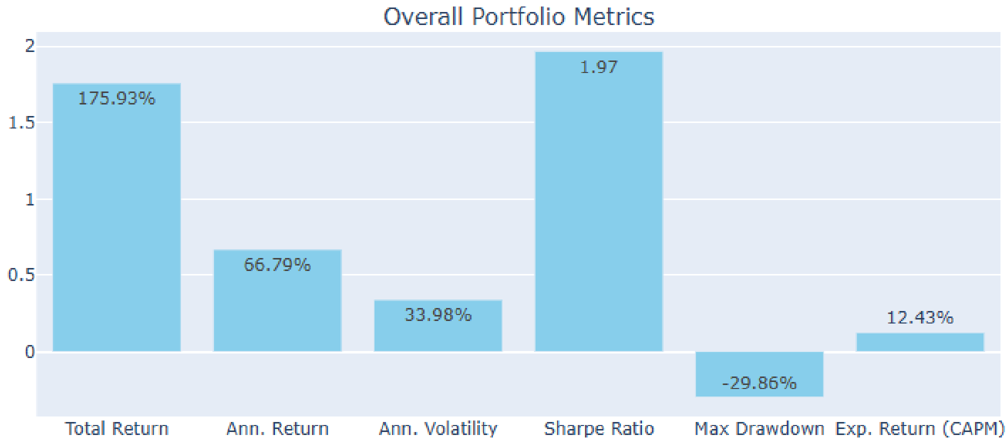
Table 6.2: Performance Comparison - Low-Risk Portfolio (Multi-agents Vs. Prompt Engineering)

Approach	Annualized Return	Annualized Volatility	Sharpe Ratio	Maximum Drawdown	VaR (95%)	VaR (99%)	Beta	Alpha
Multi-Agent Framework	19.59%	12.04%	1.63	-8.69%	-3.89%	-6.07%	0.65	0.092
Claude 3.5 Sonnet	20.66%	18.95%	1.0376	44.66%	-1.59%	-3.61%	0.8279	0.0772
LLaMA 3.1 405B	12.97%	18.06%	0.6628	3.08%	-1.61%	-3.02%	0.7261	0.015
S&P 500 Index	15.40%	21.25%	0.68	33.92%	-1.96%	-3.74%	1	-

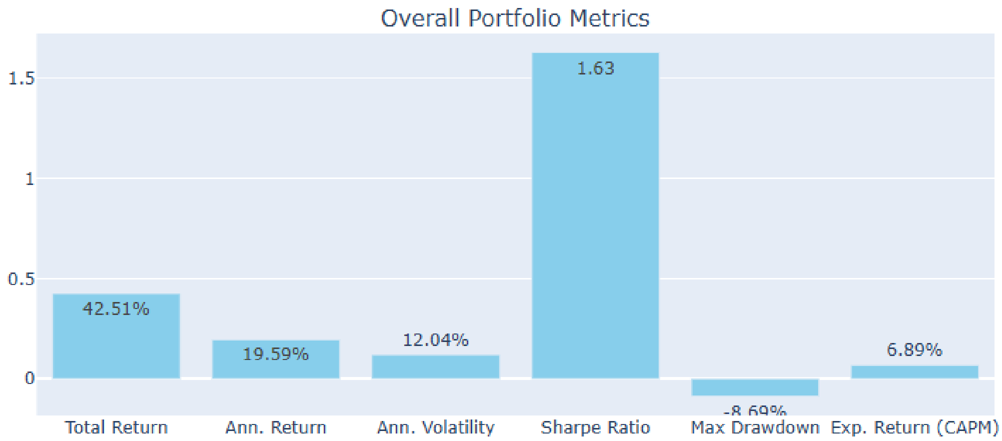
Following the comparative analysis presented in Table(s) 6.1 and 6.2, it is evident that the multi-agent framework achieved superior performance relative to the vanilla LLM-based approaches evaluated in the first phase of this research. In the high-growth portfolio case, the multi-agent system demonstrated a substantially higher annualized return (66.79%) and Sharpe ratio (1.97), outperforming both LLaMA 3.1 405B and Claude 3.5 Sonnet from our first phase and prompt engineering technique, while maintaining competitive volatility levels. For the low-risk portfolio, the multi-agent framework achieved a more favorable balance between return (19.59%) and volatility (12.04%), resulting in a higher Sharpe ratio (1.63) compared to all baseline models (also see Figure(s) 6.2 and 6.2a. These results indicate that the introduction of structured multi-agent workflows significantly enhanced the models' ability to manage diversification, optimize risk-return trade-offs, and align portfolio construction with user-specific objectives.

One can see the full metrics detail for both the high-growth and low-risk portfolios generated by the multi-agent framework with the instructions from the first phase (Chapter 3) below:

The asset-level comparison between the portfolios generated by the multi-agent framework and the prompt engineering approach (first phase of the research) highlights the progression in portfolio quality across the research phases (See Table(s) 6.3 and 6.4). While capturing a set of high-profile stocks, the first-phase portfolio lacked consistent diversification and risk controls, evidenced by extreme volatility levels and weaker Sharpe ratios across several assets. In contrast, the multi-agent framework demonstrated a more deliberate allocation strategy, with improved balance between risk and return. Assets in the agentic portfolio exhibited stronger risk-adjusted performance, more consistent momentum signals, and better alignment with user-specified investment goals. Notably, the multi-agent system leveraged validation and



(a) Portfolio Metrics for High-Growth Portfolio (Multi-Agent)



(b) Portfolio Metrics for Low-Risk Portfolio (Multi-Agent)

Figure 6.2: Metrics For Both Overall Portfolios Generated From Multi-Agents Framework (With Prompt from Phase 1)

modular reasoning to avoid over-concentration in ultra-volatile equities and instead allocated across sectors, resulting in a more resilient and explainable portfolio. This comparison confirms the added value of agentic coordination and multi-step reasoning in financial decision-making workflows.

Table 6.3: Asset-Level Metrics for High-Growth Portfolio (Multi-Agent Framework)

Asset	Weight	Ann. Return	Volatility	Sharpe	Max Drawdown	Exp. Return (CAPM)	Beta	SMA 50	SMA 200
NVDA	18%	109.32%	54.92%	1.99	-36.88%	14.01%	2.11	111.11	125.00
TSLA	15%	37.84%	62.72%	0.60	-53.77%	15.45%	2.43	261.44	293.58
AMZN	12%	37.81%	31.74%	1.19	-30.88%	10.39%	1.31	190.16	199.28
META	14%	66.17%	36.52%	1.81	-34.15%	10.07%	1.24	575.44	582.70
NFLX	10%	82.03%	34.68%	2.37	-27.51%	11.63%	1.58	990.43	855.32
AVGO	10%	90.34%	51.66%	1.75	-41.15%	9.27%	1.06	184.44	185.57
AMD	8%	6.70%	50.49%	0.13	-63.00%	13.36%	1.97	98.30	127.54
SHOP	7%	31.10%	56.74%	0.55	-42.39%	16.51%	2.67	94.50	93.52
DOCU	3%	34.16%	43.80%	0.78	-34.86%	9.94%	1.21	80.68	76.52
CRM	3%	19.99%	33.49%	0.60	-35.60%	10.89%	1.42	269.38	293.07

Table 6.4: Asset-Level Metrics for Low-Risk Portfolio (Multi-Agent Framework)

Asset	Weight	Ann. Return	Volatility	Sharpe	Max Drawdown	Exp. Return (CAPM)	Beta	SMA 50	SMA 200
AEP	7.27%	10.29%	20.04%	0.51	-18.19%	6.46%	0.43	104.80	98.37
AMGN	6.36%	12.07%	25.08%	0.48	-22.74%	6.74%	0.50	297.06	298.99
CL	5.45%	7.36%	17.51%	0.42	-20.39%	6.26%	0.39	91.84	94.16
COST	9.09%	45.28%	20.17%	2.24	-17.29%	8.95%	0.99	962.25	933.80
DUK	7.27%	13.83%	17.69%	0.78	-11.59%	6.19%	0.38	119.41	113.41
JNJ	5.45%	1.39%	17.27%	0.08	-15.95%	6.35%	0.41	158.59	155.58
KMB	4.55%	1.09%	17.88%	0.06	-17.95%	6.17%	0.37	137.99	135.52
KO	6.36%	7.58%	15.07%	0.50	-16.26%	6.63%	0.47	70.88	67.08
LLY	6.36%	33.05%	34.33%	0.96	-24.65%	6.62%	0.47	816.59	838.56
MDT	4.55%	1.33%	20.24%	0.07	-22.54%	8.23%	0.83	86.97	85.86
PG	5.45%	4.14%	16.74%	0.25	-11.76%	6.35%	0.41	165.58	166.76
SO	7.27%	13.87%	18.29%	0.76	-13.31%	6.20%	0.38	90.27	86.85
T	8.18%	34.42%	23.87%	1.44	-17.83%	7.25%	0.61	27.01	23.03
VZ	7.27%	15.97%	23.03%	0.69	-14.93%	6.22%	0.38	43.35	40.98
WMT	9.09%	40.40%	20.74%	1.95	-21.93%	7.65%	0.70	90.89	86.51

The comparison in Table 6.5 highlights that while prompt engineering provided a low-complexity starting point for portfolio generation, the multi-agent architecture introduced significant advantages in optimization quality, diversification control, and system robustness. The following commentary boxes (See Figure(s) 6.3 and 6.4) summarize the reasoning and performance analysis generated by the commentary agent in the multi-agent framework for two distinct portfolios: high-growth and low-risk. Each box highlights key metrics such as historical

Aspect	Multi-Agent Framework	Prompt Engineering (Vanilla LLMs)
Asset Weight Optimization	Handled via agent validation	Poor (manual normalization required)
Diversification Quality	Structured across asset classes	Moderate, inconsistent
Flexibility	High (modular agents, dynamic task allocation)	Low (static prompting only)
Scalability	High (can expand with more agents or tools)	Very Low (requires new prompt design manually)
Error Handling	Dedicated validation agents	No error recovery (only re-prompting)
Explainability	High (transparent agent roles and reports)	Low (LLM reasoning often implicit)
Real-Time Adaptability	Enabled (live browsing, dynamic retrieval)	Not possible (static, trained data only)
System Complexity	Higher (multi-step orchestration)	Very Low (simple single prompt)
Setup Difficulty	Higher (needs system design and orchestration)	Very Easy (prompt crafting)

Table 6.5: Qualitative Comparison Between Multi-Agent Framework and Prompt Engineering Approach (Vanilla LLMs)

returns, volatility, CAPM-based expectations, and momentum trends. The insights reflect how the system tailored each portfolio to match user-defined risk tolerance and investment horizons. Since the output is passed to the LLM-agent as JSON, the performance metrics are often represented as float values rather than percentages. This is standard practice in financial analysis, where metrics such as returns, volatility, and Sharpe ratios are commonly handled in decimal form for consistency and precision. Occasionally, the commentary may reflect these raw values directly, depending on how the model interprets the data.

As mentioned earlier in Chapter 5, the LLM-based agent responsible for generating commentary and reasoning intelligently selects the most relevant highlights ("*hotspots*") from the portfolio to emphasize. For instance, if the client profile targets high-growth and aggressive strategies, the agent prioritizes discussing top-performing assets based on Sharpe ratio and return. Conversely, for low-risk or balanced portfolios, it focuses on assets with lower volatility and stable returns that align with the client's objectives.

LLM Commentary & Reasoning – High-Growth Portfolio

The proposed portfolio allocation is designed to align with your goal of creating a high-growth portfolio from S&P 500 companies, aiming to outperform the index in returns. Given your aggressive risk tolerance and a two-year time horizon, the allocation focuses on high-growth potential assets such as NVDA, TSLA, AMZN, and META, which are known for their strong historical performance and growth prospects.

Key Metrics:

- **Historical Performance:** The portfolio includes assets like NVDA and NFLX, which have shown exceptional total returns of 3.33 and 2.28 respectively, with high annualized returns and Sharpe ratios indicating strong risk-adjusted performance.
- **Volatility:** The portfolio's assets exhibit high annualized volatility, consistent with an aggressive risk profile. For instance, NVDA and AMD have volatilities of 0.5492 and 0.5049, respectively.
- **CAPM Expected Return:** The expected returns based on the CAPM model are promising, with NVDA at 0.1401 and AMD at 0.1336, suggesting potential for high returns relative to their risk.
- **Momentum Outlook:** The momentum indicators, such as the SMA trends, show mixed signals. For example, NFLX's SMA-50 is above its SMA-200, indicating positive momentum, whereas AMD's SMA-50 is below its SMA-200, suggesting caution.

The portfolio's validation status is **PASS**, indicating that the allocation meets the necessary criteria for your investment goals and risk tolerance. No specific issues were identified during validation.

Summary: This portfolio is structured to capitalize on aggressive growth opportunities within the S&P 500, leveraging both historical performance and expected future returns. However, the inherent volatility and mixed momentum signals should be considered in the context of your risk tolerance and investment horizon.

Figure 6.3: LLM Commentary & Reasoning – High-Growth Portfolio

LLM Commentary & Reasoning – Low-Risk Portfolio

The proposed portfolio allocation is designed to align with your goal of creating a low-risk portfolio from S&P 500 companies that aims to outperform the index with minimal volatility. This allocation includes 15 stable, defensive tickers, each carefully selected to match your low-risk tolerance and two-year time horizon.

Portfolio Composition and Metrics:

- The portfolio includes a diverse mix of companies such as AEP, AMGN, CL, COST, DUK, JNJ, KMB, KO, LLY, MDT, PG, SO, T, VZ, and WMT. These companies are known for their stability and defensive characteristics, which are crucial for minimizing volatility.
- **Historical Performance:** The portfolio's historical performance shows a balanced approach with a focus on companies that have demonstrated resilience. For instance, COST has an impressive annualized return of 45.28% with a Sharpe ratio of 2.24, indicating strong risk-adjusted returns.
- **Volatility and Risk:** The portfolio's overall volatility is kept low, with individual stocks like CL and DUK showing annualized volatilities of 17.51% and 17.69%, respectively. This aligns with your low-risk preference.
- **CAPM Expected Return:** The expected returns based on the CAPM model are modest, with most stocks showing expected returns between 6% and 8%. This suggests a conservative growth outlook, consistent with your risk tolerance.
- **Momentum Outlook:** The momentum indicators, such as the 50-day and 200-day SMAs, show a positive trend for several stocks like COST and KO, indicating potential upward momentum.

Validation Outcome: The portfolio has passed the validation checks, confirming that the allocation is sound and aligns with your investment goals. No issues were found during the validation process.

