algorithms

Article

A Systematic Approach to Portfolio Optimization: A Comparative Study of Reinforcement Learning Agents, Market Signals, and Investment Horizons

Francisco Espiga-Fernández *, Álvaro García-Sánchez † and Joaquín Ordieres-Meré †

Industrial Engineering, Business Administration and Statistics Department, Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid, José Gutierrez Abascal 2, 28006 Madrid, Spain; alvaro.garcia@upm.es (Á.G.-S.); j.ordieres@upm.es (J.O.-M.)

* Correspondence: francisco.espiga.fernandez@alumnos.upm.es

† These authors contributed equally to this work.

Abstract: This paper presents a systematic exploration of deep reinforcement learning (RL) for portfolio optimization and compares various agent architectures, such as the DQN, DDPG, PPO, and SAC. We evaluate these agents' performance across multiple market signals, including OHLC price data and technical indicators, while incorporating different rebalancing frequencies and historical window lengths. This study uses six major financial indices and a risk-free asset as the core instruments. Our results show that CNN-based feature extractors, particularly with longer lookback periods, significantly outperform MLP models, providing superior risk-adjusted returns. DQN and DDPG agents consistently surpass market benchmarks, such as the S&P 500, in annualized returns. However, continuous rebalancing leads to higher transaction costs and slippage, making periodic rebalancing a more efficient approach to managing risk. This research offers valuable insights into the adaptability of RL agents to dynamic market conditions, proposing a robust framework for future advancements in financial machine learning.

Keywords: reinforcement learning; portfolio optimization; policy optimization; deep learning; deep reinforcement learning; mean-variance optimization

Citation: Espiga-Fernández, F.; García-Sánchez, Á.; Ordieres-Meré, J. A Systematic Approach to Portfolio Optimization: A Comparative Study of Reinforcement Learning Agents, Market Signals, and Investment Horizons. *Algorithms* **2024**, *17*, 570. https://doi.org/10.3390/a17120570

Academic Editor: Ulrich Kerzel

Received: 29 October 2024
Revised: 8 December 2024
Accepted: 9 December 2024
Published: 12 December 2024

1. Introduction

1.1. Overview of Portfolio Optimization

Portfolio optimization refers to the systematic selection of a combination of investment assets, or a portfolio, with the goal of maximizing returns while managing risk. The optimization process typically aims to find an allocation of assets that achieves an investor's objectives, such as maximizing return for a given level of risk or minimizing risk for a given expected return. This balancing act is commonly framed through the lens of Modern Portfolio Theory (MPT) [1], which models risk through the variance or covariance of asset returns. In practice, portfolio optimization applies to various investment horizons and strategies, involving the dynamic reallocation of assets over time in response to market changes.

The assets or investment vehicles within a portfolio can include a broad range of financial instruments. These include equities (stocks), bonds, commodities, currencies, and derivatives such as options and futures. Each asset class behaves differently in terms of risk and return, making it important for portfolio managers to consider the interrelationships between asset returns when optimizing the portfolio. Investment vehicles vary in their functions: equities are generally employed for long-term capital appreciation, whereas bonds offer more predictable income streams and reduced risk exposure.

One of the central challenges in portfolio optimization is handling transaction costs and slippage, both of which can significantly impact the returns of a portfolio. Transaction costs include brokerage fees, taxes, and other costs incurred during the buying and selling of assets. Meanwhile, slippage refers to the difference between the expected price of a trade and the actual price, which often occurs in markets with low liquidity or during periods of high volatility. These factors complicate the portfolio optimization process because frequent trading, which becomes essential in rapidly changing market environments, can erode potential returns through these hidden costs. Additionally, the need to rebalance portfolios dynamically in response to market conditions adds further complexity.

Portfolio optimization techniques are widely applied across the financial industry, particularly among large institutional players such as hedge funds, mutual funds, pension funds, and investment banks. These institutions use sophisticated quantitative models to manage the portfolios of high-net-worth individuals and institutional investors, with the objective of exceeding market benchmarks while maintaining robust risk management frameworks [2,3].

1.2. Limitations of Traditional Methods

Traditional methods for portfolio optimization, such as mean-variance optimization (MVO) [4], risk parity, and Value at Risk (VaR), have long been central to the field of asset management. Originally developed within the framework of MPT, MVO aims to balance expected returns and risk and to construct an optimal portfolio by balancing expected returns and risks. The core objective of MVO is to minimize portfolio variance, given a target return, or equivalently, maximize return for a given level of risk, represented by variance or the standard deviation. The optimization problem can be expressed as

$$\min_{w} w^T \Sigma w \quad \text{subject to} \quad w^T \mu \geq r_{min}, \quad \sum w_i = 1, \quad 0 \leq w_i \leq 1 \tag{1}$$

where w represents portfolio weights, Σ is the covariance matrix of asset returns, and μ is the vector of expected returns [2]. Risk parity, on the other hand, aims to allocate assets such that each contributes equally to the overall portfolio risk. This method focuses on diversifying risk rather than returns, making it attractive for risk-averse investors. Value at Risk (VaR) is a widely used risk measure that estimates the potential loss in portfolio value over a specific time frame with a given confidence level. VaR is often used to determine the maximum expected loss under normal market conditions, though its reliance on historical data makes it vulnerable to extreme events [3].

Despite their widespread adoption, these traditional methods face significant limitations, particularly in the context of modern financial markets. One key issue is their lack of adaptability to rapidly changing market conditions. Methods like MVO rely on historical data and assume that future asset returns and risks will behave similarly, which is frequently invalid in volatile or non-stationary markets. This reliance on historical averages can result in overfitting, where the model becomes too tailored to past performance, leading to suboptimal decisions when market dynamics shift [5]. Moreover, MVO's assumption that risk can be fully captured by variance oversimplifies the complex nature of risk in financial markets, ignoring factors such as tail risk or extreme market movements.

Another limitation of these traditional approaches is their reliance on strict assumptions about asset returns. MVO, for instance, assumes that asset returns follow a normal distribution and that correlations between assets are stable over time. In reality, financial returns tend to be skewed and exhibit fat tails, with extreme events occurring more frequently than predicted by normal distribution models [6]. Similarly, VaR assumes a linear relationship between risk factors and portfolio returns, which breaks down during periods of market stress, leading to an underestimation of risk. Furthermore, risk parity can be overly conservative in environments where certain assets exhibit low volatility but are exposed to systemic risks, thus potentially skewing allocations [7].

While this study focuses on RL methods for portfolio optimization, it is important to acknowledge the role and advances of traditional machine learning (ML) techniques in this domain. ML methods have been widely recognized for their ability to improve over classical approaches such as MVO by leveraging historical data and advanced predictive modeling. In [8], it was demonstrated that Long Short-Term Memory (LSTM) [9]-based portfolio optimization surpasses traditional MVO in cumulative returns and Sharpe ratios, while [10] showed the advantages of autoencoder-based models in annualized returns over MVO in specific scenarios. Similarly, advanced linear programming models [11] and robust optimization techniques [12] have been shown to achieve superior diversification, computational efficiency, and stability compared to traditional methods.

However, the limitations of ML methods in dynamic and adaptive environments have necessitated the adoption of RL frameworks, which incorporate the idea of environmental signals and continuous learning. RL has consistently demonstrated superior performance compared to traditional ML models by adapting dynamically to market changes and learning decision-making policies directly from data. The authors of [13] showed that RL achieves Sharpe ratios and cumulative returns that are higher than those obtained with minimum-variance strategies. Other studies [14,15] highlighted the ability of RL models to leverage technical indicators and manage risk more effectively than ML techniques. Furthermore, algorithms such as deep deterministic policy gradients (DDPGs) [16] and distributional RL [17] have showcased the resilience and risk-sensitive capabilities of RL in volatile financial markets.

This study prioritizes RL methods over traditional ML techniques and classical portfolio optimization due to their capacity for dynamic decision-making and adaptability, which are critical for addressing the complexities of real-world financial markets. Although ML methods improve significantly over classical approaches by leveraging predictive modeling and data-driven strategies, they lack the flexibility required to adapt dynamically to market changes. In contrast, RL and deep reinforcement learning (DRL) approaches do not rely on static assumptions or fixed models but instead learn optimal policies directly from data. This ability to adjust to uncertain and volatile market conditions makes RL a more robust and flexible framework for managing portfolios and addressing complex risk–return profiles effectively [2].

1.3. Reinforcement Learning for Portfolio Optimization

In the context of portfolio optimization, RL has emerged as a promising alternative to traditional optimization methods by allowing dynamic asset allocation based on market conditions. Unlike traditional approaches that assume static models or rely on historical data, RL adapts dynamically to evolving market conditions by adjusting asset weights in response to volatility, correlation shifts, and other real-time market signals [2,5]. In RL-based portfolio optimization, the agent typically learns to allocate weights to different assets in a way that optimizes a predefined objective, such as maximizing the cumulative returns or minimizing risk-adjusted returns over a specified investment horizon. The reward function, which guides the agent's learning, can be designed to reflect various financial performance measures, such as the Sharpe or Sortino ratios [18]. For a detailed discussion on RL methodologies and their applications in portfolio optimization, refer to Section 2.3 and Appendix A.

Aside from RL, there are alternative formulations for the portfolio optimization problem, such as Genetic Algorithms (GAs) and Quadratic Programming (QP), both of which are commonly found in financial optimization. Genetic Algorithms, inspired by natural selection, have been applied to portfolio optimization by evolving portfolios over time to maximize a fitness function [19,20]. These algorithms are advantageous in handling non-linear optimization problems but may suffer from slow convergence.

Quadratic Programming [4] is another well-known method, as was previously mentioned in the context of mean-variance optimization, where the problem is solved as a constrained optimization of quadratic objective functions. While QP offers efficient solutions for convex problems, it is limited by its inability to handle the complexities and non-linearities of real-world financial markets, where assumptions about normal distributions and fixed covariances often do not hold. These traditional methods, although effective in some scenarios, lack the adaptability and dynamic decision-making capabilities that RL introduces into the portfolio management space.

1.4. Related Works, Research Gaps, and Questions

1.4.1. Related Works

Recent research on RL in portfolio optimization has employed a variety of data sources. The majority of studies use OHLC (Open, High, Low, Close) price data, augmented by technical indicators such as moving averages and volume indicators [21]. Certain studies incorporate macroeconomic factors and volatility measures to capture broader market sentiment [13]. Moreover, feature extractors like multi-layer perceptrons (MLPs), Convolutional Neural Networks (CNNs) [22], and LSTM networks have been used to detect patterns in these time-series data, enabling RL agents to make informed decisions about market conditions [5]. The choice of feature extractor plays a critical role in shaping the agent's capacity to detect trends and forecast market behavior [13]. Studies have explored a wide range of investment instruments, including equity indices, cryptocurrencies, and bonds, with many focusing on highly liquid assets to guarantee reliable historical data for training RL agents [13,23].

RL agents such as Deep Q-Networks (DQNs) [24], Deep Deterministic Policy Gradient (DDPG) [25], Proximal Policy Optimization (PPO) [26], and Soft Actor–Critic (SAC) [27] have been applied to portfolio optimization [5,13,23,28].

These agents vary in their approach to handling decision-making under uncertainty, with some, such as DDPG, being particularly useful for continuous action spaces like portfolio weight adjustments. Rebalancing strategies have also been studied, ranging from continuous rebalancing to periodic intervals [29]. This plays a crucial role in determining the costs associated with transaction fees and slippage, factors that are critical in real-world implementations.

