# Integrating Large Language Models with Deep Reinforcement Learning for Portfolio Optimization

Renad Alsweed, Mohammed Alsuhaibani
Department of Computer Science-College of Computer, Qassim University, Buraydah 51452, Saudi Arabia

***Abstract*—This paper explores the application of Deep Reinforcement Learning (DRL) and Large Language Models (LLMs) to portfolio optimization, a critical financial task requiring strategies to balance risk and return in volatile markets. Traditional models often struggle with the complexity of financial markets, whereas Reinforcement Learning (RL) provides end-to-end frameworks for learning optimal, dynamic trading policies through sequential decision-making and trial-and-error interactions. The study examines key DRL algorithms, including Q-learning, Deep Q-Networks (DQN), Proximal Policy Optimization (PPO), and Twin-Delayed Deep Deterministic Policy Gradient (TD3), emphasizing their strengths in dynamic asset allocation. Crucial components of financial RL systems are discussed, such as state representations, reward function designs, its algorithms, and main approaches. Furthermore, the survey investigates how LLMs enhance decision-making by analyzing unstructured data (like news and social media) for sentiment and risk assessment, often integrating these insights to augment state representations or guide reward shaping within DRL frameworks.**

***Keywords*—Portfolio optimization; Reinforcement Learning (RL); algorithmic trading; Markov Decision Process (MDP); Large Language Models (LLM); deep learning**

## I. Introduction

Portfolio management is a crucial aspect of finance, requiring that investors monitor market dynamics and make informed decisions to periodically allocate their portfolios across multiple assets. The goal is to gain profits while maintaining the investment portfolio's stability. Adjusting funds between long and short positions is particularly important as it directly impacts both profit and risk under varying market conditions. Long positions involve buying assets with the expectation of price increases, whereas short positions involve selling borrowed assets to repurchase them at a lower price. In a volatile market, reallocating funds from long to short positions can help hedge against potential losses. This dynamic adjustment helps maintain a balanced portfolio, mitigating risks while seizing opportunities during market downturns and upswings. Margin trading allows traders to borrow funds from a broker to trade financial assets, enabling them to leverage their positions and potentially amplify their returns. However, both long and short positions come with increased risk, as losses are magnified and can exceed the original investment.

The volatility of financial markets still poses a significant challenge, making it difficult for static trading strategies to maintain a long-term competitive advantage. Traditional quantitative models can struggle to fully capture the complex interplay of diverse factors.

Reinforcement learning (RL) emerged as a transformative approach for financial trading, enabling dynamic strategy optimization in complex markets [1]. Driven by the alignment between the sequential decision-making process of trading and the learning process of RL [2], it provides algorithms that adapt and optimize trading strategies based on sequential decisions [1]. Unlike traditional rule-based systems, RL models can navigate complex market environments, continuously updating their strategies to maximize returns.

The RL agent can learn optimal policies by interacting with the environment, continuously updating its policy to maximize returns. Deep Reinforcement Learning (DRL) leverages deep neural networks to approximate value functions and policies, making RL scalable and efficient in complex tasks [1]. The reward function in RL can be designed to encourage both profitability and stability, including a profit-based reward, a balance stability penalty, and a transaction cost penalty [1]. RL agent's portfolio allocation dynamics reveal its ability to adjust asset weights over time, increasing exposure to stable assets during high volatility and balancing for diversification [3]. Yet, frequent adjustments of portfolio ratios can introduce higher volatility and lower returns, making portfolios unstable and unbalanced. Also, designing a reward function that accurately accounts for the different factors influencing the entire stock market can be difficult [2].

Large Language Models (LLMs) are trained on massive and diverse datasets, enabling them to understand instructions and perform a wide range of tasks with zero-shot or few-shot learning capabilities. Their ability to process complex natural language inputs allows them to interpret, generate, and summarize financial information efficiently.

In the financial domain, LLMs are increasingly applied to automate report and workflow generation, forecast market trends, perform entity recognition and sentiment analysis, and provide personalized financial advice or question-answering services. In trading, LLMs can analyze unstructured data sources such as news articles, social media sentiment, and earnings reports to extract market signals and support strategy development. They can assist in generating trading insights, detecting anomalies, or summarizing market movements in real time. Some advanced implementations even combine LLMs with quantitative models or reinforcement learning agents to enhance decision-making and risk management. Beyond analytics, LLMs offer enhanced reasoning, interactivity, and integration capabilities. They provide interpretable explanations for their outputs, increasing transparency and user trust. Their conversational design allows traders and analysts to iteratively refine queries and receive contextualized insights. Moreover, LLMs can seamlessly integrate with trading platforms, APIs, and financial databases using techniques such as Retrieval-Augmented Generation (RAG) to deliver comprehensive analyses and support intelligent, data-driven trading decisions [2].

This survey aims to explore the application DRL and LLMs in portfolio optimization. It reviews how RL methods adapt strategies to maximize returns and manage risk in volatile financial markets, integrating diverse and dynamic data sources. Additionally, it examines how LLMs can enhance decision-making through financial text analysis, and sentiment extraction. The survey also covers key algorithms, reward functions, prompting techniques and evaluation metrics, while comparing the strengths and limitations of different models.

## II. Research Objectives and Main Questions

The aim of this study is to examine the role of diverse data sources, review DRL algorithms and reward functions, and identify evaluation metrics in the context of DRL-based portfolio optimization. It also seeks to compare existing approaches and highlight the challenges and limitations of applying DRL methods in real-world financial markets. The following section outlines the specific research objectives and questions that guide this survey.