Market Context: The market news highlights the stability and growth potential of companies like AMGN, which has a strong presence in various markets and a positive earnings outlook. This further supports the defensive nature of the portfolio.

Figure 6.4: LLM Commentary & Reasoning – Low-Risk Portfolio

Multi-Agent System Vs. RAG-Enhanced LLM (Phase 2)

In this section, we compare the efficiency of the proposed multi-agent framework with the results obtained in the second phase of our research (see Chapter 4). We used the same prompt that we used for RAG-Enhanced LLMs (see Figure 4.1) to generate a portfolio from our multi-agent framework. In the instructions we guided the framework to create a high-risk and high-reward portfolio within a 1-year timeframe to ensure fairness.

We present a detailed performance comparison between the multi-agent system and the RAG-enhanced LLM approach, highlighting the improvements achieved in portfolio construction, risk management, and overall decision quality through the adoption of an agentic workflow.

Framework	Total Return	Annualized Return	Annualized Volatility	Sharpe Ratio	Max Drawdown	Beta	Alpha
Multi-Agent Framework	57.42%	58.00%	20.22%	2.87	14.42%	1.021	0.297
LLaMA 3.1 + RAG	39.94%	40.31%	12.86%	3.06	10.57%	0.944	0.186
LLaMA 3.1 (Normal)	42.24%	42.64%	16.29%	2.56	14.00%	1.001	0.197
S&P 500 Index	24.54%	24.76%	12.67%	1.88	10.14%	1.004	0.017

Table 6.6: Comparison between Multi-Agent Framework, RAG-Enhanced LLaMA 3.1, Normal LLaMA 3.1, and S&P 500 (2024 Performance)

Based on the performance results presented in Table 6.6, the multi-agent framework demonstrated substantial advantages in overall portfolio construction and risk-adjusted returns compared to both the RAG-enhanced and normal LLaMA 3.1 models. While the RAG-enhanced framework achieved the highest Sharpe ratio (3.06), indicating superior risk-adjusted performance for its time period, the multi-agent system exhibited a significantly higher total return (57.42%) and annualized return (58.00%), albeit with moderately higher volatility (20.22%). The multi-agent system maintained a competitive Sharpe ratio of 2.87 while offering better portfolio optimization control and resilience against market fluctuations, as evidenced by acceptable maximum drawdowns. In contrast, the RAG-based framework depended heavily on retrieval quality and lacked multi-stage validation, making it less flexible for dynamic portfolio management. Overall, the multi-agent architecture provided a more scalable, modular, and high-performing solution, capable of sustaining returns in real-time conditions while maintaining robust risk management characteristics.

To ensure a fair and consistent evaluation, we constrained the multi-agent framework to perform asset allocation strictly within the 2024 timeframe. This restriction allows for a direct and equitable comparison between the agentic approach and the RAG-enhanced LLaMA framework developed in the earlier phase of this research. Although the multi-agent system inherently possesses real-time capabilities, including web browsing and dynamic data retrieval, these features were intentionally limited for this evaluation. The goal was to isolate performance differences based solely on system architecture and reasoning processes, rather than on access to more recent or broader data. Nonetheless, it is important to note that the full deployment of the multi-agent framework can leverage real-time market conditions to curate portfolios with even greater responsiveness and contextual relevance.

Aspect	**Multi-Agent Framework**	**RAG-Enhanced LLaMA 3.1**
Portfolio Return Potential	High (57.4% Total Return)	Moderate (39.9% Total Return)
Sharpe Ratio (Risk-Adjusted Return)	High (2.87)	Very High (3.06)
Annualized Volatility	Moderate (20.2%)	Low (12.9%)
Flexibility	High (Modular agents)	Medium (Depends on retrieval quality)
Scalability	High (Easily extendable with agents)	Moderate (Requires retriever rebuilding)
Error Handling	Modular Validation Agents	Limited to output quality
Real-Time Adaptability	High (API integration, dynamic prompts)	Moderate (static retriever corpora)
System Complexity	Higher (multiple LLM agents)	Lower (simpler architecture)
Setup Difficulty	Higher (needs orchestration)	Lower (plug-and-play retrieval)
Explainability	High (structured agent roles)	Lower (flat output reasoning)

Table 6.7: Qualitative Comparison Between Multi-Agent Framework and RAG-Enhanced LLaMA 3.1

The comparison in Table 6.7 illustrates that while the RAG-enhanced LLaMA framework achieved strong risk-adjusted returns through external knowledge retrieval, it remained limited by static retrieval quality and lacked modular reasoning capabilities. In contrast, the multi-agent framework offered higher flexibility, dynamic task handling, and greater scalability, making it more adaptable for real-time financial decision-making. These architectural improvements contributed to the multi-agent system's overall superior performance and robustness.

Table 6.8: Asset-Level Metrics and Weights – Multi-Agent Framework Portfolio (Multi-Agents Framework vs. RAG-Enahnced LLM Comparison)

Asset	**Weight**	**Ann. Return**	**Volatility**	**Sharpe**	**Max Drawdown**	**Exp. Return (CAPM)**	**Beta**	**SMA 50**	**SMA 200**
AAPL	12%	36.86%	22.45%	1.64	-15.35%	10.24%	1.27	236.65	212.16
AMZN	12%	48.06%	28.15%	1.71	-19.49%	10.39%	1.31	209.24	189.51
NVDA	15%	187.93%	52.54%	3.58	-27.05%	14.01%	2.11	139.95	117.61
TSLA	10%	68.72%	63.65%	1.08	-40.92%	15.45%	2.43	342.52	237.64
GOOGL	10%	39.28%	28.11%	1.40	-22.14%	9.05%	1.01	177.06	168.69
SPY	8%	26.28%	12.59%	2.09	-8.41%	8.96%	0.99	589.27	548.28
QQQ	8%	29.10%	17.96%	1.62	-13.56%	10.54%	1.34	508.64	472.41
GLD	5%	26.40%	15.06%	1.75	-8.12%	5.88%	0.31	246.02	228.91
XLF	5%	30.14%	14.21%	2.12	-7.23%	7.69%	0.71	48.60	43.73
ARKK	5%	14.20%	35.77%	0.40	-23.57%	14.01%	2.11	55.34	47.79
JPM	5%	42.80%	23.57%	1.82	-10.13%	9.31%	1.07	234.81	208.40
BND	5%	2.00%	5.42%	0.37	-3.97%	4.78%	0.06	71.55	70.82

Figure 6.5 is the reasoning and the comments for the generated portfolio by the multi-agent framework by the commentary agent in the workflow.

The asset-level metrics and weights presented in Table 6.8 demonstrate strong diversification across asset types, sectors, and risk profiles, highlighting the system's ability to interpret and execute instructions beyond static and augmented methods. The allocation spans high-growth equities such as NVIDIA (NVDA), Tesla (TSLA), and Amazon (AMZN); core index exchange-traded funds (ETFs) like the SPDR S&P 500 ETF (SPY) and the Invesco QQQ Trust (QQQ); sector-specific exposure through the Financial Select Sector SPDR Fund (XLF); and defensive assets including the SPDR Gold Shares (GLD) and the Vanguard Total Bond Market ETF (BND). This blend of technology, financials, commodities, and fixed income instruments ensures that the portfolio is not overly concentrated in a single domain, thereby managing risk more effectively. Moreover, the inclusion of both high-beta and low-beta assets reflects a deliberate balancing strategy, where potential returns are optimized without sacrificing resilience to market volatility.

LLM Commentary & Reasoning – Agentic Portfolio (Agent vs. RAG Comparison)

The proposed portfolio allocation is designed to align with your aggressive risk tolerance and goal of creating a high-reward portfolio. It includes a diverse mix of ETFs, stocks, bonds, and commodities, focusing on high-risk sectors and asset classes. Key allocations include significant weights in high-growth tech stocks like NVDA (15%), AAPL (12%), and AMZN (12%), which have shown strong historical performance with annualized returns of 1.8793, 0.3686, and 0.4806, respectively. These stocks also exhibit high volatility, consistent with your risk appetite.

The portfolio's expected return, as estimated by the CAPM, is supported by high beta values, indicating a strong correlation with market movements. For instance, NVDA and TSLA have high betas of 2.11 and 2.43, respectively, suggesting potential for high returns in bullish markets. The CAPM expected returns for these assets are 0.1401 and 0.1545, reflecting their risk-adjusted potential.

Momentum indicators, such as the SMA trends, show positive outlooks for most assets, with current prices generally above their 50-day and 200-day SMAs, indicating upward momentum. For example, NVDA's SMA-50 is 139.95 compared to its SMA-200 of 117.61, suggesting a strong upward trend.

The portfolio's overall metrics include a balanced approach to risk and reward, with a focus on maximizing returns while managing volatility. The Sharpe ratios for key assets like SPY (2.09) and GLD (1.75) indicate efficient risk-adjusted returns. The maximum drawdown figures, such as TSLA's -0.4092, highlight the potential risks involved, which are consistent with an aggressive strategy.

Validation of the portfolio has passed, indicating that the allocation is sound and meets the specified criteria without any issues. This suggests that the portfolio is well-structured to achieve your investment goals within the given time horizon.

Figure 6.5: LLM Commentary & Reasoning – Multi-Agent Portfolio (Agents vs. RAG)

Multi-Agent System Vs Real-world Portfolio Managers (Phase 3)

In this section, we evaluate the efficiency and performance of the multi-agent framework in constructing portfolios that resemble those created by real-world portfolio managers. Specifically, we examine how the agent-generated portfolio compares to portfolios offered by Wealthsimple, one of Canada's largest online investment management platforms, with over $50 billion in assets under management [1].

To evaluate the real-world relevance of our agent-generated portfolios, we compare them against three model portfolios offered by Wealthsimple. Specifically, we examine the asset selection across three distinct risk profiles: Aggressive, Balanced, and Conservative.

1. **Aggressive Portfolio**: Wealthsimple's aggressive portfolio includes Canadian equities, emerging market equities, global equities, international equities, and U.S. equities. This corresponds to our agent prompt: "Focus on high return and high risk", targeting long-term investors with high risk tolerance.

2. **Conservative Portfolio**: This portfolio adds fixed income components such as Canadian short-term bonds, government bonds, Canadian aggregate bonds, and gold, while still maintaining exposure to equity markets. The corresponding prompt used in our system was: "This portfolio is suited for short-term goals (e.g., within 3 years) or retirees. It aims to reduce downside risk, though potential growth may be limited."

3. **Balanced Portfolio**: The balanced portfolio includes the same asset classes as the conservative one, but aims for a more even distribution between growth and preservation. Our prompt reflected this goal: "This portfolio suits medium-term goals or those nearing retirement. It balances caution with moderate growth potential."

By comparing the asset selection and allocation outcomes of these portfolios with those generated by our multi-agent framework, we assess the system's ability to align with established, professionally curated investment strategies.

[1] https://www.wealthsimple.com/en-ca

While Wealthsimple does not publicly disclose the exact underlying assets or individual security allocations within each of their model portfolios on their website[2], they do provide a breakdown of the asset classes included in each portfolio type. These typically include categories such as Canadian equities, U.S. equities, international equities, emerging market equities, fixed income instruments, and commodities like gold. Based on this high-level classification, we guided our multi-agent framework using tailored natural language prompts that reflect the same asset class composition and investment philosophy as those described by Wealthsimple. This approach enabled a fair and meaningful comparison between the agent-generated portfolios and the real-world strategies employed by a professional investment management platform, despite the lack of access to exact asset-level disclosures. It is important to note that, according to the disclaimer provided by Wealthsimple[3], the performance data associated with their portfolios is hypothetical and may not reflect actual returns. Nonetheless, these figures serve as a useful benchmark for evaluating the effectiveness of our multi-agent framework, particularly in demonstrating its ability to generate portfolios aligned with diverse instructional inputs and investor profiles[4].

- **Aggressive Portfolio (1 year period):**

  The comparison highlights how the agent-generated portfolio performs under similar asset class assumptions and risk profiles (See Table 6.9). The second Table (6.10) outlines the key financial metrics of the multi-agent portfolio, offering insight into its return characteristics, risk exposure, and momentum outlook.

[2] `https://www.wealthsimple.com/en-ca/managed-investing/classic-portfolio`

[3] `https://www.wealthsimple.com/en-ca/legal/disclaimers/portfolio-performance-projected-chart`

[4] This thesis uses publicly available data from Wealthsimple strictly for academic, non-commercial purposes, with proper citation and acknowledgment of their disclaimers.

Table 6.9: Performance Comparison Between Wealthsimple's Aggressive Portfolio and Multi-Agent Framework

Portfolio Type	Model	1-Year Total Return	1-Year Annualized Return
Aggressive	Wealthsimple (Historical)	12.8%	12.8%
Aggressive	Multi-Agent Framework (Historical)	43.62%	44.04%
Aggressive	Wealthsimple (Projected)	8.2%	8.2%
Aggressive	Multi-Agent Framework (Projected)	10.46%	10.46%

Table 6.10: Performance Metrics of Multi-Agent Framework (Aggressive Portfolio)

Metric	Value
Total Return	43.62%
Annualized Return	44.04%
Annualized Volatility	27.88%
Sharpe Ratio	1.58
Max Drawdown	-24.54%
CAPM Expected Return	10.46%
Momentum Outlook (SMA)	Bullish

The results indicate that the multi-agent framework performs effectively when instructed to construct an aggressive portfolio, successfully selecting high-growth assets consistent with the specified risk tolerance and return objectives. This performance is accompanied by a higher volatility level, which is expected given the risk-oriented prompt used to guide asset selection. The Sharpe ratio of 1.58 reflects strong risk-adjusted returns, suggesting that the agent-based system is not only capable of maximizing returns but also managing risk effectively through diversified allocations and dynamic data integration. The bullish momentum signal and strong CAPM-adjusted return further reinforce the framework's potential in constructing competitive, data-driven portfolios aligned with investor goals.