The reward function typically used reflects cumulative returns or risk-adjusted measures like the Sharpe ratio, but other metrics, such as maximum drawdown, have also been considered [30].

1.4.2. Research Gap

Despite the substantial advancements in applying RL to portfolio optimization, A significant gap persists in systematically comparing different RL configurations. The existing body of research tends to focus on specific agent architectures or a narrow range of investment instruments, leading to heterogeneity in the methodologies and results.

Furthermore, existing studies often do not account for the transaction costs and slippage that affect real-world trading. For example, there has been little exploration into the comparative impact of using continuous rebalancing versus less frequent rebalancing on agent performance, particularly in the context of transaction costs and cumulative reward [21,29]. Additionally, the lack of standardized investment instruments across studies limits the ability to systematically compare the results. While some studies employ indices or ETFs, others focus on more volatile assets like cryptocurrencies, making it challenging to evaluate the consistency of RL approaches in varying asset classes.

While RL agents like DDPG and SAC have been successfully applied, there is limited research on the influence of varying environmental signals, reward structures, and feature extractors across diverse financial assets [13,31]. The heterogeneity in data sources, agent configurations, and feature extraction methods makes it difficult to draw generalizable conclusions about the best approaches for portfolio optimization under various market conditions. To date, no study has conducted an exhaustive analysis of how different rebalancing frequencies, investment instruments, feature extractors, and reward functions affect portfolio optimization performance under RL. As a result, there is a gap in understanding the trade-offs involved in choosing one configuration over another in varying market conditions [32].

1.4.3. Research Questions

This study aimed to address these research gaps by posing several key questions. First, it explored the impact of continuous versus periodic rebalancing on portfolio performance. This included analyzing the effects of transaction costs and slippage, as well as how different rebalancing frequencies affect the agent's ability to learn optimal policies. Second, the study investigated how different feature extractors, such as CNNs versus MLPs, influence the agent's understanding of the market environment and its ability to make accurate predictions about future asset prices.

Furthermore, we examined the use of technical indicators compared to OHLC price data as inputs to the RL agent. According to the Efficient Market Hypothesis (EMH) [33], price alone should reflect all available information, but the use of additional indicators may enhance the agent's decision-making in practice [7]. Finally, we sought to determine which RL algorithm—DQN, DDPG, PPO, or SAC—and agent nature—value optimization or policy optimization—are most suitable for portfolio optimization, especially when considering different history windows and their impacts on both performance and convergence. This included evaluating the effectiveness of agents in handling a fixed set of instruments: index funds or ETFs, which track well-known market benchmarks and are widely used in the investment community [2].

2. Materials and Methods

2.1. Research Methodology

To establish a comprehensive RL algorithm framework and explore the available information from the environment, several configurations were tested, as explained in Section 2.7. This approach allowed for a ceteris paribus analysis, where the remaining variables are held constant to isolate the impact of different agent configurations or environmental factors on the reward. By keeping key factors, such as the training and evaluation data, constant across all experiments, we ensured a fair comparison of the agents. Additionally, the performance of each configuration was compared against a market benchmark and a simple equally weighted portfolio, as detailed in Section 2.7.6, to measure the relative advantage of using RL for the portfolio optimization problem. This structure ensures that the observed differences in outcomes are attributable solely to the variations in RL algorithms, feature extractors, or environmental inputs.

As for the impact of slippage and rebalancing frequency, two experimental setups were used: one with continuous rebalancing, where the portfolio is adjusted every day, and another with rebalancing every 10 periods (approximately bi-weekly). These setups were designed to evaluate the trade-off between more frequent rebalancing, which can capture market opportunities more effectively, and the higher transaction costs and slippage incurred from increased trading activity. More frequent rebalancing not only raises transaction costs but also exposes the portfolio to slippage, particularly from bid–ask spreads and the potential difference between market close prices and the next day's open. The choice of rebalancing every 10 periods was made as a compromise between daily and more widely used industry practices, such as monthly rebalancing, which tends to take a longer-term view. Additionally, this frequency ensures sufficient overlap between consecutive observations in the environment, which provides historical data with lookback periods of 16 and 28 periods, making it more likely that both rebalancing configurations will be similarly impacted by any potential correlations in market behavior. The goal is to analyze whether the gains in returns from more frequent rebalancing can outweigh the additional trading costs, providing insight into the optimal rebalancing frequency for maximizing performance in real-world scenarios [34].

With respect to the value of technical indicators versus raw prices, two different signal configurations were tested, as detailed in Section 2.4. By maintaining all other parameters constant between experiments, we aimed to isolate and assess the additional value provided by technical indicators. On the one hand, as neural networks are powerful universal function approximators [35] and feature learners [36,37], it could be argued that the raw price data alone should be sufficient for the agent to learn meaningful patterns, and the technical indicators could be learned as latent-state representations. However, given the limited amount of data available, technical indicators may offer a concise and informative summary of the market environment, potentially enhancing learning efficiency and convergence. This is particularly relevant in light of the EMH [7,33], which posits that all available information is reflected in the price. Some studies in the literature suggest that additional market indicators may contradict the EMH, providing opportunities for better decision-making by RL agents. Therefore, the experiment sought to explore the balance between raw data and engineered features in portfolio optimization.

With respect to capturing patterns and anticipating behavior in the time series of the indicators for each financial instrument, two different feature extractors—MLP and CNN—were employed. MLPs are effective at capturing general relationships across all input features but might miss localized temporal patterns, while CNNs excel at identifying local structures and sequential dependencies in time-series data, making them well suited for detecting patterns in market signals. Additionally, the experiments varied the lookback period, with lengths of 16 and 28 periods, to determine the trade-off between providing the agent with more historical context and the increased complexity of the model. A longer lookback may provide valuable historical context, allowing the agent to capture long-term market trends, but it also increases the size of the input, especially for CNNs, which could lead to more challenging training and potential overfitting. This experiment sought to determine the optimal balance between the depth of historical data and network complexity.

Lastly, in terms of performance, the study compared different RL agent families to determine whether continuous-action agents (e.g., SAC, DDPG) that allocate across multiple instruments at once can outperform discrete-action agents (e.g., DQN, PPO) that allocate the entire portfolio to a single instrument. The hypothesis is that continuous agents, by diversifying across instruments, may be better equipped to handle complex market environments, thus improving risk-adjusted returns. On the other hand, discrete-action agents, which focus on identifying the single instrument with the highest expected value, might excel in simpler, trending markets. By comparing these two approaches, the experiment aimed to evaluate the trade-offs between diversification and targeted allocation in RL-based portfolio optimization.

2.2. Problem Formulation

In this paper, the portfolio optimization problem is framed as an RL task, where the objective is to find an optimal policy that maximizes the cumulative returns of a portfolio over time. The RL environment simulates the financial market and provides observations in the form of environmental signals, which include either OHLC prices or technical indicators, as described in Section 2.4. Depending on the chosen agent configuration, the observations are processed as either vectors or 3-dimensional tensors, as detailed in Section 2.5. This flexibility allows the RL agent to handle different types of input data effectively, depending on the chosen feature extractor (MLP or CNN).

The agent consists of two core components: a feature extractor (e.g., MLP or CNN) to process data and a decision-making algorithm that determines the portfolio weights based on extracted features. The output of the agent is a vector of dimension $1 \times n_{\text{instruments}}$, including the risk-free instrument. For continuous-action agents (SAC, DDPG), the weights are normalized to sum to 1, representing a proportional allocation to each instrument. In contrast, for discrete-action agents (DQN, PPO), the vector contains a single component equal to 1, representing a full allocation to one instrument, while the remaining components are set to 0. The precise architecture and functioning of the agent are further detailed in Appendix A.

In each rebalancing period $t+f$, with f being the rebalancing frequency in trading periods, the agent proposes a new set of portfolio weights, which are combined with the previous portfolio weights decided at time t according to Section 2.2 to compute the return over the rebalancing period. The calculation accounts for slippage and transaction costs, modeled based on weight changes, which decrease the return of bought or sold positions as a percentage. One of the assumptions of our paper is that instruments are liquid and fractional assets, as opposed to discrete units (shares) priced at P_i. Specifically, the agent's proposal results in three subsets:

- H (held positions): Assets where the portfolio weight remains unchanged $w_{i,t} = w_{i,t+f}$.
- B (bought positions): Assets where the portfolio weight has increased, requiring the purchase of additional units, $w_{i,t} < w_{i,t+f}$.
- S (sold positions): Assets where the portfolio weight has decreased, requiring the sale of some units, $w_{i,t} > w_{i,t+f}$.

For the first trading period, the return is computed by adjusting the portfolio based on these three subsets, taking into account the impact of market fluctuations during the initial transition period.

Using a weight decomposition, we can consider the portfolio weights as the evolution from time $t-f$ to time t, where the weights are defined as follows:

- $\vec{w}_h$ corresponds to the portfolio weights of the held positions between t and $t+f$ and corresponds, for each instrument, to $min(w_{i,t}, w_{i,t+f})$.
- $\vec{w}_b$ corresponds to the positive weight deltas $\delta_{w_i^+} = w_{i,t+f} - w_{i,t}$, where $w_{i,t+f} - w_{i,t} > 0$, and corresponds to increased positions on an instrument (buys).
- $\vec{w}_s$ corresponds to the negative weight deltas $\delta_{w_i^-} = w_{i,t+f} - w_{i,t}$, where $w_{i,t+f} - w_{i,t} < 0$, and corresponds to decreased positions on an instrument (sales).

$$(\vec{w}_h + \vec{w}_b + \vec{w}_s) \cdot \vec{1} = 1 \quad \text{(ensures portfolio balance)} \tag{2}$$

In regard to the returns of the different positions, the returns of the held positions, r_h are computed as the ratio of closing prices between rebalancing periods $t+f$ and t:

$$r_h = \frac{P_{Close,t+f}}{P_{Close,t}} \tag{3}$$

The returns of the increased positions (buys) are assumed to execute at the opening price of the next trading period after the rebalancing date $t+f+1$, and the return r_b is

$$r_b = \frac{P_{Close,t+f+1}}{P_{Open,t+f+1}} \quad \text{(executed at the opening price after rebalancing)} \tag{4}$$

The returns of the decreased positions (sells) are assumed to execute at the opening price of the next trading day after the rebalancing date $t+f+1$, and the return r_s is

$$r_s = \frac{P_{Open,t+f+1}}{P_{Close,t+f}} \tag{5}$$

For the subsequent periods, returns are computed as the ratio of the closing price at time t over the closing price at time $t-1$. Now, a return trajectory between t and $t+f$ rebalancing dates can be obtained:

$$\forall T \in (t+f+1, t+2f] : r_{pf,T} = \vec{w}_h \cdot \vec{r}_h \tag{6}$$

and for the first trading day after the rebalancing date, it is

$$r_{pf,t+f+1} = \vec{w}_h \cdot \vec{r}_{h,t+f+1} + \vec{w}_b \cdot \vec{r}_{b,t+f+1} + \vec{w}_s \cdot \vec{r}_{s,t+f+1} \tag{7}$$

The returns of the increased and decreased positions are relatively decremented by the transaction costs and slippage. In our study, transaction costs have been fixed to 5 and 2 basis points, respectively. This ensures the realistic modeling of transaction costs and slippage, which are critical in practical portfolio management.

The goal of the agent is to maximize the total cumulative log-return over an entire episode, which consists of 252 periods, representing a full trading year. By optimizing over this time horizon, the agent learns to balance risk and return, adjusting the portfolio dynamically based on market conditions to achieve long-term profitability.