- To examine existing diverse data sources and their roles in enhancing DRL-based portfolio methods.
    - RQ1: What are the existing data sources used within DRL for the portfolio optimization problem?
    - RQ2: How do diverse data sources contribute to improving the effectiveness of DRL-based portfolio strategies?
- To review DRL algorithms used in the literature.
    - RQ3: What types of algorithms are most effective for this task?
- To review reward functions used in the literature.
    - RQ4: How do different reward function designs influence portfolio performance and risk management?
- To identify evaluation metrics commonly used and compare existing studies.
    - RQ6: What evaluation metrics are commonly used to assess DRL-based portfolio models?
    - RQ7: How do variations in data sources, reward functions and algorithms affect measurable outcomes in the literature?
- To identify challenges and limitations in existing DRL-LLM approaches for portfolio optimization.
    - RQ8: How LLM has been integrated with a DRL methods?
    - RQ9: What challenges and limitations exist in applying DRL-LLM approaches to real-world financial markets?

## III. Methodology

To fulfill the main objectives and address the research questions introduced in Section II, which define the scope of this review, we follow a systematic methodology presented in this Section III-A specifies the eligibility criteria, Section III-B details the search approach, Section III-C covers the selection process for studies, and Section III-D provides the data extraction process.

### A. Eligibility Criteria

The eligibility criteria were defined in alignment with the objectives and guiding questions presented in II. Studies falling within the scope of this work, particularly those focusing on the stock portfolios, were considered for inclusion. The specific criteria used to determine study eligibility are detailed below:

- Studies address all essential components, including its data source, the algorithm used, the reward function and algorithm evaluation.
- Studies written in English.
- Studies published in scholarly peer-reviewed journals.

The following exclusion criteria were applied to remove papers from this review:

- Review papers.
- Studies that didn't address all essential components mentioned in inclusion criteria.
- Studies written in non-English.
- Studies, thesis and dissertation that are not published in scholarly peer-reviewed journals.

### B. Search Strategy

The method for study retrieval has two primary phases. First, We have identified the primary keywords to be employed in the search process as follows:

- "deep reinforcement learning" AND ("single stock trading" OR "portfolio") AND "stock" AND ("large language" OR "LLM")

Subsequently, we utilized several databases, including Word of Science, ACM, ACL Antholog, ScienceDirect, Springer, IEEE, and MDPI, Google Scholar, arXiv to achieve thorough coverage of the relevant literature.

### C. Study Selection Strategy

This review follows a study selection strategy based primarily on the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines [4], which provide researchers with a framework for conducting high-quality reviews. The PRISMA-based selection process, summarized in Algorithm 1, is organized into three main levels: identification, screening, and inclusion. This algorithm outlines the automated and manual steps followed to retrieve, filter, and finalize the studies included in this review. Each level is described in detail in the following subsections.

*1) Identification level:* 915 papers were collected based on the method described in Section III-B. 30 have been eliminated due to duplication which come from two main reasons. First, authors may publish their papers within multiple journals as well as that journals may found in multiple indexers. Second, portfolio keywords are used interchangeably in many cases, hence same paper appears multiple times within different searches. After de-duplication only 885 papers are moved to next level.

*2) Screening level:* We have followed the next three steps within this level.

*a) Automated screening by eligibility criteria:* Due to large number of studies, we started with removing studies that doesn't match some of eligibility criteria before manual screening. Eliminated papers were either a non-english papers, and non-indexed papers in our listed databases, specifically filtering out from google scholar, as well as review and unpublished thesis and dissertations. Following this process 204 study has been kept.

*b) Manual screening by title and abstract:* This process involved excluding papers that were outside the scope of this review by examining their titles and abstracts. Papers that did not match the keywords defined in the search strategy were also removed. Following this screening, a total of 12 papers were retained for further analysis.

*c) Manual screening by eligibility criteria:* This stage involves assessing the eligibility criteria, where only papers that meet the predefined inclusion criteria are manually retained. Consequently, papers that satisfy the exclusion criteria described in Section III-A are removed from the review. With finalizing screening process we found all 12 papers categorized as eligible.

*3) Inclusion level:* This stage comprises all papers that passed the screening process. Based on the screening results, a total of 12 papers were selected for inclusion in this review and were organized using Notion software.

### D. Data Extraction Strategy

In this phase, we systematically extracted key points essential for addressing our research questions and achieving a comprehensive understanding. These points were organized and recorded in an Notion databases, facilitating efficient data management and retrieval for each paper.

**Algorithm 1:** Systematic Paper Selection and Screening Workflow

**Input:** Search keywords: "deep reinforcement learning", "portfolio", "LLM", "stock"
**Output:** Final list of included papers
**Step 1: Retrieval**
Query databases (WoS, ACM, IEEE, Springer, MDPI, ScienceDirect, arXiv, Google Scholar);
Merge results, standardize columns, and convert to lowercase;
**Step 2: Deduplication**
Group by title; retain longest metadata and first DOI/URL;
Remove duplicates;
**Step 3: Automated Filtering**
Discard if: non-English, non-indexed, or labeled as review/thesis/unpublished;
**Step 4: Topic Relevance**
Tag each paper for keywords (`LLM`, `DRL`, `Portfolio`, `Stock`);
Keep only papers satisfying all four tags;
**Step 5: Manual Validation**
Verify each remaining paper against inclusion criteria and scope;
Remove inconsistent ones;
**Step 6: Data Extraction**
Extract metadata (title, algorithm, dataset, reward, results);
Export the final list for synthesis;

## IV. Literature Review

Reinforcement learning (RL) is a machine learning approach where an agent learns to make sequential decisions by interacting with an environment, aiming to maximize cumulative rewards [1], [5]. This interaction is often framed as a Markov Decision Process (MDP), defined by a tuple of components (S,A,P or T,R,$\gamma$). In an MDP, States (S) represent different configurations of the environment that the agent observes at each time step. Actions (A) define the possible decisions or actions the agent can take to interact with the environment. There is a Transition Probability (P) which indicates the probability of transitioning from one state to another given a specific action. The Reward Function (R) provides feedback on the outcome of actions, helping the agent learn which decisions yield high or low rewards. The objective of an RL agent is to discover an optimal policy, a mapping from states to actions that maximizes the expected cumulative reward over time [1]. This section demonstrates different designs of the Markov Decision Process components introduced earlier as well as value and policy designs.