The aggressive portfolio generated by the multi-agent framework demonstrates a well-

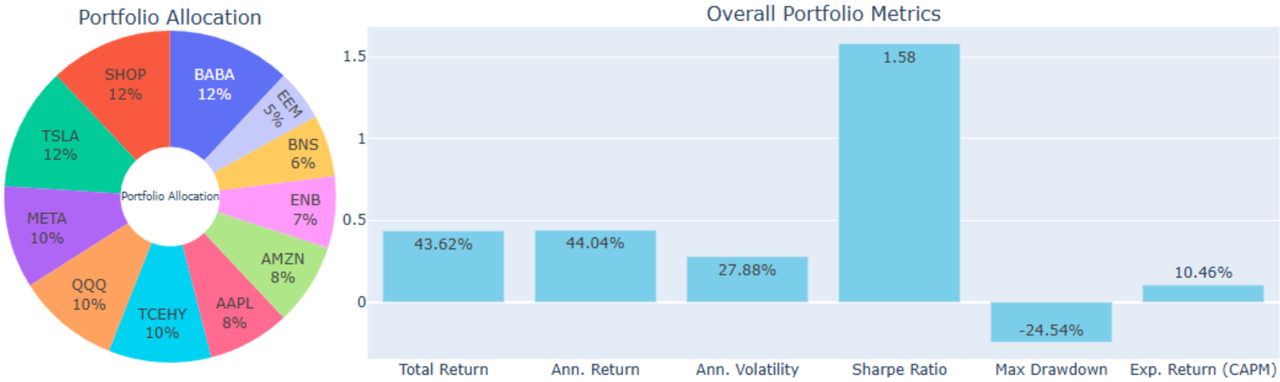


Figure 6.6: Aggressive Portfolio Allocation and Performance for Multi-Agent Framework

structured diversification strategy across both geographies and asset types (See Figure 6.6). It includes a balanced mix of high-growth U.S. equities such as Apple (AAPL), Amazon (AMZN), Meta Platforms (META), and Tesla (TSLA), alongside international exposure through Alibaba (BABA) (China)), TCEHY (Tencent Holdings), and EEM (emerging markets ETF). Additionally, the inclusion of Canadian equities like SHOP (Shopify, a Canadian company and ENB (Enbridge Inc.), as well as BNS (Nova Scotia Bank), ensures regional diversification. Notably, QQQ, a technology-focused U.S. ETF, further enhances exposure to innovative sectors. This allocation aligns well with Wealthsimple's aggressive portfolio structure, which emphasizes Canadian, U.S., international, global, and emerging market equities. However, the agent-based approach adds nuance through specific stock selection rather than relying solely on broad index funds, potentially allowing for greater alpha generation while still maintaining exposure across key asset classes. You can also see the commentary and reasoning compiled by the LLM agent in the framework in Figure 6.7.

LLM Agent Commentary & Reasoning – Aggressive Portfolio

The proposed portfolio allocation is designed to align with your aggressive risk tolerance and goal of achieving high returns within a one-year time horizon. The portfolio is diversified across various high-risk, high-return equities, with a focus on Canadian, Emerging Market, Global, International, and U.S. equities, as per your preferences.

Key Metrics:

- **Historical Performance:** The portfolio includes assets like SHOP (Shopify) and BABA (Alibaba), which have shown significant historical returns of 77.77% and 44.3%, respectively. These assets contribute to the high return potential of the portfolio.
- **Volatility and Risk:** The portfolio's assets exhibit high annualized volatility, with SHOP at 60.53% and BABA at 45.89%, reflecting the aggressive risk profile.
- **Sharpe Ratio:** The Sharpe ratios for assets like SHOP (1.3) and ENB (1.65) indicate a favorable risk-adjusted return, suggesting that these assets have historically provided good returns for the level of risk taken.
- **CAPM Expected Return:** The CAPM model suggests expected returns for high-beta stocks like SHOP (16.51%) and AMZN (10.39%), indicating potential for high returns in line with market movements.
- **Momentum Outlook:** The momentum indicators, such as the SMA trends, show that several assets like BABA and SHOP are trading above their 200-day moving averages, suggesting positive momentum.

The portfolio has passed validation, indicating that the allocation is sound and meets the specified criteria. However, it's important to note that while historical performance and CAPM expected returns provide insights, they do not guarantee future results.

In summary, this portfolio is structured to capitalize on high-risk, high-return opportunities, consistent with your aggressive investment strategy. The inclusion of diverse equities across different markets aims to enhance return potential while managing risk through diversification.

Figure 6.7: LLM Agent Commentary & Reasoning for Aggressive Portfolio

- **Conservative Portfolio:**

  To further evaluate the practical relevance of our multi-agent system for portfolio construction, we compare its output to a conservative investment strategy.

Table 6.11: Performance Comparison Between Wealthsimple's Conservative Portfolio and Multi-Agent Framework

Portfolio Type	Model	3-Year Total Return	3-Year Annualized Return
Conservative	Wealthsimple (Historical)	19.0%	6.0%
Conservative	Wealthsimple (Projected)	17.8%	–
Conservative	Multi-Agent Framework (Historical)	30.77%	9.63%
Conservative	Multi-Agent Framework (Projected)	6.34%	6.34%

Note: The projected calculation for the Multi-Agent Framework in Table 6.11 is only for one year since the CAPM formula implemented in the agents is only for a one-year time window. The CAPM was only implemented for research purposes here; the prediction at the industry level requires more sophisticated algorithms, which were out of the scope of this research.

Table 6.12: Performance Metrics of Multi-Agent Framework (Conservative Portfolio)

Metric	Value
Total Return (3Y)	30.77%
Annualized Return	9.63%
Annualized Volatility	8.82%
Sharpe Ratio	1.09
Max Drawdown	-10.13%
CAPM Expected Return (1Y)	5.69%
Momentum Outlook (SMA)	Neutral to Slightly Bullish

As shown, the conservative portfolio produced by our multi-agent system achieved a total return of 30.77% and an annualized return of 9.63%. Moreover, our framework demonstrated strong risk-adjusted performance with a Sharpe ratio of 1.09 and maintained a moderate maximum drawdown of -10.13%. These results suggest that a well-instructed LLM-based multi-agent system can not only replicate but also potentially exceed traditional portfolio construction methods in conservative investment settings (See Table 6.12).

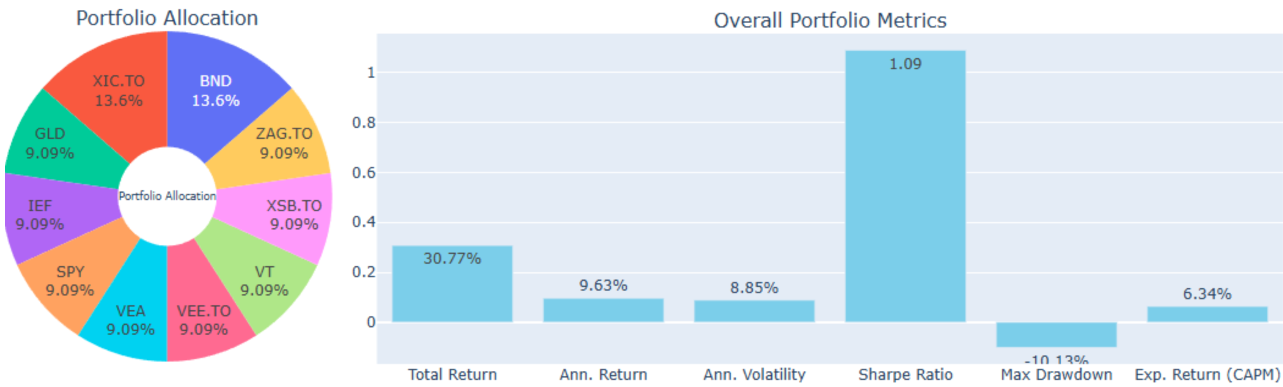


Figure 6.8: Conservative Portfolio Allocation and Performance For Multi-Agent Framework

The conservative portfolio generated by the multi-agent framework demonstrates strong diversification across asset classes and geographic regions, aligning with the objectives of a low-risk and conservative investment strategy (See Figure 6.8). It includes a significant allocation to fixed income instruments, such as BND (Vanguard Total Bond Market ETF), XSB.TO (iShares Core Canadian Short-Term Bond Index ETF), ZAG.TO (BMO Aggregate Bond Index ETF), and IEF (iShares 7-10 Year Treasury Bond ETF), all of which provide portfolio stability and consistent income. Equity exposure is maintained through SPY (SPDR S&P 500 ETF Trust), VEA (Vanguard FTSE Developed Markets ETF), VEE.TO (Vanguard FTSE Emerging Markets All Cap Index ETF), XIC.TO (iShares Core S&P/TSX Capped Composite Index ETF) and VT (Vanguard Total World Stock ETF), ensuring broad global diversification. The inclusion of GLD (SPDR Gold Shares) adds a non-correlated asset class, offering further downside protection during periods of market stress. This allocation closely resembles Wealthsimple's conservative portfolio structure, which emphasizes international equities, fixed income, and gold. Overall, the agent-based system effectively generated a portfolio that reflects diversification principles, suitable for short-term goals or retirement-focused investors. Moreover, we provided commentary and reasoning provided by the LLM agent at the last step of the workflow below on Figure 6.9.

LLM Agent Commentary & Reasoning

The proposed portfolio allocation is designed to align with your conservative risk tolerance and short-term investment horizon of 3 years, focusing on a diverse range of asset classes including Canadian equities, emerging market equities, global equities, international equities, U.S. equities, fixed income, and gold. This diversification aims to balance risk and return, which is crucial for retirees or those with short-term goals.

Key metrics for the portfolio include a total return of 30.77% and an annualized return of 9.63%, with an annualized volatility of 8.85%. The Sharpe ratio of 1.09 indicates a favorable risk-adjusted return, suggesting that the portfolio is expected to perform well relative to its risk. The maximum drawdown of -10.13% reflects the largest peak-to-trough decline, which is moderate and aligns with a conservative risk profile.

The CAPM expected return for the portfolio is 6.34%, which is consistent with the conservative nature of the allocation, providing a reasonable expectation of returns given the market risk. The portfolio's momentum outlook is bullish, as indicated by the 50-day SMA being greater than the 200-day SMA, suggesting a positive trend in asset prices.

Figure 6.9: LLM Agent Commentary & Reasoning for Conservative Portfolio

- **Balanced Portfolio:**

  For the balanced portfolio, we instructed the multi-agent system to construct an allocation that aligns with Wealthsimple's balanced investment approach. Specifically, the asset classes were modeled to reflect those used by Wealthsimple, including Canadian equities, emerging market equities, global equities, international equities, U.S. equities, fixed income, Canadian short-term bonds, government bonds, Canadian aggregate bonds, and gold. To guide the agentic system effectively, we used the following instruction: *"This portfolio suits medium-term goals or those nearing retirement. It balances caution with moderate growth potential."* These instructions helped ensure that the resulting portfolio emphasized both capital preservation and modest growth, consistent with a medium-risk tolerance level. The system then dynamically selected appropriate assets within each class and computed a risk- and return-balanced allocation to fulfill the specified investment objectives.

Table 6.13: Performance Comparison – Wealthsimple vs. Multi-Agent (Balanced Portfolio)

Portfolio Type	**Model**	**1-Year Total Return**	**1-Year Annualized Return**
Balanced	Wealthsimple (Historical)	12.4%	12.4%
Balanced	Wealthsimple (Projected)	6.8%	6.8%
Balanced	Multi-Agent Framework (Historical)	19.0%	19.68%
Balanced	Multi-Agent Framework (Projected)	7.75%	7.75%

Table 6.14: Performance Metrics of Multi-Agent Framework (Balanced Portfolio)

Metric	**Value**
Total Return (1Y)	19.00%
Annualized Return	19.68%
Annualized Volatility	10.29%
Sharpe Ratio	1.91
Max Drawdown	-8.10%
CAPM Expected Return	7.75%
Momentum Outlook (SMA)	Bullish

With a total return of 19.0% and an annualized return of 19.68%, the multi-agent generated portfolio demonstrated stronger short-term performance under the same asset class assumptions. Additionally, the projected return based on the CAPM model (7.75%) remains consistent with the risk profile of a balanced investor. This suggests that the multi-agent system is capable of matching real-world portfolio strategies in structure and generating competitive, and in this case, superior, in terms of both realized and expected returns. The results indicate a strong potential for using structured, instruction-guided LLM agents in designing portfolios tailored to moderate-risk investment goals (See Table(s) 6.14 6.13).

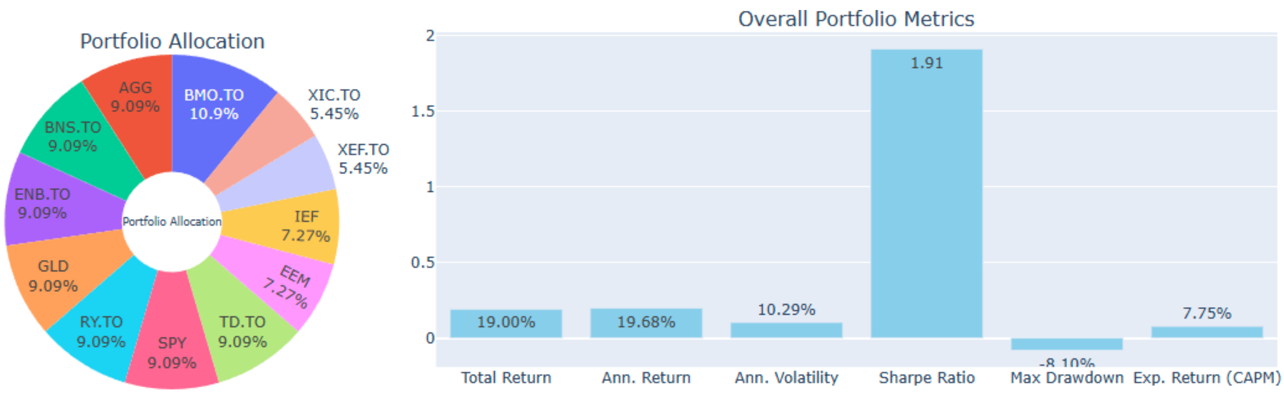


Figure 6.10: Balanced Portfolio Allocation and Performance for Multi-Agent Framework

The balanced portfolio constructed by the multi-agent framework reflects a diversified allocation across equities, fixed income, and alternative assets, well aligned with Wealthsimple's balanced asset class strategy (See Figure 6.10). It includes significant exposure to Canadian financial institutions such as BMO.TO (Bank of Montreal), RY.TO (Royal Bank of Canada), TD.TO (Toronto-Dominion Bank), and BNS.TO (Bank of Nova Scotia), providing both dividend income and regional diversification. Broader equity exposure is achieved through SPY (SPDR S&P 500 ETF), EEM (iShares MSCI Emerging Markets ETF), XEF.TO (iShares MSCI EAFE IMI Index ETF), and XIC.TO (iShares Core S&P/TSX Capped Composite Index ETF), ensuring representation across U.S., international, and Canadian markets. Fixed income allocations include AGG (iShares Core U.S. Aggregate Bond ETF) and IEF (iShares 7–10 Year Treasury Bond ETF), adding portfolio stability. The inclusion of ENB.TO (Enbridge Inc.) offers exposure to infrastructure and energy, while GLD (SPDR Gold Shares) serves as a hedge against market volatility. Overall, this portfolio reflects a prudent balance of growth and stability, tailored for investors with medium-term goals or those approaching retirement. Lastly, in Figure 6.11, you can see the commentary generated by the LLM agent for the balanced portfolio.