2.3. RL Agents in Portfolio Optimization

In general, RL agents can be broadly categorized into value-based, policy-based, and actor–critic methods. Value-based methods, such as DQNs [24] and Double DQNs [38], estimate the value of each possible action in a given state and select the action with the highest value. Policy-based methods, like PPO [26], directly optimize the policy that maps states to actions, enabling them to handle both discrete and continuous action spaces effectively. Actor–critic methods, such as DDPG [25] and SAC, combine both approaches: the actor determines the actions, while the critic evaluates the value of those actions. These distinctions allow RL agents to address varying portfolio optimization challenges, including action space granularity and decision stability.

Appendix A provides a comprehensive overview of these algorithms, including their architectures, advantages, and limitations.

A DQN is effective in discrete action spaces but is limited by overestimation bias, particularly in volatile financial environments. This bias can lead to suboptimal portfolio decisions, as the algorithm tends to overestimate the value of certain actions in noisy or stochastic market conditions. In the context of portfolio optimization, where asset prices are subject to fluctuations, such overestimation can compromise the robustness of allocation strategies [24,38].

2.4. Environmental Signals

In financial markets, Price represents the equilibrium point where buyers and sellers agree to exchange an asset. It is the result of a dynamic interaction between supply and demand, where buyers are willing to purchase and sellers are willing to sell at a specific price. This constant negotiation leads to the formation of the market price, which adjusts based on new information, investor sentiment, and broader economic factors. The price, therefore, is a key indicator of market equilibrium, capturing the collective behavior and expectations of market participants at any given moment.

2.4.1. Price Data

Price data in financial markets are typically represented using OHLC data, which provide key information about the price movement of an asset over a specific time period. Each OHLC dataset summarizes the price action within that period, irrespective of the sampling frequency, and is commonly visualized using candlestick charts, as shown in Figure 1.

Each candlestick represents the price movement for a single period, with the body showing the range between the opening (O) and closing prices (C) and the wicks indicating the highest (H) and lowest (L) prices during that period. A filled (red/black) body denotes a closing price lower than the opening, while an empty (green/white) body indicates a higher closing price.

By analyzing OHLC data alongside candlestick patterns, traders can make informed decisions about potential entry and exit points in the market. These data are fundamental to technical analysis, providing a visual representation of price action and helping to detect market sentiment shifts. Although [39] highlights that postprocessing is required to obtain representative OHLC data for a specific aggregation period, in this study, it was used as-is so as to avoid the requirement for higher-frequency OHLC data or Limit Order Book (LOB) data to build the aggregate OHLC data and ensure that our research is easily reproducible.

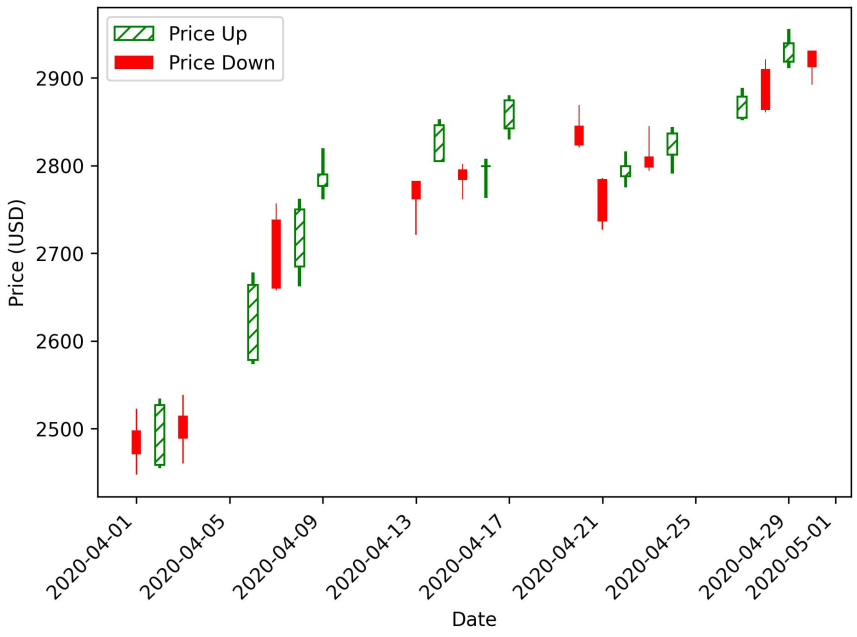

Figure 1. Candlestick chart displaying OHLC (Open-High-Low-Close) prices for the S&P 500 index.

The EMH posits that markets are fully efficient, meaning that all available information is already reflected in asset prices. If the EMH holds true, no amount of technical or fundamental analysis will allow an investor to consistently achieve returns that outperform the market, as prices already represent all known information about an asset's current behavior and future potential. In this view, price alone would contain all relevant data, rendering additional signals from technical indicators redundant or unnecessary. However, critics of the EMH [33] argue that inefficiencies, such as human behavior, market psychology, or anomalies, provide opportunities to generate excess returns through the use of advanced signals beyond just price.

2.4.2. Technical Indicators

Technical indicators are mathematical constructs built from historical price data over a specific period and, often, the volume of trading. Volume represents the total number of shares or contracts traded during a given period.

This study uses a variety of technical indicators to capture market trends and conditions. Table 1 provides a summary of these indicators along with their traditional interpretations, while Appendix B offers detailed calculations and applications.

This summary serves as a quick reference to understand the behavioral characteristics of the indicators and their practical applications in portfolio optimization. Figure 2 provides an overview of the evolution of the technical indicators for the S&P 500 in a window of time.

Table 1. Interpretation of technical indicator values and market behavior.

Market Behavior	AROONOSC	RSI	CCI	CMO	MFI	Williams %R	STOCHF
Overbought	>50 (Uptrend)	>70	>+100	>50	>80	>−20	>80
Oversold	<−50 (Downtrend)	<30	<−100	<−50	<20	<−80	<20
High Volatility	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Low Volatility	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Bullish Divergence	Price ↓, AROON ↑	Price ↓, RSI ↑	Price ↓, CCI ↑	Price ↓, CMO ↑	Price ↓, MFI ↑	Price ↓, %R ↑	Price ↓, STOCHF ↑
Bearish Divergence	Price ↑, AROON ↓	Price ↑, RSI ↓	Price ↑, CCI ↓	Price ↑, CMO ↓	Price ↑, MFI ↓	Price ↑, %R ↓	Price ↑, STOCHF ↓
Momentum Confirmation	>0 (Uptrend)	>50 (Bullish)	>+100 (Uptrend)	>0 (Uptrend)	>50 (Bullish)	>−50 (Bullish)	%K > %D
Trend Reversal	Crosses 0 (Up/Down)	Crosses 50	Crosses +100/−100	Crosses 0	Crosses 80/20	Crosses −20/−80	Crosses 80/20

Market behaviors and the associated directional changes (↑/↓) of Price and Technical indicators

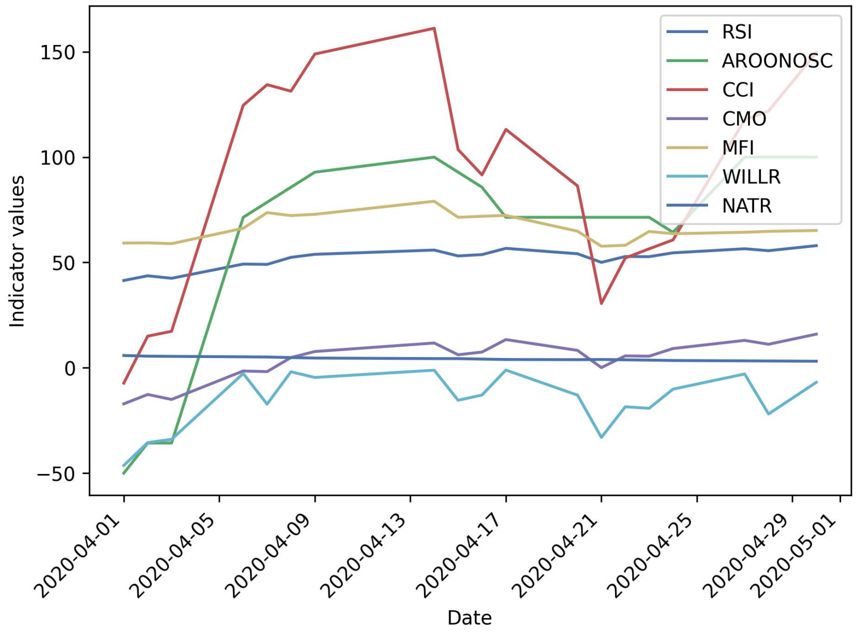

Figure 2. Technical Indicators for S&P 500 for April 2020.

2.5. Feature Extractors

Neural networks, as universal function approximators, have proven highly effective in a range of machine learning tasks due to their ability to approximate complex, non-linear relationships between input data and the target output. The Universal Approximation Theorem [35] demonstrates that a neural network with at least one hidden layer can approximate any continuous function to a desired accuracy, given sufficient parameters and non-linear activation functions. This adaptability makes neural networks particularly well suited for modeling financial data, where the relationships between market signals and asset returns are often non-linear and time-varying.

2.5.1. Multi-Layer Perceptron

The MLP is a fully connected neural network where data are propagated through multiple layers of neurons, each applying a linear transformation followed by a non-linear activation function. In an MLP, the input layer processes raw data (e.g., price, volume, or technical indicators), which are then passed through several hidden layers before reaching the output layer. Each neuron in a given layer is connected to every neuron in the next layer, and the transformation from one layer to the next follows the equation

$$z^{(l)} = W^{(l)}a^{(l-1)} + b^{(l)} \tag{8}$$

where

- $z^{(l)}$ is the pre-activation value at layer l;
- $W^{(l)}$ is the weight matrix for layer l;
- $a^{(l-1)}$ is the output (activation) from the previous layer;
- $b^{(l)}$ is the bias vector.

The output $a^{(l)}$ is then passed through a non-linear activation function σ, such as ReLU (Rectified Linear Unit) or sigmoid:

$$a^{(l)} = \sigma(z^{(l)}) \tag{9}$$

This layered structure enables the MLP to capture complex dependencies in the data.

2.5.2. Convolutional Neural Network

The CNN [22] employs a different architecture optimized for processing structured data, such as images or time series, where local patterns and correlations are important. Instead of fully connecting each layer, CNNs apply convolutional filters to local regions of the input, allowing the network to extract hierarchical features. The operation of a convolutional layer is defined as

$$z_{i,j}^{(l)} = \sum_{m,n} W_{m,n}^{(l)} a_{i+m,j+n}^{(l-1)} + b^{(l)} \tag{10}$$

where

- $z_{i,j}^{(l)}$ is the activation at position (i,j) in layer l;
- $W_{m,n}^{(l)}$ represents the convolutional kernel applied at position (m,n);
- $a_{i+m,j+n}^{(l-1)}$ is the input from the previous layer at the corresponding position;
- $b^{(l)}$ is the bias term.

CNNs are particularly advantageous for tasks where spatial or temporal relationships are crucial, such as identifying trends in financial time-series data. By sharing weights across different regions, CNNs are more parameter-efficient than MLPs, leading to faster training and fewer parameters to tune.

2.5.3. Comparison of MLP and CNN

MLPs are flexible and straightforward to implement, making them applicable to a broad range of problems. However, they require a large number of parameters, which can lead to overfitting, particularly when the input data exhibit strong local correlations, as is often the case with time-series data.