### A. Reward Function

The design of the reward function is crucial as it guides the agent's learning process towards desired outcomes, such as maximizing profit, managing risk, or aligning with market signals [1], [5]. Different frameworks utilize distinct reward mechanisms depending on their specific objectives.

*1) Multi-objective reward:* The reward function can be designed to balance profit-seeking, stability, and cost minimization by considering transaction costs at once. To accomplish this design, three components need to be combined in one equation. Profit-based reward, which is based on the net change in the agent's account balance, including both the value of held shares (unrealized gains) and cash from sold shares (realized gains). Balance stability penalty a penalty that is applied based on the deviation from the initial balance to discourage large swings and reward more stable account values. Transaction cost penalty, which is a penalty proportional to the trade amount to simulate real-world transaction fees and incentivize minimizing unnecessary trades [1].

*2) Profit maximization:* In another approach for portfolio management using PPO, the immediate reward is defined as the logarithmic portfolio return [3], as shown in Eq. (1).

$$r_{n+1} = \log\left(\frac{w_n \cdot S_{n+1}}{w_n \cdot S_n}\right) \tag{1}$$

where $w_m$ is the current portfolio weights, $S_n$ is current adjusted closing prices, and $S_{n+1}$ is the next day adjusted closing prices

Alternatively, the relative return can be used to calculate logarithmic portfolio return as Eq. (2) shows.

$$r_{n+1} = \log(1 + R_{n+1}) \tag{2}$$

where $R_{n+1}$ is the relative return.

*3) Percent change in the portfolio value:* To avoid arbitrary effects based on starting cash and value-at-risk, all agents start with zero dollars (and may spend negative) and a fixed offset is added to all portfolio value calculations to keep the percent changes on a similar scale. This is calculated every time the agent takes an action [5].

*4) Risk-adjusted return:* Sharpe ratio measures the excess return obtained for every unit of risk. The portfolio generates higher returns in proportion to the level of risk assumed when the SR value is higher [2] such in Eq. (3).

$$\text{SR} = \frac{R_p - R_f}{\sigma_p} \tag{3}$$

where $R_p$ is the return of portfoliom, $R_f$ is the risk-free rate and $\sigma_p$ is standard deviation of the portfolio's excess return.

In the FLAG-TRADER framework, the immediate reward is defined based on the daily change in the Sharpe ratio. Specifically, the reward R(st, at) at state st after taking action at is calculated as $SR_t - SR_{t-1}$, where $SR_t$ is the Sharpe ratio at day t, computed using the historical Profit and Loss (PnL) data up to that time [6].

The reward function in [7] evaluates how good a trading decision is by looking at how much the total portfolio value changes after making a trade. It rewards actions that increase the portfolio's value and penalizes those that decrease it. To make the measure realistic, it subtracts small trading costs (0.1% of the trade value) every time a buy or sell occurs, and it also includes a penalty if the portfolio experiences a drawdown, meaning a drop from a previous high. Eq. (4) shows the followed formula.

$$r_t = \left((p_{t+1}^T n_{t+1}) - (p_t^T n_t) - c_t\right) - d_t \tag{4}$$

where $p$ is the closing price, $n$ is number of shares owned, $c$ is transactional cost, and $d$ is the drawdown penalty.

*5) Strategy ranking reward:* In a multi-strategy switching model, an agent selects among different trading strategies. The reward value is determined by the ranking of the chosen strategy. The strategy ranked first receives a reward of 1, and the rewards for other strategies decrease sequentially. The ranking is calculated based on a composite score over time, which includes metrics like Sharpe ratio, maximum drawdown, total return rate, annualized return rate, and annualized volatility, weighted appropriately [8].

*6) Trend-heuristic reward:* Incorporates trend heuristics into reward function. This trend-heuristic reward function is designed to make the agent sensitive to stock price movements, thereby influencing the RL algorithm's optimization objective. Its purpose is to enhance profitability by aligning the agent's policy with observed price trends [9] as illustrated in Eq. (5).

$$r_t = \ln\left(\mu_t \times a_{t-1}^\top v_t\right) + \omega \times a_{t-1}^\top [0.5; \hat{y}_t] \tag{5}$$

where $\mu_t$ is the transaction cost, $a_{t-1}$ is the action taken by agent, $v_t$ price relative vector, $\omega$ is weight of portfolio potential, $0.5$ is the constant cash bias, and $\hat{y}_t$ is the trend heuristic.

*7) Sector performance reward:* When designing this return function by [10], two key considerations are taken into account. First, to enhance predictions of sector performance changes on current trading days, a cross-entropy structure is incorporated into the denominator. Second, the output is multiplied by the actual percentage change in the numerator to reward substantial shifts in sectors. This approach aims to promote sector rotation while maintaining accuracy in forecasting performance changes. Eq. (6) shows the formula.

$$r_m^t = \frac{p_m^s \cdot \text{pctChg}_m^t}{(p_m^s - \text{pctChg}_m^t)^2 + 1} \tag{6}$$

where $p_m^s$ represents the output of the graph layer, while $\text{pctChg}_m^t$ is the price change for sectors.

*8) Sector profit maximization reward:* The reward here is defined based on the closing and opening prices of a sector on trading day t. It utilizes a softmax function to determine the rewards for different sectors, where higher rewards are given to the sector that exhibits the most significant increase in closing prices compared to opening prices. This structure is designed to inform a portfolio management strategy that involves buying at the opening and selling at the close of each trading day [10] as shown in Eq. (7).

$$r^t = \text{SoftMax}\left(\frac{c_0^t}{o_0^t}, \ldots, \frac{c_m^t}{o_m^t}\right) \tag{7}$$

where $c_0^t$ the closing price and $o_0^t$ is the opening price.