LLM Commentary & Reasoning – Balanced Portfolio

The proposed portfolio allocation is designed to align with your goal of medium-term growth while balancing caution and moderate growth potential. The portfolio includes a diverse mix of Canadian equities, emerging market equities, global equities, international equities, U.S. equities, fixed income, and gold, reflecting your specific preferences.

Key Metrics:

- **Historical Performance:** The portfolio's assets have shown varied historical returns, with `ENB.TO` and `GLD` exhibiting strong annualized returns of 35.02% and 35.52%, respectively, and high Sharpe ratios of 2.31 and 1.98, indicating efficient risk-adjusted returns. Conversely, `TLT` has underperformed with a negative annualized return of -2.32%.
- **Volatility and Risk:** The portfolio maintains a medium risk profile, with assets like `BMO.TO` and `SPY` showing higher volatility, while `AGG` and `IEF` offer lower volatility, contributing to overall stability.
- **CAPM Expected Return:** The expected returns based on the CAPM model suggest moderate growth potential, with most equities like `RY.TO` and `SPY` having expected returns around 9%, aligning with your medium risk tolerance.
- **Momentum Outlook:** The momentum indicators, such as the SMA trends, show positive signals for several assets. For instance, `BMO.TO` and `ENB.TO` are trading above their 50-day and 200-day SMAs, suggesting upward momentum.

The portfolio has passed validation, indicating that the allocation is consistent with your investment goals and risk tolerance. The diverse asset mix aims to capture growth opportunities while mitigating risks through fixed income and gold allocations.

Figure 6.11: LLM Commentary & Reasoning for Balanced Portfolio

The full list of assets, along with their performance metrics for all three multi-agent-generated portfolios discussed above, is provided in the appendix (see Appendix A.4).

6.1.2 Risk Management and Diversification Effectiveness

As demonstrated in the previous sections and supported by Table(s) 6.1, 6.2, and 6.6, and other Figure(s) throughout this Chapter, the multi-agent framework consistently excels at constructing diverse and goal-aligned portfolios. We evaluated its performance across a range of portfolio types, including balanced, aggressive, high-growth, low-risk, and conservative, each guided by different levels of instruction specificity. In some cases, the framework was simply prompted to prioritize high growth and high risk; in others, it was explicitly directed to use certain asset classes to match real-world benchmarks, such as Wealthsimple portfolios. In both scenarios, the system demonstrated a clear ability to interpret user instructions, analyze current market data, and autonomously coordinate the full portfolio construction process, including asset selection, financial metric calculation, validation, and natural-language commentary generation.

Importantly, validation steps ensure that each portfolio adheres to user-defined constraints and risk preferences. Across all portfolio types, the multi-agent framework maintained diversification and managed risk appropriately by selecting assets aligned with the specified strategy. Notably, even when instructions were minimal (e.g., simply requesting a "low-risk" portfolio), the framework was able to infer and implement an allocation strategy consistent with that objective (see Table 6.4). These results affirm the system's robustness and adaptability in generating high-quality portfolios under both structured and open-ended instructions.

Chapter 7

Conclusion

Nowadays, more companies are joining the rally, and big companies like Google, OpenAI, Meta, and Anthropic are racing to reach the AGI[1] level. Benchmarks and results show that they are reaching that level of intelligence, and it will not be long before we must adapt and incorporate these systems into our daily workforce. This is not just a simple invention, and it is not merely a product or an idea for everyday use; we are witnessing a *"paradigm shift"*. Through this shift, no one is exempt, and sooner or later, everyone will need to integrate AI systems into their workflow. Many people compare this wave of AI to the Dot-com bubble [64] of the late 1990s, but we believe that this shift is more permanent, more akin to the invention of calculators. It is not only helping industries and individuals perform better; it is enabling them to excel.

7.1 Future Generation of Portfolio Management

It is evident that AI is fundamentally reshaping our world, transforming industries, and the financial sector is no exception. In this research, we explored AI's capabilities within one of finance's most critical sub-domains: portfolio management and asset allocation. Our findings demonstrate that large language models (LLMs), especially when orchestrated through agent-based systems, exhibit strong reasoning abilities and can effectively interpret and act upon

[1] Artificial General Intelligence, or simply human-level intelligence

user-specific profiles and instructions. However, we also highlighted their limitations in early phases, particularly in mathematical reasoning and reliance on static, outdated knowledge. With the advent of multi-agent architectures, model context protocol (MCP), and real-time internet access, these systems are evolving beyond the constraints of pre-trained models. They are now capable of observing, analyzing, planning, and executing complex financial tasks, paving the way for more autonomous, adaptive, and intelligent decision-making frameworks in finance.

In the third phase of our research, we demonstrated that by constructing a structured backbone and embedding LLM agents as the core decision-making components, the system can outperform previous approaches while exhibiting greater intelligence, creativity, and user inclusiveness. As shown throughout Chapter 5, it is possible to design a multi-agent model that guides the model through a step-by-step process, enabling it to read, reason, and act based on specific user instructions. This model architecture not only enhances modularity and interpretability but also allows the system to behave more like a human decision-maker, capable of adapting to a wide range of investment goals and constraints.

In this research, we focused specifically on how these systems can be applied to portfolio management and asset allocation tasks, and it became clear that they can be put to effective use. As emphasized throughout this thesis, it is nearly impossible for a human analyst to keep up with the vast and continuous flow of financial news and global data. These AI systems offer valuable support to enhance analysis and decision-making processes. While our results and methodologies were demonstrated solely for research purposes, with the implementation of proper safeguards, privacy measures, and the selection of appropriate tools and models, such systems can be feasibly and responsibly integrated into real-world financial operations.

LLMs and agents now act as catalysts for one another. Rather than relying solely on pre-trained LLMs, enhanced performance and more complex execution can be achieved by integrating external agents. Through the MCP and other agentic frameworks, it is possible to establish multi-agent systems capable of planning, reasoning, and executing entire workflows, from initial concept to final output. In the financial industry, such systems can be applied to a variety of use cases, including algorithmic trading, portfolio optimization, financial advisory, and question-answering agents for domain-specific queries. These multi-agent systems can be configured to operate within secure, privacy-preserving environments while executing structured chains of commands and delivering accurate, context-aware responses.

7.2 Contributions to the Field

This research contributes to the growing intersection of AI and financial decision-making, with a specific focus on portfolio management and asset allocation. Across two phases of published and presented works and a final multi-agent framework, the following key contributions have been made:

1. **Large Language Models: A New Tool for Portfolio Construction:** In the initial phase, we systematically assessed the capabilities and limitations of LLMs in constructing investment portfolios using prompt engineering techniques. This work identified major challenges in mathematical reasoning and structural validity, which laid the foundation for more robust solutions.

2. **Leveraging Large Language Models and Retrieval-Augmented Generation for Enhanced Multi-Asset Portfolio Construction [28]:** Building on these insights, we demonstrated how RAGs can improve portfolio recommendations by grounding LLM responses in external financial data. Our second paper introduced a RAG-enhanced pipeline capable of dynamically incorporating market context, thereby improving accuracy and reducing hallucinations in financial decision support.

3. **Development of a Multi-Agent Framework Using LangGraph:** In the final phase, we proposed and implemented a modular, LangGraph-based multi-agent system that coordinates LLMs, external tools, and validation nodes to autonomously generate, analyze, and explain portfolio allocations. This framework demonstrated flexibility across diverse user profiles and showed competitive performance compared to real-world portfolio managers.

Collectively, this research offers a practical roadmap for deploying LLM-based agentic systems in financial environments and serves as a foundation for future research in autonomous financial reasoning and decision-making.

7.2.1 List of Contributions

1. **Title:** *Large Language Models: A New Tool for Portfolio Construction?*

   Venue: Presented at the Generative AI in Finance Conference, hosted by the John Molson School of Business at Concordia University in collaboration with the Desjardins Centre for Innovation and Financing, October 2024.

2. **Title:** *Leveraging Large Language Models and Retrieval-Augmented Generation for Enhanced Multi-Asset Portfolio Construction* [28]

   Venue: Presented at Computational Intelligence in Financial Engineering and Economics (CIFEr), published in the Proceedings of the 2025 IEEE SSCI Symposium, Trondheim, Norway, March 2025.

3. **Title:** *Agentic Portfolio Construction: A Multi-Agent Framework for LLM-Driven Financial Asset Allocation*

   Venue: IEEE Access (Under review)

7.2.2 Research Limitations

This research is subject to several limitations that reflect the current state and challenges of working with rapidly evolving AI technologies. First, the field of LLMs and agentic frameworks is highly agile, with frequent updates to architectures, tools, and libraries. As a result, systems developed today may become partially obsolete within months, requiring continuous low-level maintenance to remain functional and up to date. Second, despite their impressive capabilities, current LLMs still struggle with mathematical reasoning and precision. While more advanced models like OpenAI's O3 and O4 demonstrate improved reasoning performance, their use is often cost-prohibitive and computationally intensive, making them less practical for time-sensitive tasks (not only financial tasks). Finally, any deployment of such systems must carefully address safety and privacy concerns. Because user data may be processed or retained during interactions, strict safeguards must be in place to ensure confidentiality and prevent misuse, especially in high-stakes domains like finance.

7.3 Future Research Directions

As mentioned earlier, this field is evolving at an exceptional pace, with new tools and techniques being introduced almost daily. This rapid progression makes it difficult to definitively outline future research directions, as emerging algorithms or frameworks may soon render current approaches obsolete or significantly more efficient. However, it is only through experience and experimentation with current architectures and techniques that we can effectively engage with future models and approaches to better serve society. Having spent nearly two years researching within this fast-moving and agile domain, it is evident that while the precise future is uncertain, the possibilities remain expansive. Promising directions include the application of AI in algorithmic trading, the development of intelligent robo-advisors, and the automation of complex portfolio optimization tasks. Being aware of the evolving landscape and preparing for the rapid wave of advancements is a crucial first step to not only survive but also excel in this field. In the near term, one of the most compelling and underexplored areas lies in the integration of the MCP and external tools with LLMs, a combination that holds the potential to accelerate problem-solving and improve accuracy across a range of financial applications.

Appendix A

Prompts, Methodology, and Supplementary details

In this appendix, we first present the full prompts that we used in Chapter 3. These prompts are based on the few-shot learning technique. Few-shot learning technique enables the LLM to use in-context learning more efficiently and unlock the full capabilities of the model. In the prompts, first, we instruct the model to obey some rules and then follow the response creations based on shown examples. Showing examples and guidance will steer the wheels of the model to be in the right direction and avoid hallucination.

Few-Shot Prompt for Portfolio Construction

Instruction: Create two hypothetical funds using S&P 500 tickers and explain your reasoning.

Step 1: Select 20 tickers from the S&P 500 index for each portfolio.

Step 2: Allocate weights to each stock aiming for higher returns than the S&P 500.

Step 3: Ensure the sum of total weights is 1 (100%), with different weights for each stock.

Step 4: Write your tickers in dictionaries (curly braces) with their respective weights. For example, `{"Ticker": Weight}`.

Output:

Tickers: `{"Ticker 1": Weight 1,"Ticker 2": Weight 2, ...}`

Rationale: I selected these companies because they are leaders in high-growth sectors like technology and healthcare. The weights are based on their potential to outperform the market...

Task: Create two hypothetical funds using S&P 500 tickers.

Portfolio 1: High Growth S&P 500 Outperformer

- Select 20 tickers
- Aim for higher returns than S&P 500
- Consider growth potential and market trends
- Mention the weight of each stock in the fund and the total weight should be 1 (100%)
- Weights can be different and not equal

Portfolio 2: Low-Risk S&P 500 Outperformer

- Select 20 tickers
- Aim to outperform S&P 500 with minimal risk
- Focus on stability, dividends, and defensive sectors
- Mention the weight of each stock in the fund and the total weight should be 1 (100%)
- Weights can be different and not equal

Output format for each fund:

Portfolio 1: `{"Ticker 1": Weight 1,"Ticker 2": Weight 2, ...}`

Portfolio 2: `{"Ticker 1": Weight 1,"Ticker 2": Weight 2, ...}`

Rationale: [Brief explanation]

Please ensure the output is in plain text without any code formatting markers like ` ```json ` or ` ```python ` and output the tickers in double quotes with a comma separating each ticker and weight pair. The tickers should be formatted exactly in a dictionary like this example: `{"AAPL": 0.08, "MSFT": 0.07, ...}`.

Few-Shot Prompt for Portfolio Optimization

Instruction: Assign weights to two given hypothetical funds of 20 stocks from S&P 500 index and explain your reasoning.

Role: You are an expert in financial analysis and portfolio management.

Task:

1. Your task is to distribute the weights of two given portfolios to maximize returns and outperform the S&P 500 index.
2. Mention the weight of each stock in the funds and the total weight should be 1 (100%).
3. Ensure the total of all weights equals exactly 1.00 (100%).
4. Aim for higher returns than S&P 500 Index.

Output:

Optimized Portfolio 1: `{"Ticker 1": Weight 1,"Ticker 2": Weight 2,"Ticker 3": Weight 3,"Ticker 4": Weight 4, ...}`

Optimized Portfolio 2: `{"Ticker 1": Weight 1,"Ticker 2": Weight 2,"Ticker 3": Weight 3,"Ticker 4": Weight 4, ...}`

Example 1:

Input Portfolio:

`{"AAPL": 0.3, "MSFT": 0.3, "AMZN": 0.2, "GOOGL": 0.1, "META": 0.1}`

Optimized Portfolio:

`{"AAPL": 0.25, "MSFT": 0.25, "AMZN": 0.20, "GOOGL": 0.15, "META": 0.15}`

Rationale: Slightly reduced overweight positions in AAPL and MSFT to increase allocation to high-growth tech companies GOOGL and META, maintaining a strong tech focus to outperform S&P 500.

Example 2:

Input Portfolio:

`{"JNJ": 0.2, "PG": 0.2, "KO": 0.2, "WMT": 0.2, "MCD": 0.2}`

Optimized Portfolio:

`{"JNJ": 0.25, "PG": 0.20, "KO": 0.15, "WMT": 0.25, "MCD": 0.15}`

Rationale: Increased weights in JNJ and WMT for their strong market positions and growth potential, while slightly reducing KO and MCD to account for changing consumer preferences.