In contrast, CNNs excel in recognizing patterns in structured data, such as the relationships between consecutive time steps in financial data. Through convolutional layers, CNNs reduce the number of trainable parameters, which enhances generalization and decreases the risk of overfitting. However, CNNs can be more computationally intensive during training and inference, especially as the input data size increases.

Overall, MLPs offer greater flexibility at the expense of parameter efficiency, while CNNs are more scalable and effective for structured data but may require more sophisticated architectures and tuning.

In this paper, the MLPs in the different experiments are the vanilla implementations for each agent from the StableBaselines3 [40] library. Table 2 provides a detailed view of each architecture, including the activation functions, number of layers, and neurons used for each agent for the MLP variant.

Table 2. Feature extractor architectures for the MLP for each agent type.

Agent	Component	Layer	Neurons	Activation fn
DQN	Agent	1	64	ReLU
		2	64	ReLU
DDPG	Actor	1	400	ReLU
		2	300	ReLU
	Critic	1	400	ReLU
		2	300	ReLU
PPO	Policy net	1	64	ReLU
		2	64	ReLU
	Value net	1	64	ReLU
		2	64	ReLU
SAC	Actor	1	256	ReLU
		2	256	ReLU
	Critic	1	256	ReLU
		2	256	ReLU

The *input_size* of the first layers is equal to a 1-D vector of dimension 24 $\times$ *lookback* in the experiments with only OHLC prices and 72 $\times$ *lookback* in the experiments with technical indicators from Section 2.4.

For the CNN, a custom model has been implemented with the architecture in Table 3.

Table 3. The architecture of the CNN feature extractor.

Convolutional Layer	Input Channels	Output Channels	Kernel Size	Stride	Padding	Activation fn
2D Convolution	input_size	32	2	1	0	ReLU
2D Convolution	32	64	2	1	0	ReLU
Fully Connected Layer	**Input Size**	**Output Size**				**Activation fn**
Flatten	(64, 32, 2, 2)	6656	-	-	-	-
Linear	6656	128	-	-	-	ReLU

The *input_size* of the first layers is equal to a 3-D tensor of dimension *lookback* $\times$ 6 $\times$ 4 in the experiments with only OHLC prices and *lookback* $\times$ 6 $\times$ 12 in the experiments with technical indicators from Section 2.4.

2.6. Reward Function

In this study, the reward function is based on the cumulative log-returns of the agent over the course of each episode. A *return*, in financial terms, represents the profit or loss generated by an asset over a specific time interval. While both arithmetic and log-returns are commonly used, log-returns are generally preferred in this context for several reasons. One of the key advantages of log-returns is their additive property over multiple periods, which facilitates the calculation of cumulative returns in reinforcement learning (RL) settings. This is particularly important when summing rewards over an entire episode. However, there is some debate in the literature regarding the appropriateness of log-returns, as certain studies suggest they may underestimate extreme price fluctuations compared to arithmetic returns [6,41]. The log-return for a given time period is computed as

$$r_t = log(\frac{P_{\text{close}}}{P_{\text{open}}}) \tag{11}$$

where P_{close} is the closing price, and P_{open} is the opening price for the time period t.

The reward function is designed to align with the agent's objective of maximizing cumulative log-returns over an entire episode. The log transformation simplifies the reward computation, allowing the total reward R for an episode to be expressed as the sum of the log-returns between rebalancing periods. For each rebalancing interval, the log-return is calculated based on the portfolio value V at the beginning and end of the interval. The portfolio value is derived as the dot product of the asset weights w and the corresponding asset prices P at each time step:

$$R = \sum_{t=1}^{T} \log\left(\frac{V_t}{V_{t-1}}\right) \tag{12}$$

where V_t represents the portfolio value at time t, which is given by

$$V_t = \vec{w}_t \cdot \vec{P}_t \tag{13}$$

with w_t being the vector of portfolio weights and P_t the asset prices at time t.

Cumulative returns are a well-established metric within the financial industry, frequently used to evaluate and compare the performance of various investment strategies. One of the main advantages of cumulative returns is their ability to be annualized, allowing for consistent comparisons across different investment horizons and rebalancing frequencies, typically expressed in terms of years. This makes it easier to analyze both the risk (volatility) and reward (return) in a standardized manner. Moreover, cumulative log-returns are widely employed as the reward metric in RL-based portfolio optimization studies. Although cumulative returns are the most common choice for RL agents, alternative reward metrics, such as the Sharpe ratio, which adjusts returns based on risk, or other measures, like the Sortino ratio, Calmar ratio, and maximum drawdown (Section 2.7.5), are also used to better account for downside risk [13,30].

2.7. Experimental Design

2.7.1. Data Sources

The raw OHLC price data utilized in this study were obtained through the *Yahoo! Finance* API via the Python package *yfinance*. These data comprise historical prices for the financial instruments under investigation. Additionally, technical indicators were derived using the *TALIB* library, which provides an extensive collection of technical analysis tools in Python. All datasets used in this study are publicly available at [42], ensuring the full replicability of the experimental results.

The analysis focuses on six financial instruments with varying return profiles, complemented by an additional instrument representing a risk-free asset with a constant return of 1, serving as a stable, inflation-adjusted instrument. The six financial instruments examined are as follows:

1. ^GSPC (S&P 500): This index tracks the performance of 500 large-cap companies listed on U.S. stock exchanges. It is widely regarded as a benchmark for the overall health of the U.S. economy and is dominated by blue-chip companies across various sectors, including technology, healthcare, and financial services.
2. ^GDAXI (DAX 30): The DAX 30 is composed of 30 large-cap companies listed on the Frankfurt Stock Exchange. It is a key indicator of the German economy and represents major industries, such as automotive, industrial goods, and chemicals, making it a key player in European markets.
3. ^IXIC (NASDAQ Composite): The NASDAQ Composite tracks a broad set of stocks listed on the NASDAQ, with a significant focus on technology and growth-oriented

companies. It includes large-cap tech giants as well as smaller, innovative companies, making it a leading index in global technology markets.

4. ^RUT (Russell 2000): This index represents small-cap companies primarily from the U.S. It is widely used as a benchmark for the performance of smaller, growth-oriented businesses and often reflects the health of the domestic economy in comparison to larger multinational firms.
5. ^N225 (Nikkei 225): The Nikkei 225 represents the Japanese stock market and tracks 225 large-cap companies listed on the Tokyo Stock Exchange. The index is a key measure of the economic performance of Japan, with significant representation from manufacturing, technology, and financial sectors.
6. ^FTSE (FTSE 100): The FTSE 100 index tracks the top 100 companies listed on the London Stock Exchange by market capitalization. It is heavily weighted toward multinational companies in sectors such as energy, financial services, and consumer goods, making it a key indicator of both U.K. and global economic conditions.

These instruments provide diverse representations of global markets, offering a broad perspective for evaluating portfolio performance across varying economic environments.

2.7.2. Simulation Environment

The simulation environment employed in this study is a custom implementation developed in accordance with the *Gym* [43] and *Gymnasium* [44] standards. It is specifically designed to train reinforcement learning agents from the *StableBaselines3* [40] library, as described in Section 2.3, and is tailored to the requirements of portfolio optimization. This environment ensures compatibility with various RL agents while providing realistic financial simulations for effective agent training.

In each rebalancing period, the environment generates an observation consisting of time-series data of selected technical indicators, as described in Section 2.4. The observation covers the preceding *lookback_steps*, providing the RL agents with the necessary context to make informed portfolio allocation decisions. The way the observations are structured depends on the feature extractor network used:

- MLP Configuration: The observation is flattened into a 1-D vector of size $1 \times X$, where X is determined by the product of the number of instruments, the lookback periods, and the number of technical indicators (i.e., $X = n_{\text{instruments}} \times \text{lookback\_periods} \times \text{technical\_indicators}$). Each instrument's data over the lookback window are concatenated sequentially across this vector.
- CNN Configuration: The observation is organized as a 3-D tensor with dimensions lookback_periods $\times$ $n_{\text{instruments}}$ $\times$ technical_indicators. This format preserves the temporal and spatial relationships between the data points, enabling the CNN to identify both time-series trends and correlations across instruments.

Figure 3 provides an illustration of an observation with $lookback = 3$ for the same technical indicator across four different investment instruments, shown as both a 1-D vector for the MLP feature extractor and a tensor with dimensions $lookback \times indicators \times instruments$ for the CNN. Additional indicators will be concatenated along the same axis for the 1-D vector (MLP) and stacked along the Y-axis for the 3-D tensor (CNN).

The output of the feature extractor is designed to encapsulate the context provided by the recent history of technical indicators, creating a latent representation of the market state. This representation is then passed to the agent network, where it is used to either map directly to an action in continuous-action agents or to estimate the Q-value for discrete-action agents based on the current latent state.

The action selected by the agent varies according to the type of RL algorithm employed:

- For discrete-action agents such as the DQN, the action corresponds to a full allocation to a single instrument, with the agent selecting the action via an argmax operation rather than normalizing Q-values to generate continuous portfolio weights.

- For continuous-action agents, the action represents the percentage allocation across instruments, ensuring that $\forall_i : w_i \geq 0$ and $\sum_i w_i = 1$, which guarantees long-only positions and a fully invested portfolio.

The reward function in the environment is computed as the log-return between consecutive rebalancing periods. This is calculated by multiplying the portfolio weights by the prices of each instrument, as outlined in Section 2.6. This framework enables the agent to optimize cumulative log-returns over the duration of the episode.

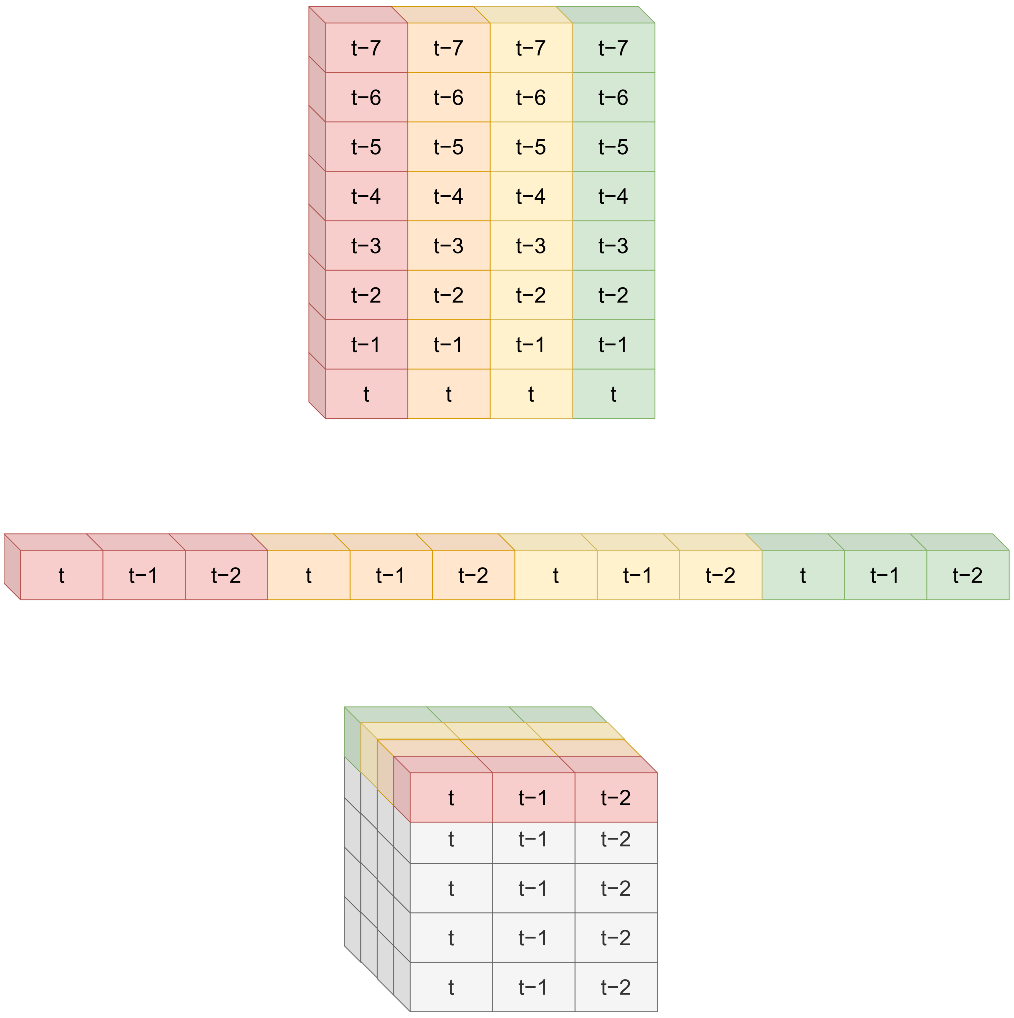

Figure 3. Environment observations as vector for MLP and tensor for CNN. Each color represents a different technical indicator for a chosen investment instrument