### B. Algorithms

Reinforcement Learning (RL) agents learn to make sequential decisions through trial and error, aiming to maximize their expected cumulative rewards in an environment. This learning process fundamentally involves maintaining a balance between exploration (trying new actions to discover potentially better strategies) and exploitation (using the current best known policy). RL algorithms are often broadly categorized based on what they learn: a value function, a policy function, or a combination of both.

*1) Value-based methods:* Value functions are central to reinforcement learning, estimating the expected cumulative rewards associated with states or state-action pairs, and guiding the agent in making decisions that maximize long-term rewards.

- State Value Function ($V_\pi(s)$): Calculates the expected return when starting from a specific state (s) and following a given policy ($\pi$).

- Action-State Value Function ($Q_\pi(s, a)$): Represents the expected return of taking a specific action (a) in a particular state (s) and subsequently following policy ($\pi$).

Value-based methods focus on learning the value of states or state-action pairs. The agent's policy is then typically derived from this learned value function, often by choosing the action that is estimated to yield the highest future reward (a greedy policy).

Q-learning is a foundational value-based algorithm described as an experimental approach to optimizing action selection. It is off-policy, greedy, and model-free. The agent iteratively estimates the Q-function by optimizing a version of the Bellman equation, representing the expected sum of immediate and discounted future rewards from taking an action and following the current policy [5].

Deep Q-Network (DQN) is a significant variant that uses deep neural networks to approximate the Q-function ($Q_\pi(s, a)$). This allows DQN to handle complex, high-dimensional state and action spaces. DQN employs techniques like experience replay and fixed Q-targets to improve stability during learning [8].

*2) Policy-based methods:* Policies are functions that determine the agent's action based on the current state. Policy-based methods directly learn and optimize a policy function without necessarily explicitly learning a value function. Policies can be deterministic, where the network outputs a single action for a given state. Or stochastic, outputting a probability distribution over possible actions for a given state. Stochastic policies are typical for environments with discrete action spaces, where the network outputs probabilities for each action. For environments with continuous action spaces, the policy network typically outputs continuous values representing the action [9].

Policy gradient methods are a class of RL algorithms that directly optimize the parameters of the policy function. They adjust parameters in the direction of the gradient of the expected cumulative reward [9].

*3) Actor-critic methods (Hybrid):* Actor-Critic methods combine the strengths of both value-based and policy-based approaches. They typically consist of two components: an "actor" which is a policy network that selects actions, and a "critic" which is a value function network that estimates the value of states or state-action pairs . The critic's value estimates are used to evaluate the actor's chosen actions and guide the actor's policy update [6].

The critic learns a value function (either state-value or action-value) to evaluate the actor's policy. This evaluation, often in the form of an advantage estimate (how much better the taken action was compared to the expected value of the state), is used to update the actor's policy parameters [6].

Proximal Policy Optimization (PPO) is a widely used policy optimization algorithm and a type of policy gradient method. It is known for its stability and efficiency. PPO stabilizes learning by clipping the probability ratios between the new and old policies, preventing large, destabilizing updates. PPO is specifically noted as being effective for continuous action spaces. PPO is used as the core RL algorithm in several frameworks [1], [11], [12], [3], [6].

Twin-Delayed Deep Deterministic Policy Gradient (TD3) [5] utilized the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm as the foundation for their strategic market agent. TD3 extends the Deep Deterministic Policy Gradient (DDPG) framework, which enables learning in continuous state and action spaces through an actor–critic architecture. The actor network deterministically maps each state to an action, while the critic network evaluates their expected return. TD3 enhances this approach by employing two critic networks to counteract Q-value overestimation and by delaying policy updates, leading to more stable and reliable learning performance.

### C. Approaches

A basic workflow of reinforcement learning for portfolio trading begins with the agent observing the environment state, which typically includes market data and, in many cases, sentiment information extracted from news using an LLM model or API. The agent also tracks its internal state, such as cash balance, portfolio holdings, or net worth. This internal information can be utilized for reward calculation, portfolio evaluation, and ensuring only valid actions are taken. It can also be combined with the environment state to apply the agent's policy to select an action, which may involve buying, selling, holding, or rebalancing portfolio weights. The environment then updates according to the new market conditions, and the agent receives a reward signal that reflects one or more of the portfolio optimization objectives, such as profit maximization, risk minimization, and transactional cost reduction. This reward guides the agent in updating its policy, forming a continuous cycle of state observation, action execution, and policy refinement.

Following this workflow, multiple papers have been contributed by employing different LLM models, prompting LLM in various techniques, to find different sentiment types, such as plural sentiment or stock recommendations, risk assessment,