Please provide:

1. The optimized portfolio with redistributed weights (ensuring they sum to exactly 1.00).
2. Please make sure the total weight equals 1.00 (100%) and not more or less.

A.1 Metrics Used For Comparison

Table A.1: Summary of Financial Metrics and Their Mathematical Definitions

Metric	**Definition with Mathematical Expression**
Average Annual Return	Average return per year over a given period: $\bar{R}_{\text{annual}} = \frac{1}{T}\sum_{t=1}^{T} R_t$
Annualized Volatility	Standard deviation of returns scaled to a yearly basis: $\sigma_{\text{annual}} = \sigma_{\text{periodic}} \times \sqrt{N}$
Sharpe Ratio	Risk-adjusted return relative to volatility: $\text{Sharpe} = \frac{R_p - R_f}{\sigma_p}$
Maximum Drawdown	Percentage loss from peak to trough: $\text{MDD (\%)} = \frac{\text{Peak Value} - \text{Trough Value}}{\text{Peak Value}} \times 100$
Alpha	Excess return relative to CAPM expectation: $\alpha = R_p - [R_f + \beta(R_m - R_f)]$
Beta	Sensitivity of portfolio returns to market fluctuations: $\beta = \frac{\text{Cov}(R_p, R_m)}{\text{Var}(R_m)}$
Value-at-Risk (VaR) 95%	Max expected loss with 95% confidence: $\text{VaR}_{95\%} = \mu - 1.645\sigma$
Value-at-Risk (VaR) 99%	Max expected loss with 99% confidence: $\text{VaR}_{99\%} = \mu - 2.326\sigma$

Notation:

- R_t is the return in period t; T is the total number of periods.
- σ is the standard deviation (volatility); N is the number of periods per year.
- R_p is the portfolio return, R_f is the risk-free rate, and σ_p is the portfolio's volatility.
- R_m is the market return.
- μ is the mean return; VaR is the Value-at-Risk at the specified confidence level.

A.2 Total Occurrences and Top Stock Selections For Models

In this appendix, we plot the total occurrences of all eights models and their top 20 stocks from iterations.

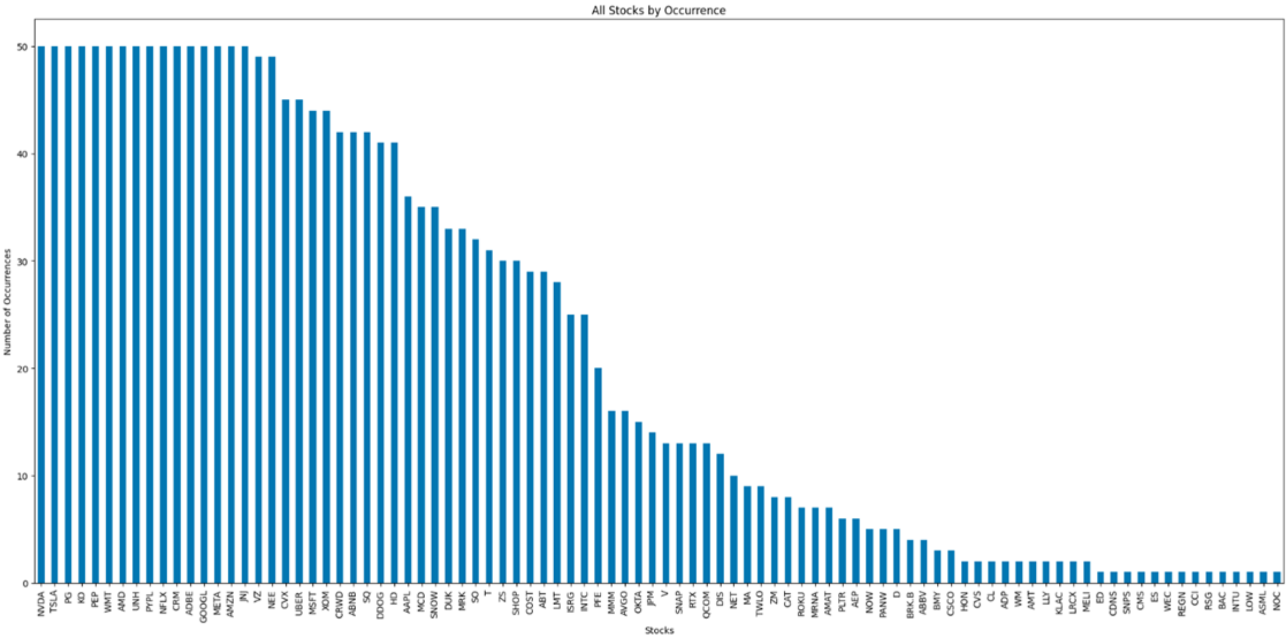
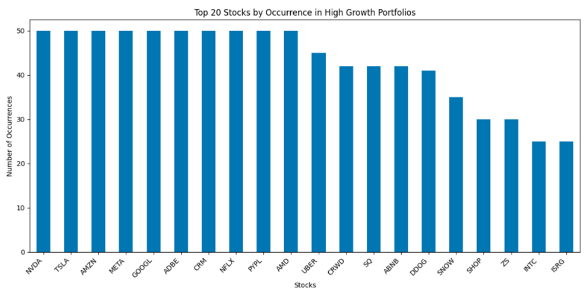
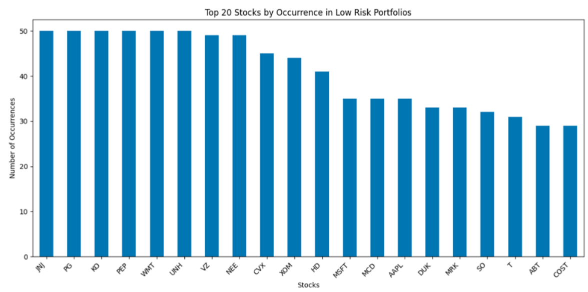


Figure A.1: Total occurrence of stocks in Claude 3.5 Sonnet



(a) Top 20 stocks in high-growth portfolios for Claude 3.5 Sonnet



(b) Top 20 stocks in low-risk portfolios for Claude 3.5 Sonnet

Figure A.2: Top 20 Stocks in high-growth and Low-risk portfolios - Claude 3.5 Sonnet, this model mostly selected aggressive tickers, yet was able to maintain the balance by selecting balanced and defensive tickers

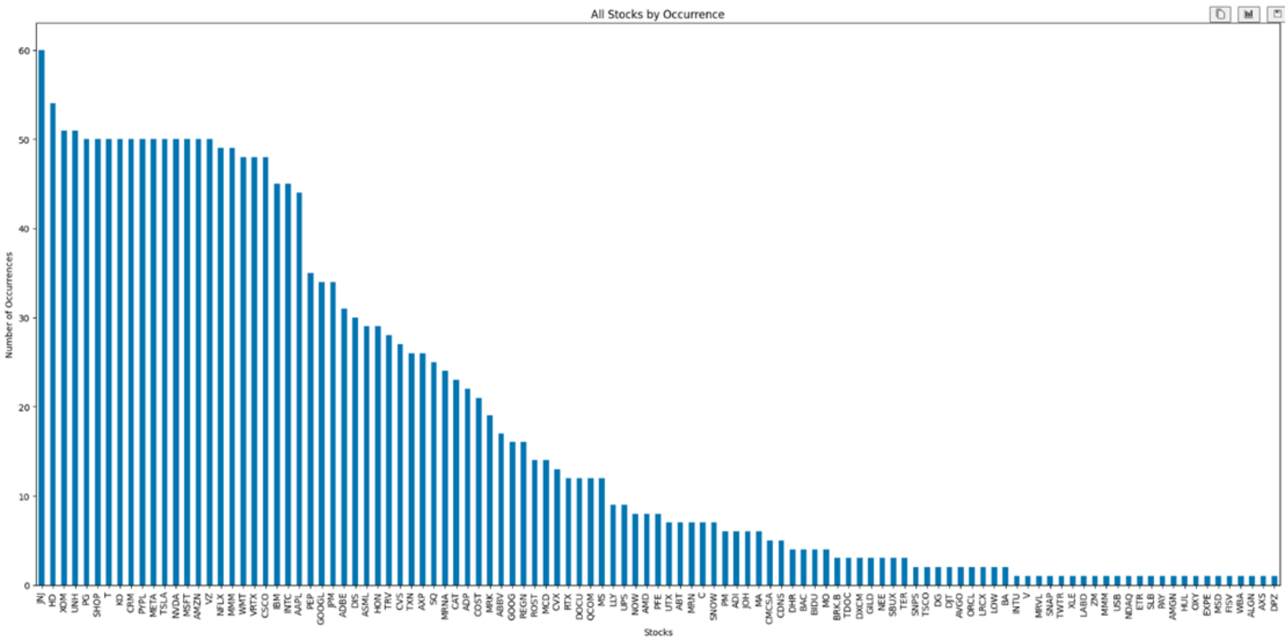
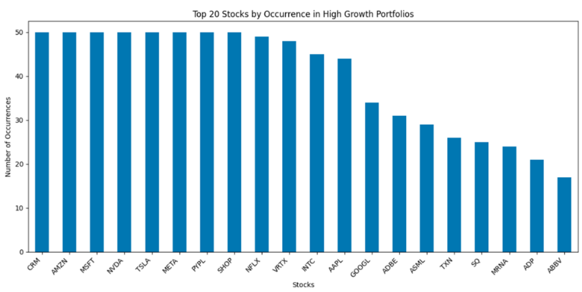
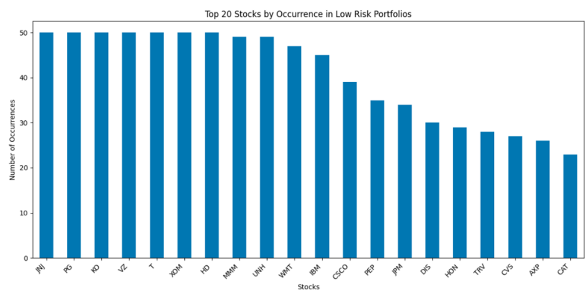


Figure A.3: Total occurrence of stocks in Gemma 2 - 9b



(a) Top 20 stocks in high-growth portfolios for Gemma 2 - 9b



(b) Top 20 stocks in low-risk portfolios for Gemma 2 - 9b

Figure A.4: Top 20 Stocks in high-growth and Low-risk portfolios - Gemma 2 - 9b, this model selected a well-balanced ticker list from both defensive and aggressive tickers

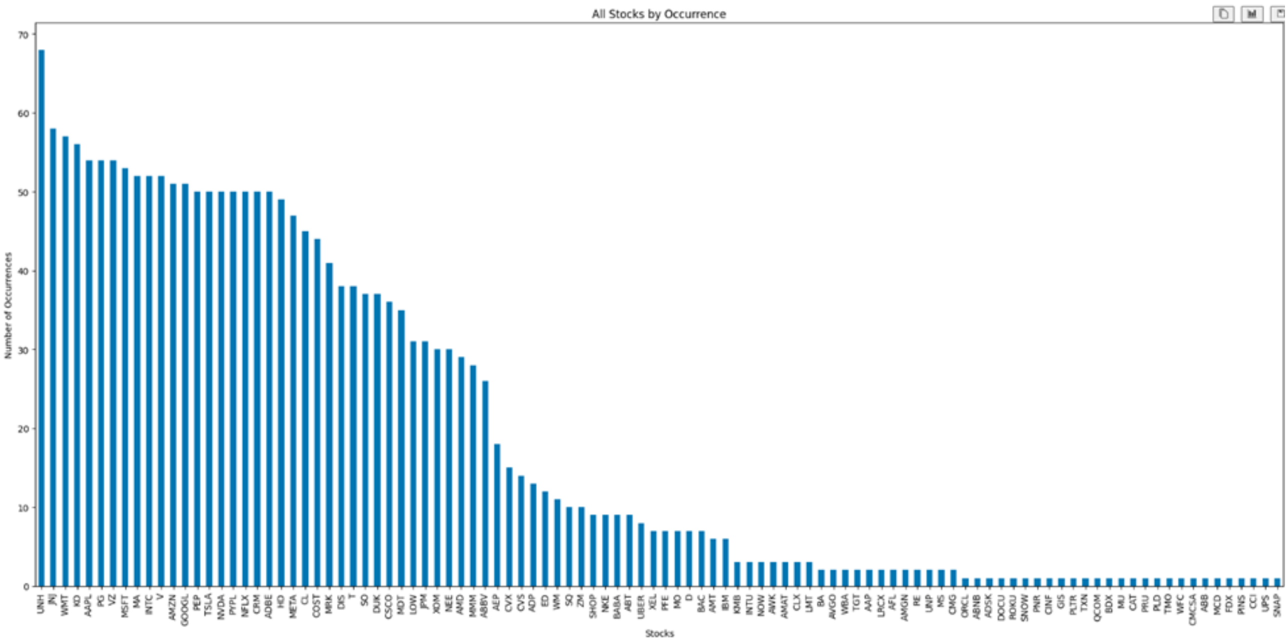
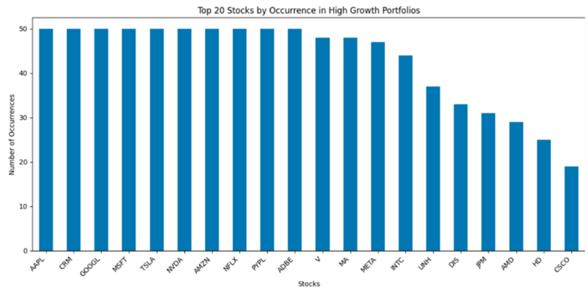
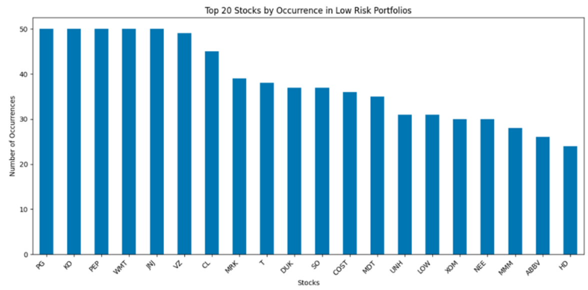


Figure A.5: Total occurrence of stocks in GPT-3.5-Turbo-0125



(a) Top 20 stocks in high-growth portfolios for GPT-3.5-Turbo-0125



(b) Top 20 stocks in low-risk portfolios for GPT-3.5-Turbo-0125

Figure A.6: Top 20 Stocks in high-growth and Low-risk portfolios - GPT-3.5-Turbo-0125, this model was somehow selected more balanced tickers