2.7.3. Training and Evaluation

The training dataset covers the period from mid-November 1987 to the end of December 2015. This starting point was deliberately selected to exclude the extreme market volatility caused by Black Monday on 19 October 1987,and the subsequent market disruptions, which could introduce bias into the training process. Due to the inclusion of technical indicators such as moving averages, the training data become fully effective starting in February 1988, as sufficient historical data are required to compute these indicators reliably. Consequently, the effective training period spans from February 1988 to December 2015.

The evaluation dataset encompasses the period from January 2016 to April 2024, inclusive. This evaluation phase includes a wide range of market conditions, offering a robust test environment for the RL agents. It covers the bullish market trends of 2021 and 2023, as well as periods of bearish and sideways movements, notably including the market disruptions caused by the COVID-19 pandemic. The diversity of market conditions in this evaluation period allows for a comprehensive assessment of the agents' adaptability and performance across different market environments.

For technical indicators, particularly those that rely on moving averages, no embargo period has been implemented, as suggested by [39], since the indicators used in this study do not exhibit target leakage. Consequently, the entirety of the evaluation data has been utilized without restrictions.

To ensure the robustness and validity of the results, the evaluation period from 2016 to April 2024 has been consistently applied across all RL agents and baseline models. This consistent evaluation timeframe guarantees that each model is tested under identical market conditions, allowing for a fair and meaningful comparison of their performance.

2.7.4. Experimental Parameters and Configurations

In addition to varying the agent algorithms (DQN, DDPG, PPO, and SAC) (Section 2.3) and the two types of feature extractors (MLP and CNN) (Section 2.5), several key parameters were adjusted across experiments to evaluate agent performance under diverse conditions. One critical parameter is the rebalancing frequency, which was set to either continuous rebalancing ($f = 1$) or periodic rebalancing every 10 periods (approximately bi-weekly). The lookback period—the number of historical data points used for decision-making—was also varied, with configurations of 16 and 28 periods to examine the influence of shorter versus longer historical contexts (Section 2.1).

Additionally, the experiments varied the type of data input into the RL agents, either using raw OHLC prices or incorporating technical indicators derived from these prices (Section 2.4). This setup allows for an analysis of how the inclusion of more complex market signals influences the agents' portfolio optimization strategies.

The set of financial instruments remained constant across all experiments, consisting of six indices plus a risk-free asset (Section 2.7.1). Each experiment consisted of 252 periods, approximating one full investment year.

To address the convergence challenges observed in certain RL agents, the best-performing model—defined as the one with the highest average reward over the preceding 100 episodes—was retained for testing with the evaluation set. To maintain consistency across agents, a batch size of 256 and a total of 10,000,000 timesteps were applied to all training runs. These standardized parameters ensured a rigorous and fair comparison between different agent configurations and experimental setups. Table 4 summarizes the different options for each of the experiment parameters.

Table 4. Experiment parameters and configuration options.

Parameter	Option	Total Options
RL agent	DQN DDPG PPO SAC	4
Feature extractor	MLP CNN	2
Rebalancing frequency	continuous (1) bi-weekly (10)	2
Lookback periods	16 28	2
Environmental signals	OHLC (dim = 24) OHLC + technical indicators (dim = 72)	2

Experimental configurations for a total of 64 experiments.

2.7.5. Financial Metrics

In portfolio optimization, relying solely on final reward values or loss metrics, such as actor and critic losses, may not provide a comprehensive evaluation of an RL agent's performance in real-world financial contexts. These internal metrics are primarily focused on the agent's learning process but do not adequately reflect the key financial outcomes that investors prioritize, such as risk-adjusted returns or resilience to large drawdowns. To provide a robust and meaningful comparison of RL agents, it is essential to assess performance using established financial metrics that account for both return and risk. These metrics offer a clearer perspective on how each strategy would perform in real-world market conditions.

For this study, we employ the following financial metrics: annualized return, annualized volatility, Sharpe ratio [45], Sortino ratio [46], and Calmar ratio [47]. Additionally, we include maximum drawdown to measure the worst-case scenario in terms of wealth loss. Together, these metrics provide a comprehensive assessment of portfolio performance, considering different dimensions of risk and return.

1. Annualized Return: This metric measures the yearly compounded return of the portfolio, making it easier to compare results across different time horizons.

$$R_{\text{Pf,annualized}} = \left(\prod_{t=1}^{T} (1 + r_t) \right)^{\frac{1}{T}} - 1 \tag{14}$$

   where r_t represents the return at time t, and T is the number of time periods in a year.
2. Annualized Volatility: Volatility is a measure of risk, reflecting the standard deviation of the portfolio's returns over time, annualized to match the return horizon.

$$\sigma_{\text{Pf,annualized}} = \sigma \times \sqrt{T} \tag{15}$$

   where σ is the standard deviation of returns, and T is the number of time periods in a year.
3. Sharpe Ratio: The Sharpe ratio measures risk-adjusted returns, considering the excess return over the risk-free rate relative to volatility.

$$R_{\text{Sharpe}} = \frac{R_{\text{Pf,annualized}} - R_{\text{f}}}{\sigma_{\text{Pf,annualized}}} \tag{16}$$

   where R_{f} is the risk-free rate.
4. Sortino Ratio: The Sortino ratio is similar to the Sharpe ratio, but it penalizes only downside volatility, ignoring positive deviations from the mean return.

$$R_{\text{Sortino}} = \frac{R_{\text{annualized}} - R_{\text{risk-free}}}{\sigma_{\text{downside}}} \tag{17}$$

   where σ_{downside} is the standard deviation of negative returns.
5. Maximum Drawdown: This metric measures the largest peak-to-trough decline in portfolio value, providing insight into the worst-case wealth loss.

$$\text{Maximum Drawdown} = \frac{\max(V_t) - \min(V_t)}{\max(V_t)} \tag{18}$$

   where V_t is the portfolio value at time t.

6. Calmar Ratio: The Calmar ratio is a risk-adjusted return metric that evaluates the portfolio's performance relative to its maximum drawdown.

$$R_{\text{Calmar}} = \frac{R_{\text{Pf, annualized}}}{\text{Maximum Drawdown}} \tag{19}$$

These metrics complement each other by offering a balanced view of portfolio performance. While the Sharpe ratio uses symmetrical volatility as a risk measure, the Sortino ratio focuses solely on downside risk, making it more suitable for risk-averse investors. The Calmar ratio and maximum drawdown, on the other hand, emphasize worst-case loss scenarios, which are particularly relevant for investors focused on capital preservation. By combining these metrics, we can assess which RL agents and configurations excel in maximizing returns, minimizing risk, or avoiding significant drawdowns, depending on the specific investment objectives.

2.7.6. Baseline Models

To provide a comprehensive evaluation of the RL agent configurations, it is essential to compare their performance not only against each other but also relative to established market benchmarks. This ensures that the results are meaningful and realistic within the broader financial context, allowing for a more accurate assessment of the RL-based strategies.

Financial markets are dynamic, experiencing various economic cycles with bullish, bearish, and sideways trends. To account for this variability, we compare the RL agents' performance against well-recognized benchmarks and simple portfolio models that capture overall market movements. This contextualizes the agents' performance by relating it to market conditions, providing a point of reference for evaluating the robustness of each model.

One straightforward baseline approach is the equally weighted portfolio (EWP), where each asset in the portfolio is assigned an equal weight. This strategy assumes equal currency allocations across all instruments, simplifying the process by automatically adjusting to market movements without introducing additional complexity. It assumes that the assets are fully liquid and fractional. Additionally, we employ the S&P 500 index as an industry-standard benchmark. This index, which tracks the top 500 U.S. companies, serves as a widely accepted reference for evaluating portfolio performance.

For the EWP, given that there are 6 instruments (Section 2.7.1: ^GSPC, ^GDAXI, ^IXIC, ^RUT, ^N225, ^FTSE) plus an additional risk-free asset, the portfolio weights are a constant vector of dimension $1 \times n_{\text{instruments}}$, where each element of the vector is $\frac{1}{n_{\text{instruments}}}$. With $n_{\text{instruments}} = 7$, the weights are equally distributed across all instruments:

$$\vec{w}_{\text{ewp}} = \left[\frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}\right] \tag{20}$$

For the S&P 500 benchmark (market portfolio), the portfolio is fully invested in the ^GSPC instrument, representing the market index. The portfolio weights are a constant vector, where only the S&P 500 index receives a non-zero weight:

$$\vec{w}_{\text{market}} = [1, 0, 0, 0, 0, 0, 0] \tag{21}$$

These two portfolio models—the equally weighted portfolio and the market portfolio—serve as baselines for evaluating the RL agent configurations. By comparing their performance to these benchmarks, we gain insight into how well the RL agents perform in relation to standard, market-based investment strategies. The cumulative reward of the market and EQP portfolio for the period 2016-April 2024 is shown in Figure 4.

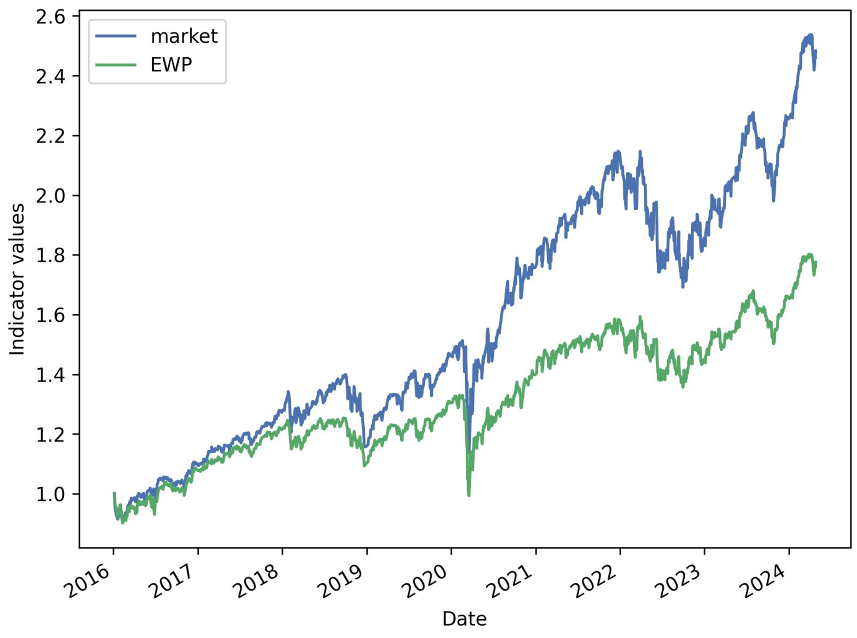

Figure 4. Cumulative returns for the market (S&P 500 proxy) and the EWP for the period 2016–April 2024.