TABLE I. KEY DRL COMPONENTS FOR PORTFOLIO OPTIMIZATION

| Paper | Algorithm | Agent State | Environment State (dataset) | Action |
|---|---|---|---|---|
| [1] | PPO | Cash balance, Shares quantity or Net worth based on trading setting (portfolio or SST), Cost basis | Price data, Volume, Weighted signed scoring of plural sentiments (News headlines) [Last 5 states] | Trading signal, Trade proportion (balance/share) adjusted by 0.1 of sentiment score |
| [2] | A2C | long/short positions, Cash balance, Shares quantity | Close price data, Macroeconomic indicator, Microeconomic indicator, Technical indicators, Distinct signed scoring of market trends (News) | Share quantity rebalancing for each position separately |
| [3] | PPO | Cash balance, Portfolio Weights | Weighted signed scoring of plural sentiments (News) | Portfolio weights rebalancing |
| [5] | TD3 | Shares quantity | Limit order book snapshots, News, Synthetic social media posts | Share quantity rebalancing, Weighted signed scoring of plural sentiments |
| [6] | PPO | Cash balance, Shares quantity, Net value | Price data, Volume, Net value, Sentiments (external tool) [Last 10 trajectories] | Trading signal |
| [7] | PPO | - | Price data, adjusted close price, MACD, MACD signal line, MACD histogram, CCI, RSI ADX, News headlines embedding | Share quantity rebalancing |
| [8] | - | Cash balance and other account data | Interstice relation of price data, News headlines | Strategy selection |
| [9] | Classical direct policy gradient algorithm | - | State Representation using: Price data, c, Technical indicators ($z_{open}$, $z_{high}$, $z_{close}$, $z_{adjclose}$, $z_{d5}$, $z_{d10}$, $z_{d15}$, $z_{d15}$, $z_{d20}$, $z_{d20}$, $z_{d25}$, $z_{d30}$), News headlines [Last state] | Portfolio weights rebalancing (including cash) |
| [10] | Modified DPG/policy gradient | - | Sector Features obtained from: Price data, Volume, trading amount, turnover rate, rise and fall range, rolling price/earnings ratio, price/book ratio, rolling price/sales ratio, and rolling price/cash ratio [30 day window] | Sector trend prediction |
| | Classical stochastic policy gradient | - | Predicted Sector trend, Distinct signed scoring of sentiment (News), Stocks relationships within sector [last state] | Portfolio weights Allocation |
| [11] | PPO | - | Technical indicators (MACD, RSI, Bollinger Bands, ADX, VWMA, ATR), Distinct signed scoring of plural sentiments (News headlines, Financial Articles, Social media posts) [Last state] | Weights of RAG knowledge source summing to one, Trading signals dervien by Sentiment score for equally weighted portfolio |
| [12] | PPO | Portfolio Weights | Distinct positive scoring of stock recommendation (News), Distinct positive scoring of risk assessment (News) | Portfolio weights rebalancing adjusted by recommendation score |
| [13] | PPO | - | Price data, Technical indicators, Distinct positive Trading Score(Distinct positive scoring of plural sentiments (news), adjusted close percentages) | - |

etc. The extracted sentiment is then utilized to adjust the reward, action, or guide the policy learning by adjusting the advantage function, which is part of the PPO objective function. Few studies have made adjustments to state representation, either by learning the state representation or by rule-based modifications.

One example of basic flow is [1], where extracted sentiment integrated into the algorithm's state observations, action adjustments, and reward calculations. The study in [1] presents two setting of problem that single stock trading (SST) and portfolio trading. A modification on reward represented at Section IV-A1 was done adding sentiment-based reward to encourage trades that follow market sentiment alongside with volatility adjustment which reduces sentiment-based reward when prices are unstable. One additional component of reward is added for portfolio settings which is portfolio net worth change. As the name suggest it calculated by summing the value of all stocks held and cash balance. The influence on action is done with simple calculation, modifying trade proportion (balance/share) by 0.1 of sentiment score. Table I shows key DRL components for portfolio optimization.

Another study by [3] following the basic flow has applied LLaMA 3.3 trained on an extensive collection of earnings reports, market commentaries, and analyst insights, enabling it to differentiate between neutral reporting, speculative viewpoints, and sentiment-influenced market trends. This model has been fine-tuned for financial text analysis to extract daily financial news sentiment. Market sentiment has been part of state representation and the advantage function of PPO model. The study has illustrated that integrating market sentiment can enhance adaptability and robustness in portfolio optimization.

The study in [13] introduces a Market-Cap Stratified Subset (MCSS) strategy that is which groups stocks by market capitalization into small, mid and big. Allowing different impact of sentiment score on different groups, due to different sensitivity to sentiment for each group, especially as smaller stocks tend to react more strongly. In addition, they introduced Market-Aware Module that modify the observed state in rule-based way. It checks whether short-term price movements align or conflict with sentiment signals producing trading score which fed later into the RL model.

The study in [12] uses the risk assessment scores generated by the LLMs like DeepSeek V3, Qwen 2.5, and Llama 3.3, to adjust the trajectory returns in the Conditional Value-at-Risk (CVaR) extended objective formalized by [14]. The CVaR objective is a risk-sensitive constrained optimization problem

in reinforcement learning, aiming to maximize expected return while penalizing high-loss trajectories by keeping the policy's CVaR below a specified threshold. CVaR itself is a risk measure that estimates the expected loss of a portfolio given that the loss is greater than the Value-at-Risk (VaR) threshold. The importance of CVaR is that it goes further than VaR by calculating the average loss in the worst $\alpha$ portion of outcomes.

In addition to risk-sensitivity contribution, [12] had contributed by modifying the model action with generated trade recommendation signals from financial news using the same LLM models.

The study [8] suggests a method for dynamic strategy switching based on large language models, GPT-4 in particular, to adapt market changes. [8] has utilized a financial concept named asset pricing model that is a framework that links asset risk to expected return. The model have been formulated as in Eq. (8).

$$\hat{R}_t = \alpha_t + f_t \cdot \beta_t \tag{8}$$

Where $\alpha_t$ is the Local Data Model which mines the intrinsic information specifically their linear relationship. While $\beta_t$ represents yields of Global Algorithm Trade Model which select the corresponding trading strategy. $f_t$ component represents the strategy selection made by the algorithm model at time t. $\hat{R}_t$ is the final return of a stock at time t.

The model $f_t$ applies reinforcement learning principles on LLM prompting by processing historical data features, executing the selected strategy, calculating ranking reward as explained in section IV-A5, and iterating after each strategy selection action, thereby enabling adaptive multi-strategy switching in environments where conventional RL methods would demand extensive retraining and hyperparameter tuning across different datasets. To achieve this, the prompt is structured into modular components that are role definition, Few-Shot examples, strategy description, and output formulation via the Chain-of-Thought approach, guiding the LLM to reason analytically through data trend analysis, strategy selection rationale, and strategy critique, while also outputting the chosen strategy in JSON format.