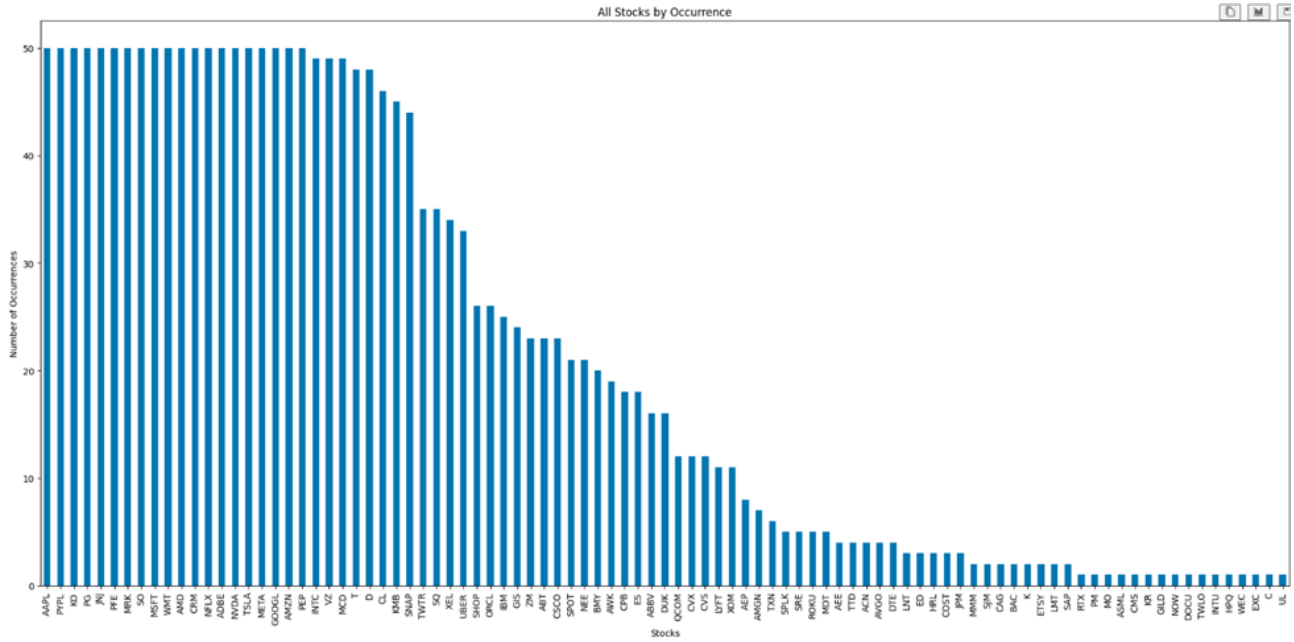
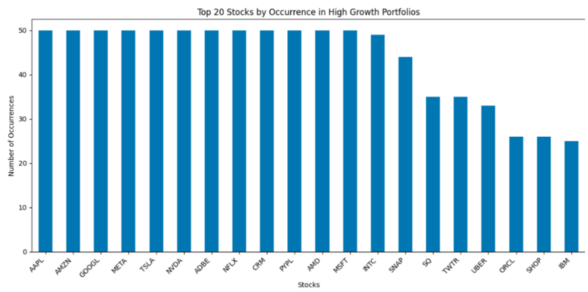
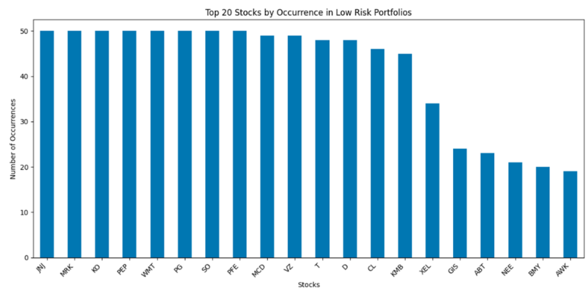


Figure A.7: Total occurrence of stocks in GPT-4-Turbo



(a) Top 20 stocks in high-growth portfolios for GPT-4-Turbo



(b) Top 20 stocks in low-risk portfolios for GPT-4-Turbo

Figure A.8: Top 20 Stocks in high-growth and Low-risk portfolios - GPT-4-Turbo, this model mostly selected aggressive tickers

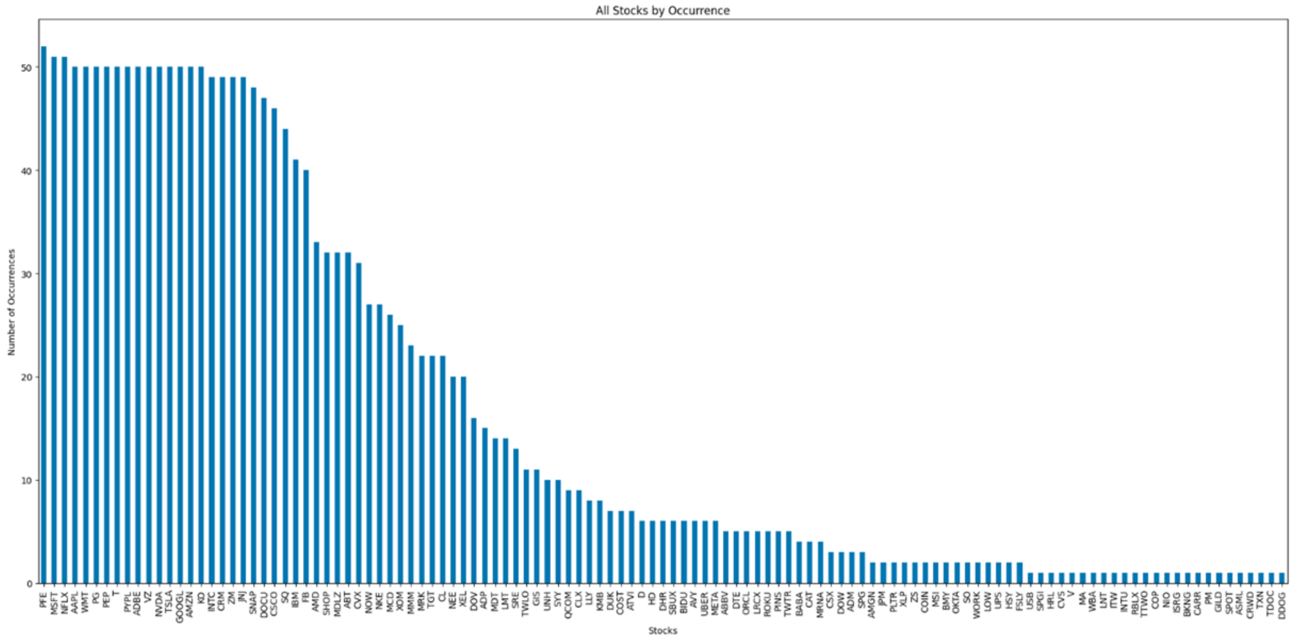
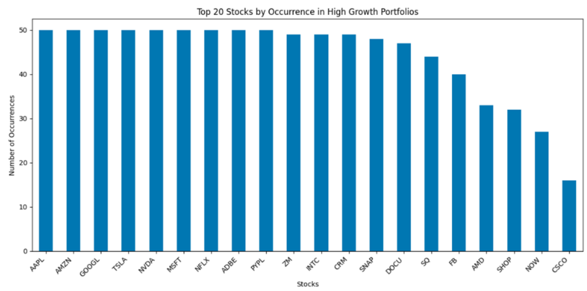
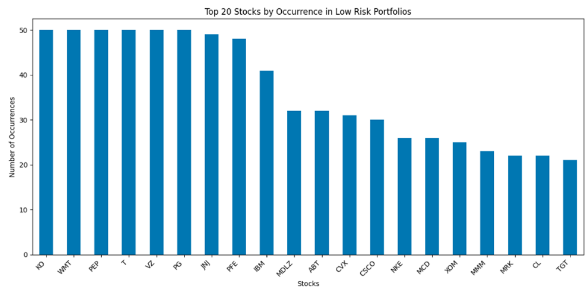


Figure A.9: Total occurrence of stocks in GPT-4o-Mini



(a) Top 20 stocks in high-growth portfolios for GPT-4o-Mini



(b) Top 20 stocks in low-risk portfolios for GPT-4o-Mini

Figure A.10: Top 20 Stocks in high-growth and Low-risk portfolios - GPT-4o-Mini, this model also focused on a mix of aggressive and defensive tickers

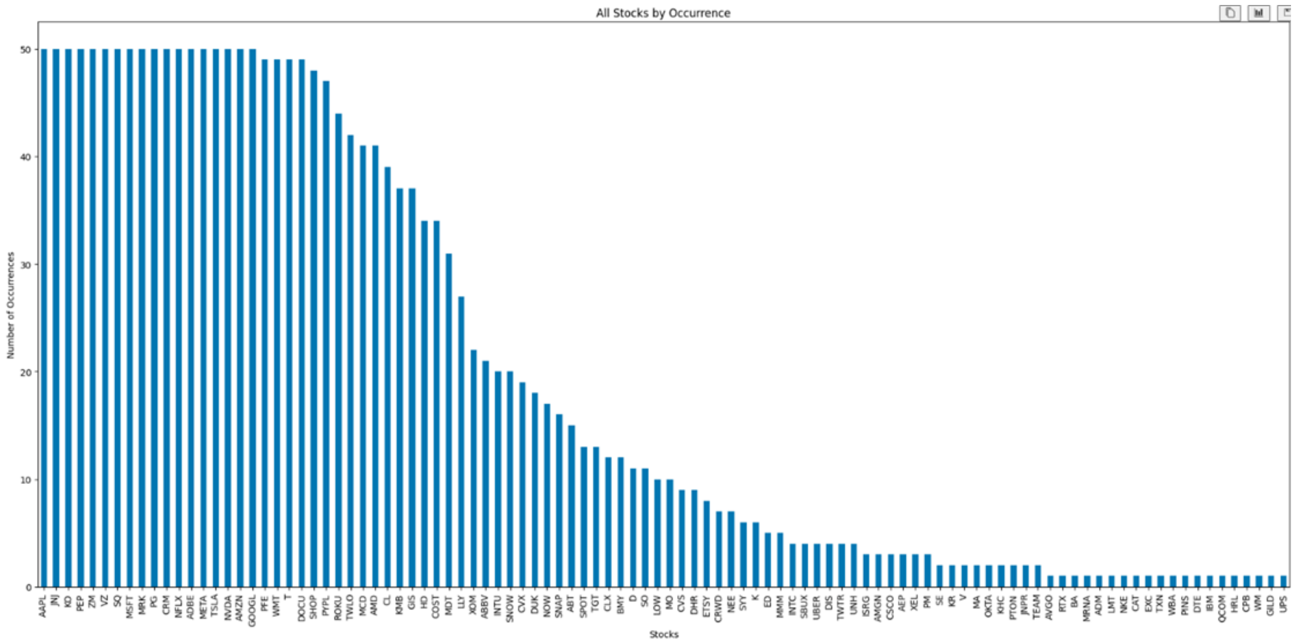
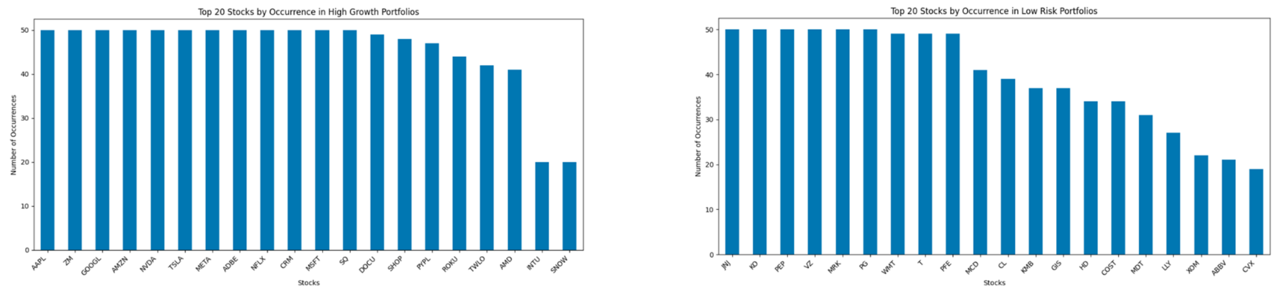


Figure A.11: Total occurrence of stocks in GPT-4o



(a) Top 20 stocks in high-growth portfolios for GPT-4o

(b) Top 20 stocks in low-risk portfolios for GPT-4o

Figure A.12: Top 20 Stocks in high-growth and Low-risk portfolios - GPT-4o, this model selected a diverse list of tickers from defensive tickers to highly-aggressive tickers such as TSLA and NVDA

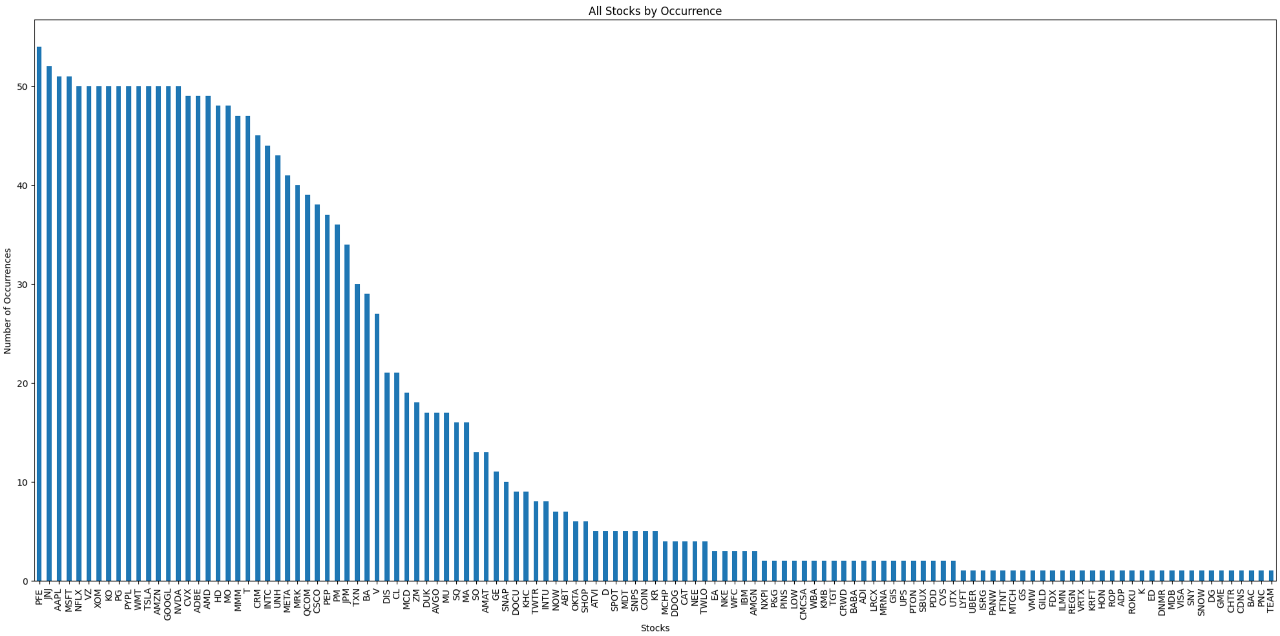
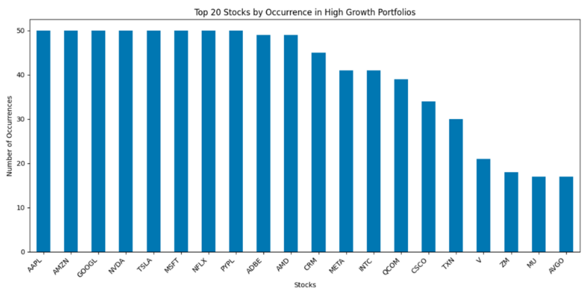
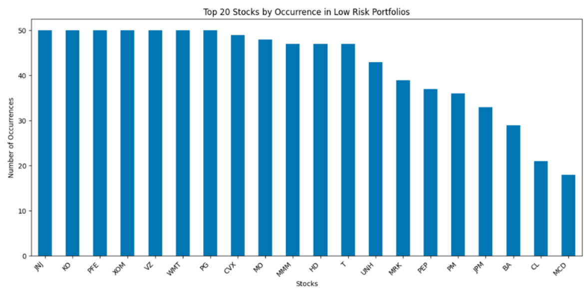