3. Results

3.1. Agents' Performance

The results of the different RL agent configurations are summarized in terms of key financial metrics. These metrics were calculated for each configuration of agent type, feature extractor, lookback period, and rebalancing frequency and are shown in Table 5.

Table 5. Performance metrics for top-10 RL agent configurations.

Ann. Return	Ann. Volatility	Calmar	Sharpe	Sortino	Max Drawdown	Agent	Feat.Ext.	Lookback	Reb. Freq.	Indicators
0.26	0.38	0.56	0.8	1.12	−0.47	DQN	CNN	16	1	No
0.26	0.46	0.45	0.74	1.02	−0.57	DQN	CNN	28	1	Yes
0.26	0.38	0.55	0.8	1.12	−0.47	DQN	MLP	28	1	No
0.19	0.37	0.34	0.66	0.91	−0.57	DDPG	MLP	16	10	No
0.19	0.31	0.39	0.72	1	−0.49	DDPG	MLP	28	1	Yes
0.19	0.32	0.37	0.7	0.96	−0.5	DDPG	CNN	28	10	Yes
0.19	0.25	0.55	0.8	1.18	−0.33	PPO	MLP	16	1	Yes
0.18	0.35	0.31	0.66	0.91	−0.59	DQN	MLP	16	1	No
0.18	0.34	0.5	0.66	0.95	−0.36	DQN	MLP	16	1	Yes
0.18	0.29	0.44	0.7	0.97	−0.39	DDPG	MLP	16	1	Yes
0.14	0.19	0.57	0.8	1.12	−0.25	^GSPC	-	-	1	-
0.09	0.14	0.35	0.69	0.94	−0.25	EWP	-	-	1	-

Top-10 experimental results by annualized return compared to the benchmark portfolios (in grey).

For example, the DQN agent using an MLP feature extractor and a 16-period lookback window achieved an annualized return of 18.25% with a Sharpe ratio of 0.66. A similar configuration with a 28-period lookback showed a higher annualized return of 25.76% and a Sharpe ratio of 0.80. These results suggest that longer lookback periods can yield higher returns but introduce additional risk, particularly during periods of market volatility.

On the other hand, the PPO agent with an MLP feature extractor exhibited underwhelming performance, with an annualized return of −1.27% and a low Sharpe ratio, indicating inferior results when compared to the baselines, as shown in Table 6.

Table 6. Performance metrics for bottom-10 RL agent configurations and baselines.

Ann. Return	Ann. Volatility	Calmar	Sharpe	Sortino	Max Drawdown	Agent	Feat.Ext.	Lookback	Reb. Freq.	Indicators
0.14	0.19	0.57	0.80	1.12	−0.25	^GSPC	-	-	1	-
0.09	0.14	0.35	0.69	0.94	−0.25	EWP	-	-	1	-
0.04	0.31	0.09	0.29	0.40	−0.50	PPO	MLP	16	10	Yes
0.03	0.30	0.05	0.23	0.32	−0.52	DQN	CNN	28	10	No
0.02	0.31	0.02	0.21	0.28	−0.69	PPO	CNN	16	1	No
0.01	0.26	0.02	0.19	0.25	−0.59	DQN	MLP	28	10	Yes
0.01	0.03	0.27	0.36	0.63	−0.04	DQN	CNN	28	10	Yes
0.00	0.02	0.11	0.18	0.25	−0.03	DQN	CNN	16	10	Yes
0.00	0.25	−0.01	0.12	0.14	−0.62	PPO	MLP	28	1	No
−0.01	0.39	−0.02	0.17	0.24	−0.51	DQN	CNN	16	1	Yes
−0.01	0.24	−0.02	0.07	0.09	−0.51	PPO	MLP	16	1	No
−0.04	0.28	−0.06	0.00	0.00	−0.66	PPO	CNN	28	10	Yes

Bottom-10 experimental results by annualized return compared to the benchmark portfolios (in grey).

When comparing RL agents to the market benchmark (S&P 500) and the equally weighted portfolio (EWP), we observed the following:

- The S&P 500 baseline had a Sharpe ratio of 0.80 and an annualized return of 14.18%, which outperformed several agent configurations, particularly those with shorter lookback periods and high rebalancing frequency.
- The equally weighted portfolio baseline yielded an annualized return of 8.73% and a Sharpe ratio of 0.68, serving as a middle-ground benchmark that several RL agents surpassed.

Additionally, it is worth mentioning that the SAC experiments experienced convergence issues during training. Despite several tuning attempts, the models did not converge consistently, and as a result, the SAC agent configurations were excluded from the final analysis due to their unreliable performance.

In summary, the DQN consistently outperforms the other agent families in terms of annualized return and risk-adjusted metrics such as the Sharpe, Sortino, and Calmar ratios, especially when using longer lookback windows. On the other hand, DDPG excels in managing risk, demonstrating the lowest volatility, downside risk, and max drawdown among the agent families, making it more robust in risk-averse scenarios. Regarding feature extractors, the CNN tends to perform better in terms of maximizing returns and achieving higher risk-adjusted ratios (Sharpe, Sortino, Calmar), whereas the MLP shows superior performance in risk mitigation, as evidenced by lower volatility and max drawdown. This suggests that the choice between the CNN and MLP depends on whether the primary focus is on maximizing returns or minimizing risk.

3.2. Instrument Allocation

The overall proportion of each instrument in a portfolio is calculated by aggregating the total currency allocated to that instrument throughout the entire testing period. This cumulative allocation is divided by the total cumulative portfolio value for the same period. This method provides a robust measure of average exposure to each financial instrument, accounting for market fluctuations and changes in portfolio weights over time. This allows us to assess the relative importance of each instrument in the portfolio and observe how consistently agents favored specific assets during the evaluation phase.

The S&P 500 (^GSPC) and NASDAQ (^IXIC) indices are prominently favored across many agent configurations and feature extractors. DDPG agents with MLP feature extractors, for instance, frequently allocate a large portion of the portfolio to ^GSPC, indicating its dominance. Similarly, ^IXIC receives significant allocation in some setups, such as PPO with CNN extractors.

In contrast, the Russell 2000 (^RUT) and FTSE (^FTSE) indices are either underrepresented or completely absent in several configurations, suggesting that the agents might find these indices less favorable for maximizing returns given the prevailing market conditions. Their lower inclusion could indicate that these indices are less predictable or offer lower risk-adjusted returns.

CASH, representing the risk-free asset, holds a significant portion in some agent configurations, particularly with DQN agents using CNN feature extractors. This suggests a conservative, risk-averse strategy, where the agent opts to hedge against market volatility by allocating a larger portion of the portfolio to CASH. In certain DQN configurations, such as with CNN feature extraction, the allocation to CASH is notably high, indicating a preference for safety during periods of market instability.

In general, DDPG agents (with either MLP or CNN extractors) show more diversified portfolios across various instruments, whereas DQN agents tend to focus on fewer instruments or exhibit extreme concentration in one or two indices. Some DQN configurations allocate the entire portfolio to a single asset, as observed in cases with complete allocation to ^IXIC as summarised in Table 7.

Table 7. Instrument allocation per agent.

^RUT	^IXIC	^GDAXI	^FTSE	^N225	^GSPC	CASH	Agent	Feat.Ext.	Lookback	Reb. Freq.	Indicators
0.00	0.27	0.27	0.00	0.03	0.14	0.29	DDPG	MLP	16	1	No
0.15	0.00	0.11	0.00	0.03	0.36	0.36	DDPG	MLP	28	1	No
0.12	0.59	0.02	0.01	0.00	0.05	0.21	DQN	MLP	16	1	No
0.00	0.00	1.00	0.00	0.00	0.00	0.00	DQN	MLP	28	1	No
0.14	0.09	0.17	0.25	0.11	0.10	0.13	PPO	MLP	16	1	No
0.11	0.15	0.06	0.12	0.23	0.28	0.06	PPO	MLP	28	1	No
0.21	0.21	0.00	0.00	0.15	0.21	0.21	DDPG	MLP	16	1	Yes
0.25	0.25	0.25	0.00	0.25	0.00	0.00	DDPG	MLP	28	1	Yes
0.01	0.02	0.18	0.00	0.78	0.01	0.00	DQN	MLP	16	1	Yes
0.00	0.00	0.00	0.00	0.00	1.00	0.00	DQN	MLP	28	1	Yes
0.17	0.06	0.15	0.32	0.15	0.04	0.11	PPO	MLP	16	1	Yes
0.13	0.12	0.13	0.15	0.18	0.14	0.14	PPO	MLP	28	1	Yes
0.00	0.39	0.03	0.00	0.00	0.19	0.38	DDPG	MLP	16	10	No
0.25	0.02	0.07	0.12	0.23	0.25	0.06	DDPG	MLP	28	10	No
0.00	0.01	0.09	0.61	0.00	0.28	0.01	DQN	MLP	16	10	No
0.50	0.01	0.04	0.00	0.20	0.00	0.24	DQN	MLP	28	10	No
0.14	0.10	0.25	0.03	0.28	0.07	0.12	PPO	MLP	16	10	No
0.22	0.29	0.27	0.02	0.03	0.11	0.07	PPO	MLP	28	10	No
0.28	0.33	0.00	0.05	0.33	0.02	0.00	DDPG	MLP	16	10	Yes
0.31	0.31	0.31	0.03	0.03	0.00	0.01	DDPG	MLP	28	10	Yes
0.38	0.00	0.38	0.02	0.10	0.01	0.11	DQN	MLP	16	10	Yes
0.39	0.01	0.04	0.16	0.02	0.28	0.10	DQN	MLP	28	10	Yes
0.01	0.08	0.13	0.08	0.10	0.26	0.35	PPO	MLP	16	10	Yes
0.17	0.16	0.02	0.09	0.13	0.39	0.04	PPO	MLP	28	10	Yes
0.25	0.00	0.25	0.25	0.00	0.00	0.25	DDPG	CNN	16	10	No
0.08	0.01	0.20	0.20	0.18	0.14	0.20	DDPG	CNN	16	1	No
0.00	0.29	0.00	0.29	0.00	0.29	0.13	DDPG	CNN	28	10	No
0.01	0.31	0.31	0.00	0.00	0.06	0.30	DDPG	CNN	28	1	No
0.00	0.00	0.01	0.00	0.00	0.00	0.99	DQN	CNN	16	10	No
0.00	0.00	0.00	0.00	1.00	0.00	0.00	DQN	CNN	16	1	No
0.32	0.00	0.03	0.65	0.00	0.00	0.00	DQN	CNN	28	10	No
0.00	0.00	0.00	0.00	0.00	1.00	0.00	DQN	CNN	28	1	No
0.12	0.08	0.46	0.16	0.02	0.13	0.03	PPO	CNN	16	10	No
0.02	0.19	0.13	0.06	0.25	0.14	0.20	PPO	CNN	16	1	No
0.01	0.15	0.06	0.07	0.01	0.11	0.60	PPO	CNN	28	10	No