The study [6] proposed an advanced approach using an LLM-based actor–critic architecture. In their design, the layers of the SmolLM2-135M-Instruct model were divided into frozen and trainable layers. The trainable layers were optimized through reinforcement learning fine-tuning, where reward signals guided their learning. This setup allows the model to retain broad language comprehension while efficiently adapting to financial decision-making tasks with low computational cost.

Additionally, a policy MLP network and a value MLP network were incorporated, both leveraging the output of the trainable layers for domain-specific learning. Unlike conventional integrations of RL with LLM, the model in [6] was directly prompted to generate trading actions.

The study [7] have proposed using Partially Observable Markov Decision Process (POMDP) with stock trading. Which 2 additional elements above regular MDO, that are observation ($\Omega$) and conditional observation probability (O). In POMDP, the agent receives an observation with o $\in \Omega$ that depends on $s_{t+1}$ and $a_t$ , with probability O($o_t \mid s_{t+1}, a_t$). In this paper [7] FinBERT was used to extract embedding mean from news headlines combined with market data creating an observation vector. A PPO architecture were modified to contain LSTM layers alongside the casual MLP layers for both actor and critic heads.

A study that focus on sector rotation and multi-RL was proposed by [10]. Sector rotation is a macro-level portfolio strategy, where investment shifts between sectors and capital allocation is managed across sector distributions rather than individual stocks. The approach employs a two-layer Markov Decision Process (MDP) framework. The first layer uses a multi-agent reinforcement learning model to capture sector trend dynamics from time-series data that taken from raw stock data and transformed to sector data. The second layer focuses on portfolio management, integrating multiple sources such as time-series data for all stocks, outputs from the first layer, sentiment insights from news analyzed with large language models, and sector-level stock relationships extracted through graph convolution. By fusing these diverse inputs, the model produces the final portfolio decisions.

The study [2] introduced an adaptive and explainable framework that integrates Large Language Models (LLMs) with Reinforcement Learning (RL) to enable dynamic long–short position adjustments in response to evolving market conditions. This approach overcomes the limitations of traditional margin-trading systems, in which the long–short ratio is predetermined and remains static throughout the trading period, by allowing periodic and adaptive reallocation. The proposed framework comprises two primary components. The first is the Explainable Market Forecasting/Reasoning Pipeline, which employs LLMs to determine optimal adjustment ratios every k steps. The LLM utilizes two specialized prompt pipelines, one tailored for macroeconomic indicator time Series and another for firm-specific news. The second component is the Portfolio Reallocation Stage, which employs a pre-trained RL agent to rebalance the portfolio by reallocating funds between long and short positions in order to achieve the updated ratio. After the reallocation, the RL agent continues its trading operations using the adjusted portfolio for the next k steps. In addition to predicting adjustment ratios and near-term trends, this approach offers explicit reasoning and clear insights that improve portfolio decision-making.

The study [11] proposed a novel framework that builds on the capabilities of LLaMA 2 and enhances it through an advanced fine-tuning approach called Retrieval-Augmented Generation (RAG). This method integrates multiple knowledge sources to enrich model outputs with current, domain-specific information, thereby reducing hallucinations and improving accuracy. Furthermore, the language model was refined using reinforcement learning (RL) feedback, where the LLM's generated sentiment scores were evaluated against market returns. These evaluations were incorporated into a reward function within a PPO-based RL structure, guiding the model's optimization and enabling adaptive weighting across the K knowledge sources in the RAG module.

The study in [5] presents an attempt that the first of its kind based on researcher believes to asses market manipulation, pump-and-dump schemes specifically, through LLM-generated

text within a realistic financial market simulation. They have shown how the RL agent learns to enhance its reward by deliberately influencing market sentiment by its posts. The experiment have been conducted in an offline simulated environment to ensure ethical integrity. The isolated offline feed is mainly generated by replaying historical order data to be feed to Analyst prompt posts and Trader prompt posts. To model how traders respond to language-based manipulation, a sentiment-driven trading agent is introduced that analyze the sentiment of social media feed using RoBERTa model and computes weighted sentiment score as its trading indicator. In the other hand, RL trader is used to influence on the sentiment-based agent. While it takes order streams to make its trading action and sentiment action. It optionally generate social media posts reflecting its sentiments which then are visible to the sentiment agent. The RL agent structure is based on Twin Delayed Deep Deterministic Policy Gradient (TD3) that employ three fully connected layers with normalization, and the actor uses an LSTM-based embedding of sequential observations.

### D. Evaluation Metrics

A robust evaluation of quantitative trading strategies requires the use of diverse metrics spanning profitability, risk, and risk-adjusted returns. These metrics are crucial for assessing performance and ensuring that algorithms meet investment objectives, which typically center on maximizing return while maintaining portfolio stability. The metrics utilized across the papers can be generally categorized into Risk-Adjusted Performance, Return, Risk and Volatility, and other specialized metrics. Evaluation often prioritizes Cumulative Return (CR) and Sharpe Ratio (SR) for assessing long-term gains and risk-adjusted returns.

*1) Risk-adjusted performance metrics:* It provide a more comprehensive assessment of a strategy's efficacy than absolute returns by relating the gains achieved to the level of risk undertaken. The Sharpe Ratio (SR) is the most frequently cited metric across the papers, appearing in nine papers. It is a critical financial metric that assesses the risk-adjusted return of an investment or portfolio. SR has been explained in Eq. (3).

*2) Return metrics:* It quantify the absolute profitability or monetary gain generated by a trading strategy over a specific period. Cumulative Return (CR) is a key performance indicator that measures the total value change of an investment over time. It is typically expressed as the overall percentage gain or loss of a portfolio over a specified period. Eq. (9) show how to measure CR.

$$CR = \frac{P_{end} - P_0}{P_0} \tag{9}$$

where $P$ represents the portfolio value.