Figure A.13: Total occurrence of stocks in Mistral Large 2



(a) Top 20 stocks in high-growth portfolios for Mistral Large 2



(b) Top 20 stocks in low-risk portfolios for Mistral Large 2

Figure A.14: Top 20 Stocks in high-growth and Low-risk portfolios - Mistral Large 2, this model was also acting surprisingly well in selecting diverse tickers

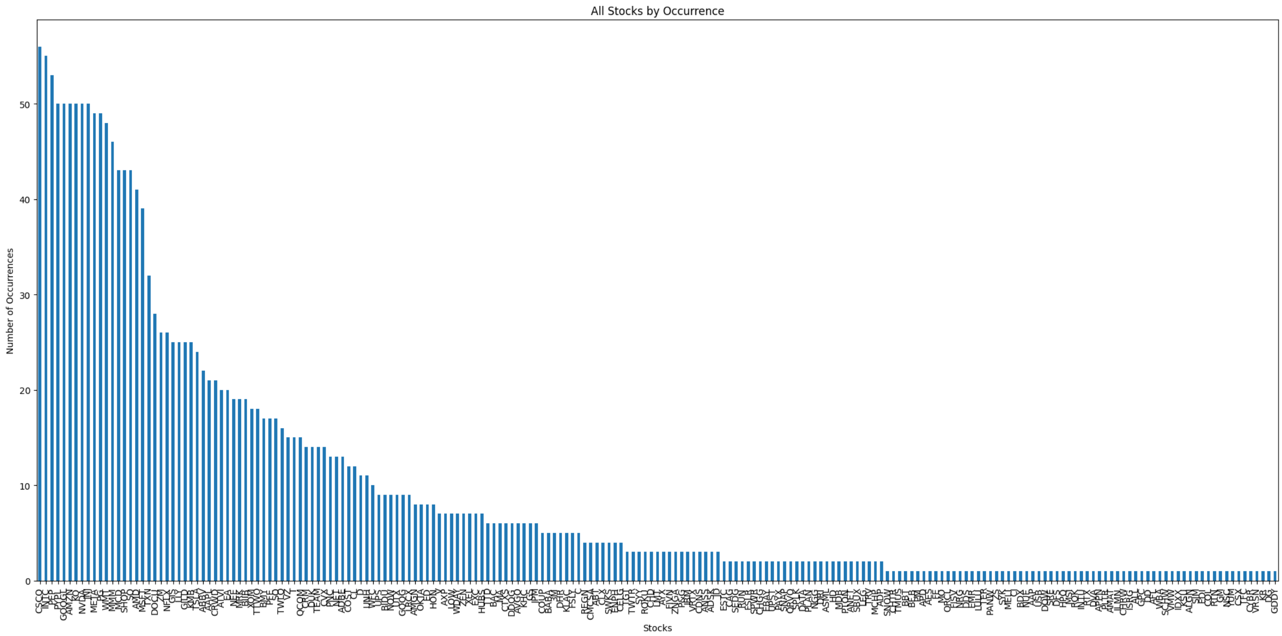
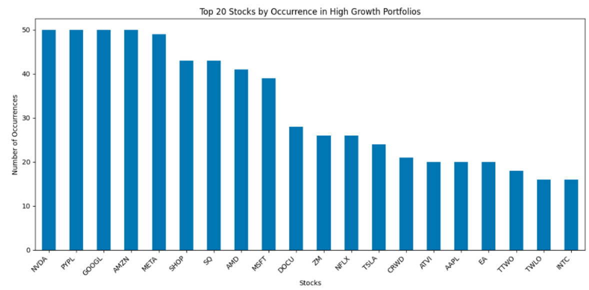
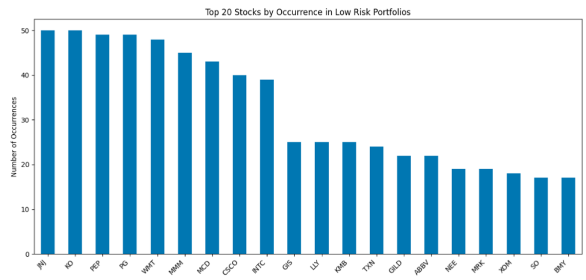


Figure A.15: Total occurrence of stocks in Llama 3.1 405b



(a) Top 20 stocks in high-growth portfolios for Llama 3.1 405b



(b) Top 20 stocks in low-risk portfolios for Llama 3.1 405b

Figure A.16: Top 20 Stocks in high-growth and Low-risk portfolios - Llama 3.1 405b, this model selected the most diversified tickers among all the other models

A.3 System Prompt For LLM Agents

Below, we provide the complete system prompts used to guide the behavior of the LLM-based nodes (agents) within the workflow in Chapter 5.

User Request Parser Agent

System Prompt: Parse User Request

You are an expert financial analyst assistant. Parse the user's request to understand their investment profile. Extract the goal, risk tolerance, time horizon, initial capital, and any specific preferences mentioned (store simple preferences as a string in 'specific_preferences'). Identify any specific assets the user suggested.

Asset Generation Rules (Strict Adherence Required): 1. **CRITICAL & ABSOLUTE REQUIREMENT:** If the user explicitly requests a specific number of tickers (e.g., "select 20 tickers", "give me 10 stocks"), you MUST generate **EXACTLY** that number of diverse assets matching the profile. This instruction overrides any other general guidelines on asset count, including any defaults suggested in field descriptions. The 'suggested_assets' field in your JSON output must reflect this exact count. 2. If the user suggests 5 or more specific assets AND does not specify an exact number, use those primarily, potentially adding more diverse assets to reach a count of around 15. 3. If the user suggests fewer than 5 assets AND does NOT specify an exact number, generate a diverse list of approximately 20 suitable assets (considering stocks, bonds, ETFs relevant to the profile).

Populate the 'suggested_assets' field with the final list of tickers. Ensure the list contains ONLY valid tickers. Output ONLY the JSON object matching the required schema. **IMPORTANT:** Do NOT include any comments (like //) inside the JSON output.

Today's Date: [insert date dynamically]

Portfolio Proposer Agent

System Prompt: Propose Portfolio Allocation

You are an expert portfolio manager. Your task is to propose a portfolio allocation based on the user's profile, available asset data metrics (including historical performance, CAPM expected return, and SMA indicators), and recent market news.

User Profile: [User Profile JSON here]

Available Assets with Data: [list of assets]

Asset Metrics Summary (Historical Return/Vol/Sharpe/Drawdown, Beta, CAPM Expected Return, SMA 50/200, Portfolio Momentum Outlook): [Metrics Summary here]

Recent Market News Context: [News here]

Constraints: - Allocate ONLY among the 'Available Assets with Data'. - Proposed weights MUST sum to 1.0 (or very close to it). - Strive for a diverse range of allocation percentages, reflecting a detailed analysis. For instance, feel free to use precise values like 7.3%, 12.8%, etc., rather than rounding to simpler percentages, if the underlying data and user profile suggest such a nuanced distribution. - Consider the user's goal and risk tolerance foremost. - Also consider the CAPM expected return and the portfolio momentum outlook (SMA trend) when making allocations. - Provide brief reasoning.

Output ONLY the JSON object matching the required schema. Ensure ticker symbols in the output JSON match the available assets exactly.

Commentary/Reporting Agent

System Prompt: Generate Portfolio Commentary

You are a financial advisor AI. Generate a clear and concise commentary explaining the proposed portfolio allocation, its key metrics (including historical performance, CAPM expected return, and momentum outlook), and validation results to the user. If the allocation model provided reasoning, incorporate it. If validation failed, explain the issues.

Context: - User Profile: [User Profile JSON] - Proposed Portfolio: [Proposed Portfolio JSON] - Portfolio & Asset Metrics (Includes Historical, CAPM Exp. Return, SMAs, Momentum): [Metrics JSON] - Validation Result: [Validation Result Text] - Market News Context: [Market News Text] - Initial Allocation Reasoning (if provided): [LLM Reasoning Text]

Instructions: - Explain the reasoning behind the allocation in relation to the user's profile, historical performance, expected returns (CAPM), and the overall portfolio momentum outlook (SMA trend). - Briefly interpret the key portfolio metrics (Return, Volatility, Sharpe, Drawdown, CAPM Expected Return, Momentum Outlook). - Mention the validation outcome. If issues were found, briefly explain them clearly. - Keep the tone informative and objective. - **Include a disclaimer that this is not financial advice.**

Output only the commentary text.

A.4 Supplementary Data and Analysis

The following tables present the full multi-agent generated portfolios referenced in Chapter 6.

A.4.1 Metrics and Perfomance Table For Multi-Agent Generated Portfolios

Table A.2: Asset-Level Metrics for Aggressive Portfolio (Multi-Agent Framework)

Asset	Ann. Return	Volatility	Sharpe	Max Drawdown	Exp. Return (CAPM)	Beta	SMA 50	SMA 200
AAPL	8.03%	33.00%	0.24	-33.36%	10.24%	1.27	207.19	225.57
AMZN	10.62%	34.64%	0.31	-30.88%	10.39%	1.31	190.76	200.00
BABA	44.72%	45.89%	0.97	-32.66%	5.57%	0.24	124.85	103.12
BNS	11.19%	17.86%	0.63	-21.85%	9.88%	1.20	48.52	50.90
EEM	8.37%	19.34%	0.43	-17.29%	7.58%	0.68	43.78	43.31
ENB	28.25%	17.14%	1.65	-9.59%	8.51%	0.89	44.59	42.23
META	35.87%	37.10%	0.97	-34.15%	10.07%	1.24	576.95	588.01
QQQ	14.49%	25.52%	0.57	-22.77%	10.15%	1.26	474.47	492.00
SHOP	78.59%	60.53%	1.30	-40.54%	16.51%	2.67	95.37	94.99
TSLA	97.11%	72.62%	1.34	-53.77%	15.45%	2.43	270.97	297.16
TCEHY	32.01%	37.81%	0.85	-24.88%	7.14%	0.59	63.17	55.62

Table A.3: Asset-Level Metrics for Conservative Portfolio (Multi-Agent Framework)

Asset	Ann. Return	Volatility	Sharpe	Max Drawdown	Exp. Return (CAPM)	Beta	SMA 50	SMA 200
BND	1.49%	6.80%	0.22	-9.60%	4.90%	0.09	72.63	72.15
GLD	20.93%	15.38%	1.36	-13.36%	5.25%	0.17	292.70	258.06
IEF	-0.33%	8.62%	-0.04	-13.13%	4.76%	0.06	94.39	93.56
SPY	16.76%	18.20%	0.92	-18.76%	9.05%	1.01	554.91	572.29
VEA	11.41%	16.86%	0.68	-20.14%	7.95%	0.77	51.76	50.20
VEE.TO	10.23%	14.63%	0.70	-14.97%	6.57%	0.46	38.22	37.83
VT	13.97%	16.75%	0.83	-16.51%	8.58%	0.91	116.01	117.35
XIC.TO	12.12%	13.59%	0.89	-12.48%	7.25%	0.61	39.36	38.93
XSB.TO	4.00%	3.05%	1.31	-2.85%	4.59%	0.02	26.94	26.53
ZAG.TO	3.11%	7.30%	0.43	-7.67%	4.77%	0.06	13.91	13.76

Table A.4: Asset-Level Metrics for Balanced Portfolio (Multi-Agent Framework)

Asset	Ann. Return	Volatility	Sharpe	Max Drawdown	Exp. Return (CAPM)	Beta	SMA 50	SMA 200
AGG	4.76%	5.37%	0.89	-4.82%	4.58%	0.02	97.83	97.20
BMO.TO	17.23%	21.69%	0.79	-16.57%	9.92%	1.21	134.26	129.40
BNS.TO	15.72%	15.46%	1.02	-18.12%	9.88%	1.20	67.84	69.92
EEM	8.62%	19.34%	0.45	-17.29%	7.58%	0.68	43.78	43.31
ENB.TO	35.02%	15.18%	2.31	-9.59%	8.51%	0.89	61.81	57.82
GLD	35.52%	17.93%	1.98	-8.12%	5.02%	0.12	292.71	258.06
IEF	4.74%	6.64%	0.71	-6.89%	4.37%	-0.03	94.39	93.56
RY.TO	26.65%	16.93%	1.57	-14.03%	9.10%	1.02	162.61	164.92
SPY	13.20%	20.31%	0.65	-18.76%	9.14%	1.03	554.91	572.29
TD.TO	22.20%	18.35%	1.21	-15.04%	8.90%	0.98	85.11	80.23
XEF.TO	14.28%	15.54%	0.92	-14.32%	7.19%	0.60	40.43	38.92
XIC.TO	18.72%	14.49%	1.29	-12.27%	7.29%	0.62	39.36	38.93

A.5 Emergence of Transformers and Evolution of Large Language Models

The core breakthrough of GPT models stems from adapting the Transformer architecture, particularly its decoder section, for autoregressive language generation. While Transformers were initially designed for sequence-to-sequence translation, they feature multi-head self-attention mechanisms, position-wise feedforward layers, residual connections, and layer normalization. GPT implementations utilize only the decoder stack and implement causal masking within the self-attention layers, restricting each token's visibility to preceding tokens only. This architectural constraint maintains the autoregressive nature crucial for sequential token prediction, allowing models to produce coherent text in sequential order. GPT systems also employ trainable positional embeddings and Byte-Pair Encoding (BPE) tokenization to effectively encode input sequences for large-scale training efficiency [2]. Regarding the training framework, GPT models maximize the likelihood of token sequences by minimizing cross-entropy loss between predicted and ground-truth subsequent tokens across vast unlabeled text datasets. This straightforward, domain-agnostic objective enables models to organically learn syntactic patterns, semantic relationships, and factual knowledge through self-supervised learning. This approach represents a fundamental shift from traditional NLP methodologies that relied on manually labeled data and specialized architectures for individual tasks. Through a single unified framework and training approach, GPT models can adapt to diverse applications, spanning question answering, text summarization, and logical reasoning, simply by providing contextual prompts. This "pretrain once, prompt anywhere" methodology, supported by scaling laws showing consistent improvements with increased model size and training data [65], drove the transformation from narrow, task-specific systems to broad, multipurpose foundation models.