Table 7. *Cont.*

^RUT	^IXIC	^GDAXI	^FTSE	^N225	^GSPC	CASH	Agent	Feat.Ext.	Lookback	Reb. Freq.	Indicators
0.10	0.27	0.05	0.26	0.07	0.04	0.21	PPO	CNN	28	1	No
0.04	0.24	0.24	0.03	0.00	0.24	0.20	DDPG	CNN	16	10	Yes
0.13	0.00	0.22	0.21	0.22	0.22	0.00	DDPG	CNN	16	1	Yes
0.00	0.20	0.20	0.00	0.20	0.20	0.20	DDPG	CNN	28	10	Yes
0.02	0.37	0.23	0.06	0.32	0.00	0.00	DDPG	CNN	28	1	Yes
0.00	0.00	0.00	0.00	0.00	1.00	0.00	DQN	CNN	16	10	Yes
0.00	0.00	0.01	0.00	0.00	0.00	0.99	DQN	CNN	16	1	Yes
0.98	0.00	0.00	0.00	0.01	0.00	0.00	DQN	CNN	28	10	Yes
0.05	0.00	0.95	0.00	0.00	0.00	0.00	DQN	CNN	28	1	Yes
0.20	0.04	0.10	0.22	0.29	0.03	0.12	PPO	CNN	16	10	Yes
0.03	0.49	0.17	0.05	0.11	0.00	0.14	PPO	CNN	16	1	Yes
0.11	0.19	0.08	0.34	0.25	0.02	0.01	PPO	CNN	28	10	Yes
0.07	0.31	0.06	0.01	0.29	0.18	0.08	PPO	CNN	28	1	Yes

Instrument allocation weights.

The top-performing portfolios (top 10) typically demonstrate a more diversified asset allocation, with balanced distributions across multiple instruments. For instance, in the *DDPG-MLP-28* configuration using technical indicators, the allocation is spread evenly, with 25% allocated across different assets and minimal reliance on cash. This reflects a more aggressive investment strategy aiming to capitalize on growth across various markets.

In contrast to the top performers, the worst-performing portfolios (bottom 10) display either extreme concentration or an over-allocation to CASH, limiting their potential to capture the upside during market rallies. For example, some *DQN-CNN* configurations are almost entirely allocated to the risk-free asset (CASH), neglecting other instruments, which severely hampers growth potential. Similarly, certain CNN-based portfolios show poor diversification, focusing heavily on one or two instruments, such as ^GDAXI, with little to no allocation to other markets. This lack of diversification reduces the ability to mitigate risks, resulting in underperformance. Additionally, these portfolios often feature a higher reliance on CASH, reflecting a more conservative stance, but this ultimately sacrifices opportunities for growth in more favorable market conditions.

4. Discussion

4.1. RL Agents

The primary goal of the RL agents was to maximize cumulative log-returns, aiming for a consistent increase in the portfolio's value over time. While risk management is fundamental in traditional portfolio optimization, in this study, risk was implicitly addressed only when the agent incurred losses, reflected as negative log-returns. The allocation weights assigned to different instruments, such as ^IXIC, ^GSPC, and ^N225, illustrate how each agent family approached asset allocation. SAC models, however, faced convergence issues and are excluded from further discussion due to unreliable results.

Distinct patterns emerged among the different RL agent families. DDPG agents, which operate in continuous action spaces, exhibited greater portfolio diversification, distributing allocations across multiple instruments. For example, the *DDPG-CNN-16* configuration allocated equal weights (0.25) to ^RUT, ^IXIC, and ^FTSE while maintaining a CASH allocation of 0.25, demonstrating a balanced approach. In contrast, DQN and PPO agents, operating in discrete action spaces, tended to focus on identifying and fully investing in a single, high-performing asset. For instance, *DQN-MLP-28* allocated 100% of its portfolio to ^GDAXI, reflecting a concentrated strategy aimed at maximizing returns, with less consideration for risk diversification.

Quantitatively, DDPG agents averaged an annualized return of approximately 15.4%, with an annual volatility of 31%, suggesting a well-balanced approach between risk and reward. DQN agents achieved a slightly lower average annualized return of 13.2%, with a higher volatility of 34%, reflecting a more aggressive strategy with greater exposure to risk. PPO agents, on the other hand, delivered more moderate returns of around 9%, paired with volatility levels of approximately 25%, indicating a more conservative approach with lower risk but also reduced return potential.

In comparison to the baseline models, RL agents consistently outperformed both the ^GSPC and equally weighted portfolio (EWP) benchmarks in terms of cumulative log-returns as reflected in Figure 5. The ^GSPC achieved an annualized return of 14.18%, with a volatility of 18.7%, while the EWP benchmark yielded a lower annualized return of 8.73%, paired with 13.5% volatility. Several RL configurations, particularly DDPG and DQN agents, exceeded these returns—DDPG with CNN reached over 18%—but with the trade-off of higher volatility. Notably, DDPG agents demonstrated superior diversification compared to the baseline models, which leaned heavily on a few key instruments. The dynamic asset allocation exhibited by RL agents, which adapted to market conditions, contrasts with the static allocations of the baselines. This adaptability suggests a more robust approach to optimizing portfolio performance.

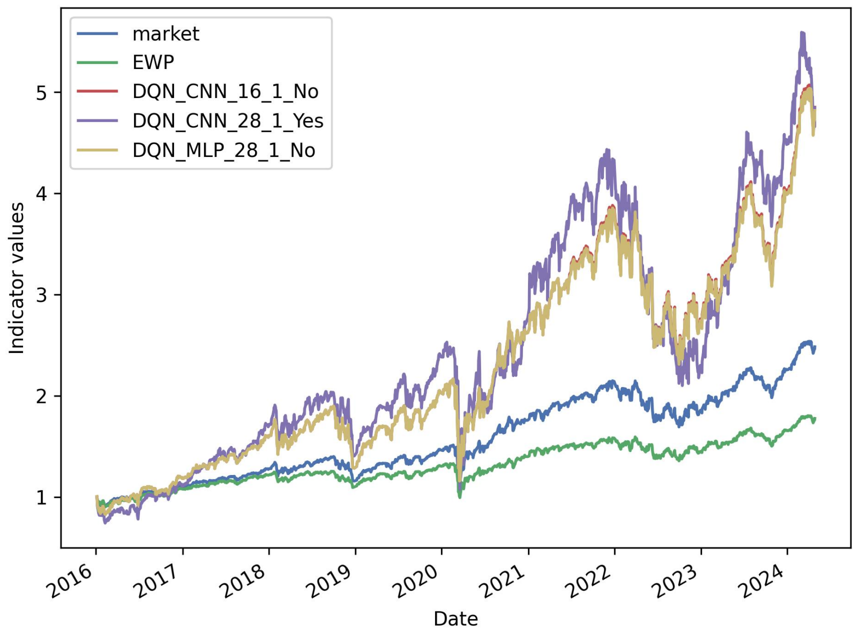

Figure 5. Performance comparison of RL agent portfolios and baseline models.

4.2. *Environmental Observations and Future Market Conjunctures*

The selection of feature extractors (CNN vs. MLP) and lookback periods (16 vs. 28 periods) plays a crucial role in the agent's ability to forecast market trends and adjust portfolio allocations accordingly. CNNs, known for their capacity to capture localized patterns in sequential data, demonstrated superior performance in identifying shifts in volatility and market trends. For instance, the *DQN-CNN-16* configuration, despite not using technical indicators, achieved Sharpe and Sortino ratios identical to the market portfolio (0.8 and 1.12, respectively) while also capturing an excess of 12% in annualized returns. This performance highlights the CNN's ability to effectively recognize upward trends and maintain a favorable return-to-risk ratio. The elevated Sortino ratio, in particular, suggests that CNN-based agents capitalized on volatile growth periods while successfully minimizing downside risk. In comparison, the EWP, with a Sharpe ratio of 0.69 and Sortino ratio of 0.94, exhibited a more conservative profile, which CNN-based agents often outperformed in more dynamic market conditions.

In contrast, MLP-based agents, particularly those using shorter lookback periods (e.g., 16 periods), struggled to effectively anticipate market downturns. For example, *PPO-MLP-16* experienced higher maximum drawdowns, reflecting its difficulty in adjusting to rapid market declines. These MLP-based agents frequently underperformed relative to the EWP in risk-adjusted returns, as indicated by their lower Sharpe ratios and comparable Sortino ratios. For instance, *DDPG-MLP-16* yielded a Sharpe ratio of 0.66 and a Sortino ratio of 0.91, showing moderate performance but an inability to capitalize on bullish periods as effectively as CNN-based models. The market portfolio itself often outperformed these MLP configurations during periods of heightened volatility.

In summary, CNNs—especially with longer lookback periods—were more adept at recognizing and reacting to periods of volatility, allowing for higher allocations to riskier, high-reward assets like ^IXIC and ^GSPC. On the other hand, MLPs, particularly those with shorter lookback windows, exhibited more conservative behavior but were less effective at adapting to shifting market conditions. Overall, CNN-based models delivered stronger risk–return performance, as evidenced by superior Sharpe and Sortino ratios, outperforming MLP configurations and baseline portfolios in many cases.

4.3. Impact of Rebalancing Frequency on Performance

The choice between continuous rebalancing (every 1 period) and periodic rebalancing (every 10 periods) introduces clear trade-offs in terms of performance and risk–reward dynamics. Continuous rebalancing allows agents to capture short-term market movements more effectively, potentially leading to higher returns. However, this approach also incurs increased transaction costs and slippage due to frequent adjustments. For instance, *DDPG-MLP-16* with continuous rebalancing achieved an annualized return of 17.5% and a Sharpe ratio of 0.70. However, it also experienced a downside risk of 21.04%, reflecting the heightened costs and volatility associated with more frequent market exposure. In comparison, the same agent using a 10-period rebalancing strategy delivered a slightly lower return of 17.4%, but with a marginally lower Sharpe ratio of 0.67 and reduced downside risk of 22.47%. This suggests that less frequent trading helped mitigate exposure to transaction costs and volatility.

Periodic rebalancing, by contrast, allowed agents to avoid some inefficiencies associated with daily slippage, leading to more favorable risk-adjusted performance. For example, *PPO-CNN-16* with a 10-period rebalancing strategy achieved an annualized return of 18.5% and a Sharpe ratio of 0.72. In contrast, the same agent with continuous rebalancing delivered a slightly lower Sharpe ratio of 0.68, despite similar volatility levels and a marginally lower downside risk. These results highlight that while continuous rebalancing may offer more responsiveness to market changes, the higher transaction costs and slippage associated with frequent adjustments tend to erode its long-term advantage over periodic rebalancing.

4.4. Technical Indicators and the Efficient Market Hypothesis

The inclusion of technical indicators in RL agents provides mixed results, presenting a challenge to the EMH, which posits that all available information is reflected in price alone. While the EMH suggests that additional data such as technical indicators should offer no advantage, the empirical results demonstrate that, in some cases, these indicators can improve decision-making.