The Annualized Rate of Return (ARR), or Annualized Return (AR), represents the average rate of return for a managed portfolio over a full year of trading days. It is derived from the daily returns average over the investment period, adjusted to a yearly scale based on 252 trading days. Eq. (10) show how to measure ARR.

$$ARR = \frac{\text{Total net profit/Number of years}}{\text{Initial investment}} \tag{10}$$

*3) Risk and volatility metrics:* are essential for quantifying the instability and maximum potential downside of a trading strategy. Maximum Drawdown (MDD), sometimes referred to as Maximum Pullback (MPB), quantifies the largest single drop in an asset's value from its highest peak to its lowest trough over a specific period. It serves as a crucial indicator of portfolio risk. Eq. (11) show how to measure MDD.

$$MDD = \max\left(\frac{V_{\text{peak}} - V_{\text{trough}}}{V_{\text{peak}}}\right) \tag{11}$$

Annualized Volatility (AV) quantifies the return fluctuations of a portfolio. Eq. (12) show how to measure AV.

$$AV = DV \times \sqrt{252}. \tag{12}$$

where $DV$ is standard deviation of daily returns adjusted to an annual scale by the square root of 252.

## V. Findings and Discussion

This section synthesizes insights from the reviewed studies, emphasizing how Deep Reinforcement Learning (DRL) and Large Language Models (LLMs) contribute to portfolio optimization.

### A. Algorithmic Trends and Performance

Across the 12 studies reviewed, Proximal Policy Optimization (PPO) emerged as the most frequently used algorithm due to its training stability and effectiveness in continuous control problems. PPO-based models demonstrated robust adaptability to market fluctuations and achieved high risk-adjusted returns, with Sharpe Ratios (SR) ranging from 1.46 to 3.34 across various datasets. TD3, modified DPG/policy gradient, and hybrid actor–critic methods were less common but effective in handling continuous action spaces and noisy financial environments.

### B. Reward Function Design

Reward function formulation remains a decisive factor influencing portfolio behavior and risk sensitivity. Studies using risk-adjusted metrics, such as the Sharpe Ratio or Conditional Value-at-Risk (CVaR), encouraged smoother trading behavior and improved robustness in volatile markets. In contrast, heuristic and trend-based rewards. On the other hand, Sector-level or ranking rewards extend learning to macro or multi-strategy settings but increase model complexity. Overall, effective reward design depends on balancing interpretability, stability, and economic realism while mitigating overfitting to short-term or sentiment-driven fluctuations.

TABLE II. EXPERIMENTAL RESULTS OF REVIEWED STUDIES, SHOWING THEIR BEST EXPERIMENT RESULT

| Paper | Experimental Setup | Index / Fund | Date | Period | ↑ SR | ↑ CR | ↑ ARR | ↓ MDD | ↓ AV |
|---|---|---|---|---|---|---|---|---|---|
| [1] | GPT | LEXCX | 16/11/2023 to 10/11/2024 | 1 Year | - | 0.4201 | - | - | - |
| [2] | GPT-4o; Firm news; Takes 20% of trend prediction | DJIA | 5/2020 to 4/2024 | 4 Years | 1.314 | 1.508 | 2.4385 | - | - |
| [3] | LLaMA 3.3 | GOOGL, MSFT, META | 1/2013 to 1/2020 | 7 Years | 1.90 | - | 0.302 | -0.138 | 0.112 |
| [6] | SmolLM2-135M-Instruct | JNJ | 1/7/2020 to 6/5/2021 | 1.2 Year | 3.344 | 33.724 | - | 9.320 | 17.174 |
| [7] | FinBERT | 30 of DJIA, GOOGL, NVDA, AMZN | 1/1/2016 to 24/1/2020 | 4 Years | 1.46 | 1.3439 | - | - | - |
| [8] | GPT-4 | NASDAQ | 12/2010 to 1/2022 | 12 Years | 1.482 | - | 0.136 | -0.06 | 0.092 |
| [9] | FinGPT | DJIA | 1/2000 to 12/2022 | 22 Years | - | - | 33.45 | 25.33 | 0.234 |
| [10] | gpt-4o-turbo | NASDAQ, NYSE | 2011 to 2023 | 13 Years | 1.744 | - | 0.25 | - | 1.084 |
| [11] | LLaMA 2; Bullish market | S&P 500 | 1/1/2021 to 31/12/2021 | 1 Year | 2.3557 | 0.3393 | 0.3450 | 0.1464 | -0.0781 |
| [12] | DeepSeek V3; 10% LLM influence | NASDAQ-100 | 2013 to 2023 | 10 Years | - | 2.1 | - | - | - |
| [13] | DeepSeek-R1 7B | S&P 500 | 1999 to 2023 | 24 Years | 3.1 | - | - | - | - |

### C. Integration of Large Language Models

LLMs play a complementary yet transformative role in portfolio optimization. Their integration primarily occurs in multiple forms. One is sentiment Integration by extracting sentiment from news or social media to augment state representation, guide reward shaping, or produce direct trading actions. Another form of integration is using LLM within actor-critic networks. Studies show that incorporating LLM-derived signals improves adaptability and interpretability. For instance, [3] and [12] demonstrated how LLM-based sentiment and risk assessments enhance DRL's responsiveness to macroeconomic shifts. However, over-dependence on textual inputs can introduce bias and instability, especially when news sources are inconsistent or temporally lagged.

### D. Data Sources and Environment Complexity

The reviewed studies use diverse data sources, including price data, technical indicators, macroeconomic indicators, and unstructured text from financial news. Combining structured (numerical) and unstructured (textual) data consistently improved model generalization and return stability. For example, models combining price data and sentiment, such in [11] study, achieved higher SR and lower maximum drawdown (MDD). Nevertheless, few studies explicitly address non-stationarity or market regime shifts, which remain critical challenges for DRL-based approaches.