The evolution of LLMs began with GPT-1, introduced by OpenAI in 2018. GPT-1 applied the Transformer architecture, originally proposed by Vaswani et al. [2], to unsupervised language modeling. While relatively small by modern standards (117 million parameters), GPT-1 demonstrated the effectiveness of pretraining on large text corpora followed by task-specific fine-tuning, marking a significant shift from traditional supervised NLP approaches

In 2019, GPT-2 emerged with 1.5 billion parameters and training on a vastly expanded dataset. This model marked a significant departure from previous approaches by demonstrating the remarkable potential of zero-shot and few-shot learning. GPT-2 proved that one model could tackle diverse NLP tasks simply through clever prompting, eliminating the need for task-specific fine-tuning. This capability to generalize across different applications without additional training fundamentally transformed the field's approach, establishing the influential "pretrain once, deploy everywhere" methodology [66].

GPT-3's arrival in 2020 represented a dramatic escalation, expanding to 175 billion parameters. This iteration garnered global recognition for producing remarkably fluent and contextually sophisticated text across diverse applications, frequently matching human-level quality in creative composition, programming, language translation, and logical reasoning. GPT-3 validated the critical principle that model capabilities improve proportionally with larger datasets and parameter counts, a discovery that would guide the trajectory of future AI development [17].

Subsequently, the LLM landscape expanded dramatically. OpenAI launched ChatGPT, powered by GPT-3.5 and later GPT-4 (now we have reasoning models such as ChatGPT O series), which brought multimodal functionality (processing both text and visual inputs) alongside enhanced instruction adherence and alignment. These systems incorporated methods like Reinforcement Learning from Human Feedback (RLHF) to produce outputs more closely matching human preferences. Simultaneously, rival organizations entered the field: Anthropic created Claude using Constitutional AI frameworks, Google DeepMind unveiled Gemini (previously known as Bard), and Meta contributed open-source alternatives, including LLaMA. Each platform sought to balance competing priorities of accessibility, safety, performance, and computational efficiency.

Current models of ChatGPT, Claude, Gemini, and LLaMA define a new era of LLMs characterized by expanded context windows, multimodal processing, and modular agent architecture. These capabilities unlock applications far beyond simple text generation, encompassing autonomous agents, enhanced search systems, and sophisticated tool integration. The industry has pivoted from emphasizing raw model size toward developing more sophisticated workflows, seamless application integration, and streamlined fine-tuning approaches such as LoRA, RAG, and function calling methodologies.

A.6 Acronyms and Definitions

Book-to-Market Ratio: A financial metric that compares a company's book value (its net asset value) to its market capitalization, commonly used to identify undervalued or value-oriented stocks.

Premium: In finance, a premium refers to the excess return or price paid over a benchmark value, such as the additional return expected for bearing a specific type of risk (e.g., size, value, or market risk).

R_f **(Risk-Free Rate)** : The theoretical return on an investment with zero risk, typically approximated using government treasury securities.

R_m **(Market Return)** : The expected return of the overall market portfolio, often proxied by a broad market index like the S&P 500.

SMB (Small Minus Big) : A factor in the Fama-French models that captures the return difference between small-cap and large-cap stocks, reflecting the size premium.

HML (High Minus Low) : A factor representing the value premium, calculated as the return difference between high and low book-to-market ratio stocks.

RMW (Robust Minus Weak) : A profitability factor in the five-factor Fama-French model representing the return spread between firms with robust and weak profitability.

CMA (Conservative Minus Aggressive) : An investment factor capturing the return difference between firms with conservative and aggressive investment policies.

F_k **(Factor Premium)** : The additional expected return associated with exposure to a specific risk factor k, used in APT models.

Bibliography

[1] Harry Markowitz. Portfolio Selection. *The Journal of Finance*, 7(1):77–91, 1952. Publisher: [American Finance Association, Wiley].

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[3] Meta AI. LLaMA 4: Advancing Open Foundation Models with Multimodal Intelligence. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`, 2024. Accessed: 2025-05-02.

[4] NVIDIA Developer. RAG 101: Demystifying Retrieval-Augmented Generation Pipelines. `https://developer.nvidia.com/blog/rag-101-demystifying-retrieval-augmented-generation-pipelines/`, 2024. Accessed: 2025-05-02.

[5] Nathan Most and Steven Bloom. Exchange traded mutual funds. *The Journal of Index Investing*, 1(1):14–20, 2010.

[6] Meir Statman. How many stocks make a diversified portfolio? *The Journal of Financial and Quantitative Analysis*, 22(3):353–363, 1987.

[7] William F. Sharpe. Capital Asset Prices: A Theory of Market Equilibrium under Condi-

tions of Risk. *The Journal of Finance*, 19(3):425–442, 1964. Publisher: [American Finance Association, Wiley].

[8] Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, 1993.

[9] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. 2018.

[10] OpenAI. Introducing o3 and o4-mini. `https://openai.com/index/introducing-o3-and-o4-mini/`, 2024. Accessed: 2025-05-21.

[11] Anthropic. Introducing claude 4. `https://www.anthropic.com/news/claude-4`, 2024. Accessed: 2025-05-21.

[12] Google Research. Model cards. `https://modelcards.withgoogle.com/model-cards`, 2024. Accessed: 2025-05-21.

[13] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. BloombergGPT: A Large Language Model for Finance, December 2023. arXiv:2303.17564 [cs, q-fin].

[14] Yi Yang, Yixuan Tang, and Kar Yan Tam. Investlm: A large language model for investment using financial domain instruction tuning. *preprint arXiv:2309.13064*, 2023.

[15] Isik Akin and Meryem Akin. Behavioral finance impacts on us stock market volatility: an analysis of market anomalies. *Behavioural Public Policy*, page 1–25, 2024.

[16] Yaqing Wang, Quanming Yao, James Kwok, and Lionel Ni. Generalizing from a Few Examples: A Survey on Few-shot Learning. *ACM Computing Surveys*, 53:1–34, June 2020.

[17] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya

Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners, July 2020. arXiv:2005.14165 [cs].

[18] 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, April 2021. arXiv:2005.11401 [cs].

[19] Anthropic. Introducing the model context protocol (mcp). `https://www.anthropic.com/news/model-context-protocol`, 2024. Accessed: 2025-05-21.

[20] OpenAI. Introducing operator. `https://openai.com/index/introducing-operator/`, 2024. Accessed: 2025-04-29.

[21] William F Sharpe. Mutual fund performance. *The Journal of Business*, 39(1):119–138, 1966.

[22] Eugene F Fama and Kenneth R French. A five-factor asset pricing model. *Journal of Financial Economics*, 116(1):1–22, 2015.

[23] Stephen A. Ross. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, 13(3):341–360, 1976.

[24] Fischer Black and Robert Litterman. Global portfolio optimization. *Financial Analysts Journal*, 48(5):28–43, 1992.

[25] Hersh Shefrin and Meir Statman. Behavioral portfolio theory. *Journal of Financial and Quantitative Analysis*, 35(2):127–151, 2000.

[26] Edward Qian. Risk parity portfolios: Efficient portfolios through true diversification. *PanAgora Asset Management*, 1(1):1–9, 2005.

[27] Joy Dip Das, Avanthi M. Gedara, Sulalitha Bowala, Ruppa K. Thulasiram, and Aerambamoorthy Thavaneswaran. Non-linear data representation with machine learning for dynamic covariance based financial portfolio optimization. In *2025 IEEE Symposium Series on Computational Intelligence (SSCI)*, Trondheim, Norway, mar 2025. IEEE. in press.

[28] Ahmadreza Hajaghie and Ruppa K. Thulasiram. Leveraging large language models and retrieval-augmented generation for enhanced multi-asset portfolio construction. In *2025 IEEE Symposium Series on Computational Intelligence (SSCI)*, Trondheim, Norway, March 2025. IEEE.

[29] Thibaut Théate and Damien Ernst. An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications*, 173:114632, 2021.

[30] Yang Li, Wanshan Zheng, and Zibin Zheng. Deep robust reinforcement learning for practical algorithmic trading. *IEEE Access*, 7:108014–108022, 2019.

[31] Chien Yi Huang. Financial trading as a game: A deep reinforcement learning approach. *preprint arXiv:1807.02787*, 2018.

[32] Golnaz Sababipour ASL, Kiarash Shamsi, Ruppa K. Thulasiram, Cuneyt Akcora, and Carson Leung. Deep learning-based credit score prediction: Hybrid lstm-gru model. In *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 395–400, 2023.

[33] Pradheepan Raghavan and Neamat El Gayar. Fraud detection using machine learning and deep learning. In *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 334–339, 2019.

[34] Jason Kingdon. *Intelligent systems and financial forecasting*. Springer Science & Business Media, 2012.

[35] Golnaz Sababipour Asl, Ruppa K. Thulasiram, and Aerambamoorthy Thavaneswaran. Stock volatility forecasting with transformer network. In *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 90–96, 2023.

[36] Joy Dip Das, Ruppa K. Thulasiram, Christopher Henry, and Aerambamoorthy Thavaneswaran. Encoder–decoder based lstm and gru architectures for stocks and cryptocurrency prediction. *Journal of Risk and Financial Management*, 17(5), 2024.

[37] Ashish Lakhmani, Ruppa K. Thulasiram, and Parimala Thulasiraman. Nature-inspired portfolio diversification using ant brood clustering. In *Proc. (Part I) 27th European Conference on the Applications of Evolutionary and bio-inspired Computation (SPECIES EvoStar)*, pages 115–130, 2024.

[38] Bhanu Sharma, Ruppa K. Thulasiram, and Parimala Thulasiraman. Evaluating value-at-risk using genetic algorithm. *Journal of Risk Finance*, 16(2):170–189, 2015.

[39] Zhengyao Jiang, Dixing Xu, and Jinjun Liang. A deep reinforcement learning framework for the financial portfolio management problem. *preprint arXiv:1706.10059*, 2017.

[40] Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep learning for portfolio optimization. *The Journal of Financial Data Science*, 2(4):8–20, August 2020.

[41] OpenAI. Openai o1 hub, 2024. Accessed: 2025-04-28.

[42] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and many others. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[43] David Balsiger, Hans-Rudolf Dimmler, Samuel Egger-Horstmann, and Thomas Hanne. Assessing large language models used for extracting table information from annual financial reports. *Computers*, 13(10):257, 2024.

[44] Shijie Han, Haoqiang Kang, Bo Jin, Xiao-Yang Liu, and Steve Y Yang. Xbrl agent: Leveraging large language models for financial report analysis. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 856–864, 2024.

[45] Alex Kim, Maximilian Muhn, and Valeri Nikolaev. Financial statement analysis with large language models. *preprint arXiv:2407.17866*, 2024.

[46] Xiao-Yang Liu, Guoxuan Wang, Hongyang Yang, and Daochen Zha. Fingpt: Democratizing internet-scale data for financial large language models. *arXiv preprint arXiv:2307.10485*, 2023.

[47] Oleksandr Romanko, Akhilesh Narayan, and Roy H Kwon. Chatgpt-based investment portfolio selection. In *Operations Research Forum*, volume 4, page 91. Springer, 2023.

[48] Christian Fieberg, Lars Hornuf, and David J. Streich. Using gpt-4 for financial advice. Working Paper No. 10529, CESifo Working Paper Series, July 2023. Available at SSRN: `https://ssrn.com/abstract=4499485`.

[49] Matthias Pelster and Joel Val. Can ChatGPT assist in picking stocks? *Finance Research Letters*, 59:104786, January 2024.

[50] Hyungjin Ko and Jaewook Lee. Can ChatGPT improve investment decisions? From a portfolio management perspective. *Finance Research Letters*, 64:105433, June 2024.

[51] Jang Ho Kim. What if ChatGPT were a quant asset manager. *Finance Research Letters*, 58:104580, December 2023.

[52] Haoqiang Kang and Xiao-Yang Liu. Deficiency of large language models in finance: An empirical examination of hallucination. *arXiv preprint arXiv:2311.15548*, 2023.

[53] Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

[54] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36–45, January 1966.

[55] Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A Rossi, Subhabrata Mukherjee, Xianfeng Tang, et al. Retrieval-augmented generation with graphs (graphrag). *preprint arXiv:2501.00309*, 2024.

[56] Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation. *Under Review for ICLR2025, Openreview: https://openreview.net/forum?id=JnWJbrnaUE*, 2024.

[57] Boyu Zhang, Hongyang Yang, Tianyu Zhou, Muhammad Ali Babar, and Xiao-Yang Liu. Enhancing financial sentiment analysis via retrieval augmented large language models. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, ICAIF '23, page 349–356, New York, NY, USA, 2023. Association for Computing Machinery.

[58] Ivan Iaroshev, Ramalingam Pillai, Leandro Vaglietti, and Thomas Hanne. Evaluating retrieval-augmented generation models for financial report question and answering. *Applied Sciences (2076-3417)*, 14(20), 2024.

[59] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, October 2024.

[60] Uber Engineering. Query-GPT: Using LLMs to Understand Data Intents at Uber. `https://www.uber.com/en-CA/blog/query-gpt/`, 2024. Accessed: 2025-05-02.

[61] Google Developers. A2A: A new era of agent interoperability. `https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/`, 2024. Accessed: 2025-04-29.

[62] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*, 2024.

[63] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[64] Wikipedia contributors. Dot-com bubble — wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/Dot-com_bubble`, 2024. Accessed: 2025-05-21.

[65] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[66] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.