### E. Comparative Results

Table II summarizes the quantitative results across studies. Sharpe Ratios ranged between 0.42 and 3.34, and Cumulative Returns (CR) reached up to 33.7%, indicating substantial improvements compared to traditional benchmarks. A study that integrates LLMs with PPO as its network achieved the highest SR values, meaning employing LLM as a core component ensures an accurate understanding of market dynamics and sentiments, which ultimately enhances performance, regardless of the fact that the LLM model utilized, SmolLM2, is a lightweight LLM model compared to others. In other hand, DeepSeek-R1 demonstrated one of the strongest results 3.1 in SR. Both SmolLM2 and DeepSeek-R1 underscore the benefits of fine-tuned, lightweight LLMs in trading contexts.

### F. Challenges and Limitations

Despite notable advances, several challenges persist. One is Reward design sensitivity; minor adjustments can destabilize learning, especially under sparse or delayed rewards. Another is Computational cost; combining DRL with LLM fine-tuning increases training overhead. Also, Overfitting and limited generalization, many models are trained and tested on specific indices (e.g., NASDAQ, S&P 500) or narrow time horizons, making them vulnerable to overfitting. Explainability and interpretability, although LLMs improve textual reasoning, both DRL and LLM components remain largely black-box systems.

## VI. CONCLUSION

This survey demonstrated the transformative role of Deep Reinforcement Learning (DRL), particularly in conjunction with Large Language Models (LLMs), in addressing the challenges of dynamic portfolio optimization. PPO emerged as the most widely adopted and stable DRL algorithm, proving effective in continuous control environments and achieving robust, high risk-adjusted returns (Sharpe Ratios up to 3.34). The design of the reward function is a critical factor, with risk-adjusted metrics like the Sharpe Ratio and Conditional Value-at-Risk (CVaR) encouraging smoother and more stable portfolio behavior in volatile markets. The integration of LLMs significantly enhances these models by extracting market sentiment, risk assessments, and trading insights from unstructured text data, thereby improving adaptability and interpretability. Combining structured (price and technical indicators) and unstructured (textual) data sources consistently leads to better model generalization and return stability. Nonetheless, the field still contends with significant challenges: the sensitivity of reward function design, high computational costs associated with fine-tuning LLMs alongside DRL, the risk of overfitting to narrow time horizons, and the persistent issue of limited explainability in these black-box systems. Future research must

prioritize solutions for non-stationarity and market regime shifts to enhance the long-term competitive advantage of DRL-LLM approaches in real-world financial markets

## REFERENCES

[1] A. Unnikrishnan, "Financial news-driven llm reinforcement learning for portfolio management," *arXiv preprint arXiv:2411.11059*, 2024.

[2] J. Gu, J. Ye, G. Wang, and W. Yin, "Adaptive and explainable margin trading via large language models on portfolio management," in *Proceedings of the 5th ACM International Conference on AI in Finance*, 2024, pp. 248–256.

[3] K. Kirtac and G. Germano, "Leveraging llm-based sentiment analysis for portfolio allocation with proximal policy optimization," in *ICLR 2025 Workshop on Machine Learning Multiscale Processes*.

[4] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan *et al.*, "The prisma 2020 statement: an updated guideline for reporting systematic reviews," *bmj*, vol. 372, 2021.

[5] D. Byrd, "Exploring sentiment manipulation by llm-enabled intelligent trading agents," *arXiv preprint arXiv:2502.16343*, 2025.

[6] G. Xiong, Z. Deng, K. Wang, Y. Cao, H. Li, Y. Yu, X. Peng, M. Lin, K. E. Smith, X.-Y. Liu *et al.*, "Flag-trader: Fusion llm-agent with gradient-based reinforcement learning for financial trading," *arXiv preprint arXiv:2502.11433*, 2025.

[7] M. Veisi, S. Berangi, M. S. Khojasteh, and A. Salimi-Badr, "A deep reinforcement learning approach combining technical and fundamental analyses with a large language model for stock trading," in *2024 14th International Conference on Computer and Knowledge Engineering (ICCKE)*. IEEE, 2024, pp. 224–229.

[8] X. Zhong, Z. Zhao, B. Yan, and Q. Xu, "Large language model for dynamic strategy interchange in financial markets," in *2024 9th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*. IEEE, 2024, pp. 306–312.

[9] W. Ding, Z. Chen, H. Jiang, Y. Lin, and F. Lin, "Trend-heuristic reinforcement learning framework for news-oriented stock portfolio management," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 5120–5124.

[10] Y. Yan, C. Zhang, Y. An, and B. Zhang, "A deep-reinforcement-learning-based multi-source information fusion portfolio management approach via sector rotation," *Electronics*, vol. 14, no. 5, p. 1036, 2025.

[11] Z. Zhao and R. E. Welsch, "Aligning llms with human instructions and stock market feedback in financial sentiment analysis," *arXiv preprint arXiv:2410.14926*, 2024.

[12] M. Benhenda, "Finrl-deepseek: Llm-infused risk-sensitive reinforcement learning for trading agents," *arXiv preprint arXiv:2502.07393*, 2025.

[13] S. Arshad, H. Ameer, N. Azhar, and S. Latif, "Finrl contest 2025 task 1: Market-aware in-context learning framework for proximal policy optimization in stock trading using deepseek," in *2025 IEEE 11th International Conference on Intelligent Data and Security (IDS)*. IEEE Computer Society, 2025, pp. 76–78.

[14] C. Ying, X. Zhou, H. Su, D. Yan, N. Chen, and J. Zhu, "Towards safe reinforcement learning via constraining conditional value-at-risk," *arXiv preprint arXiv:2206.04436*, 2022.