For instance, in the case of the *DDPG-MLP* agent, using technical indicators with a 28-period lookback and continuous rebalancing led to an annualized return of 19.2%, accompanied by a Sharpe ratio of 0.72 and a Sortino ratio of 0.99. In contrast, the same agent relying solely on OHLC data achieved a lower annualized return of 18.1%, with a Sharpe ratio of 0.66 and a Sortino ratio of 0.88. This suggests that the additional information from technical indicators enabled more efficient asset allocation. Similarly, the *DQN-CNN* agent with technical indicators outperformed its OHLC-only counterpart, yielding an annualized return of 25.9%, a Sharpe ratio of 0.80, and a Sortino ratio of 1.12, compared to 25.8%, 0.74, and 1.02 for the OHLC configuration.

However, in some cases, technical indicators did not provide a significant advantage. For example, the *PPO-CNN-16* agent showed little difference between configurations, with the technical indicator version achieving an annualized return of 18.5% and a Sharpe ratio of 0.72, versus 18.0% and 0.71, respectively, for the OHLC-only version. This suggests that, while technical indicators can provide added insight in certain scenarios, their utility is not universally applicable across all agent configurations and market conditions.

In conclusion, technical indicators can enhance an agent's ability to identify market trends and manage risk in specific configurations, but their effectiveness is contingent on factors such as the agent architecture, the complexity of the market environment, and the length of the lookback period. These findings do not fully contradict the EMH but indicate that under certain conditions, technical indicators may offer incremental value.

4.5. Agent Portfolios and the Efficient Frontier

Figure 6 shows that the top-performing RL agent portfolios, in terms of annualized return, exhibit higher levels of volatility, underscoring the inherent trade-off between risk and reward. Many of these RL agent portfolios are positioned near the efficient frontier, particularly the highest performers, indicating that these agents have successfully constructed portfolios that approach optimality within the risk–return framework.

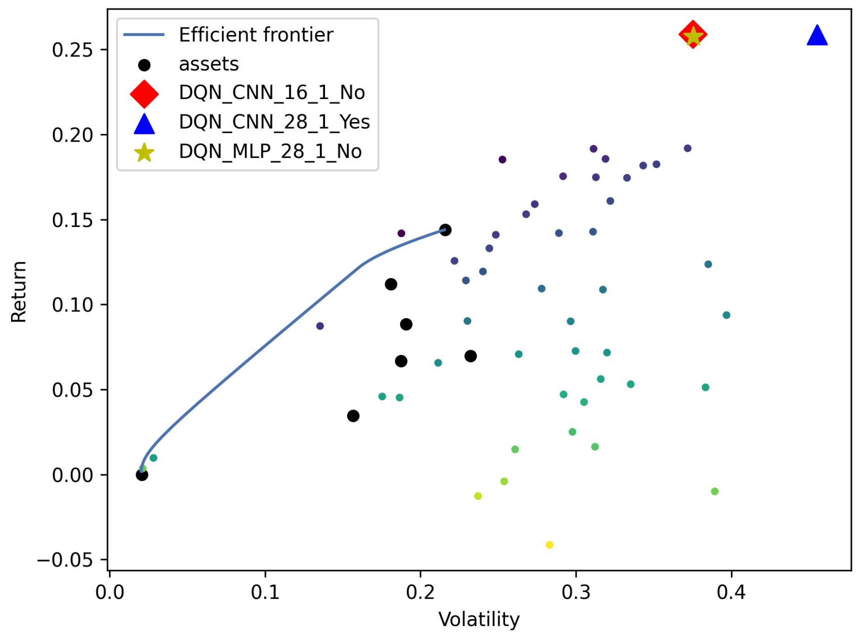

Figure 6. Efficient frontier and RL agent portfolios.

It is important to note, however, that RL agent portfolios are not bound by the efficient frontier. The efficient frontier assumes static mean returns and covariances for assets, which is a simplified view of financial markets. In contrast, RL agents dynamically adapt to changing market conditions, potentially enabling them to achieve superior performance in specific environments. This adaptability allows RL agents to optimize portfolios in ways that traditional mean-variance optimization may overlook, demonstrating the flexibility and potential of reinforcement learning in portfolio management.

4.6. Limitations

This study presents several limitations that could impact the generalizability of the results. Firstly, the technical indicators and hyperparameters used were selected based on prior knowledge rather than a systematic optimization process. As a result, it is possible that alternative configurations could yield improved outcomes. The chosen set of indicators may not represent the optimal solution across all market conditions, potentially limiting the performance of the models in different environments.

Moreover, the reliance on daily data based on market close and open prices may not fully capture the real-world execution prices faced by traders. Price fluctuations between the close and the following day's open can introduce discrepancies, leading to inaccuracies in the simulation. Although the inclusion of slippage and transaction costs mitigates this effect to some extent, alternative methods of preparing bar data, such as those proposed by [39], may offer a more accurate reflection of trading conditions.

Additionally, this study assumes that financial instruments are both liquid and fractional, simplifying the complexities of actual trading environments. In reality, liquidity constraints and the need to trade in discrete units may affect portfolio adjustments, particularly in markets with limited liquidity or where fractional shares are not available. These assumptions may limit the applicability of the results to real-world scenarios, where trading constraints can significantly impact portfolio rebalancing and execution.

4.7. Future Research Directions

Future research could explore the use of recurrent feature extractors, such as LSTM networks, to build on the positive results observed with CNNs in this study. Given that LSTMs have been shown to effectively capture temporal dependencies in financial data [5], experimenting with recurrent architectures could further improve the agents' ability to anticipate market trends. In addition, refining CNN configurations by experimenting with parameters such as kernel size and depth may enhance their capacity to identify and respond to localized market patterns.

Another promising direction is to investigate the use of autoregressive models based on Transformer architectures. The encoder–decoder framework of Transformers, originally developed for sequence modeling, could allow agents to better capture long-term dependencies in financial time series. By leveraging these architectures, future studies could improve the ability of agents to model complex, multi-step market trends, potentially leading to more accurate and robust portfolio optimization strategies.

Lastly, adjusting the reward function to prioritize risk-adjusted returns by incorporating metrics such as the Sharpe or Sortino ratios could further enhance the performance of RL agents. By focusing on a balance between risk and return, agents could be better equipped to manage volatility and achieve more stable, real-world investment outcomes. Exploring these alternative reward functions may lead to the development of more robust and risk-aware RL-based portfolio optimization strategies.

5. Conclusions

This study has demonstrated the potential of deep reinforcement learning (RL) agents in portfolio optimization by evaluating various configurations of agents, feature extractors, and rebalancing frequencies. The findings indicate that DQN and DDPG agents generally outperform traditional baselines, such as the market portfolio (S&P 500) and the equally weighted portfolio, in terms of both annualized returns and risk-adjusted performance, with the best portfolios providing an additional yearly return of 10% compared to the market portfolio and 17% compared to the equally weighted portfolio. Notably, CNN-based feature extractors, particularly with longer lookback periods, were more effective at identifying market patterns and adapting to volatile conditions, yielding superior Sharpe and Sortino ratios compared to MLP-based agents.

The analysis of rebalancing strategies revealed that while continuous rebalancing can capture short-term opportunities, it often incurs higher transaction costs and slippage. As a result, periodic rebalancing (e.g., every 10 periods) emerged as a more efficient strategy for balancing risk and managing costs. Furthermore, the inclusion of technical indicators provided marginal improvements in certain configurations, suggesting that while they may enhance the agent's understanding of market dynamics, they do not consistently outperform raw OHLC data.

These findings underscore the dynamic adaptability of RL agents in real-time market conditions, moving beyond the static assumptions of traditional portfolio optimization models like the efficient frontier. RL agents offer robust portfolio strategies that adjust dynamically to shifting market environments. Their strong performance, particularly in high-volatility settings, highlights their potential as valuable tools for portfolio management. Nonetheless, challenges remain, including the optimization of hyperparameters, the reduction in transaction costs, and the exploration of advanced architectures such as Transformers, offering promising directions for future research.

Author Contributions: Conceptualization, F.E.-F., Á.G.-S. and J.O.-M.; methodology, F.E.-F., Á.G.-S. and J.O.-M.; software, F.E.-F.; validation, Á.G.-S. and J.O.-M.; formal analysis, F.E.-F., Á.G.-S. and J.O.-M.; investigation, F.E.-F.; resources, F.E.-F. and J.O.-M.; data curation, F.E.-F.; writing—original draft preparation, F.E.-F.; writing—review and editing, F.E.-F., Á.G.-S. and J.O.-M.; visualization, F.E.-F.; supervision, Á.G.-S. and J.O.-M.; project administration, Á.G.-S. and J.O.-M. All authors have read and agreed to the published version of the manuscript.

Funding: The authors want to thank the Spanish Agencia Estatal de Investigación because of this research has been partially supported by the Ministerio de Ciencia e Innovación of Spain (Grant Ref. PID2022-137748OB-C31 funded by MCIN/AEI/10.13039/501100011033) and "ERDF A way of making Europe".

Data Availability Statement: Data are available using the *yfinance* library for the OHLC prices and *TALIB* to compute the technical indicators. All technical indicator parameters have been kept as the library defaults.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RL	Reinforcement learning
MLP	Multi-layer perceptron
CNN	Convolutional Neural Network
EMH	Efficient Market Hypothesis
DQN	Deep Q-Network
DDPG	Deep Deterministic Policy Gradient
PPO	Proximal Policy Optimization
SAC	Soft Actor–Critic
RSI	Relative Strength Index
NATR	Normalized Average True Range
AROONOSC	Aroon Oscillator
OHLC	Open-High-Low-Close
CCI	Commodity Channel Index
CMO	Chande Momentum Oscillator
MFI	Money Flow Index
QP	Quadratic Programming
GA	Genetic Algorithm
MDP	Markov Decision Process

Appendix A. RL Agents

Appendix A.1. DQN

The DQN [24] extends classical Q-learning by employing deep neural networks to approximate the Q-value function. The DQN estimates the action-value function $Q(s,a)$, which represents the expected cumulative future reward for taking action a in state s, and follows the Bellman [48] equation to update Q-values iteratively. A key feature of the DQN is its use of experience replay and target networks to stabilize training by breaking the correlation between samples and reducing Q-value variance.

The Q-learning update in the DQN is defined by

$$y_t^{DQN} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta^-) \tag{A1}$$

where θ^- represents the parameters of the target network, which are periodically updated from the main Q-network.

While the DQN can achieve human-level performance in many environments, especially in discrete action spaces, it suffers from overestimation bias due to the maximization step over noisy Q-value estimates. This overestimation can lead to suboptimal policies, especially in environments with high variance or noise in the reward signals.

Double Q-learning [38], originally proposed to address the overestimation bias in classical Q-learning, was later adapted to the DQN. In the Double DQN, the key idea is to decouple the action selection from the action evaluation to reduce the overestimation bias in the Q-values. Instead of using the same Q-values for both selecting and evaluating actions, the Double DQN selects actions using the main Q-network and evaluates them using the target Q-network. This modification reduces overestimation and improves stability. The update rule in the Double DQN is modified as follows:

$$y_t^{DoubleDQN} = R_{t+1} + \gamma Q(S_{t+1}, \arg\max_a Q(S_{t+1}, a; \theta), \theta^-) \tag{A2}$$

Here, the action is selected based on the online network parameters θ, while the evaluation is performed using the target network θ^-.

In comparison to the vanilla DQN, the Double DQN reduces overestimation substantially, improving policy evaluation and performance, especially in noisy environments.

The Dueling DQN [49] architecture further improves upon the standard DQN by separating the estimation of the state-value function $V(s)$ and the advantage function $A(s,a)$, which measures the relative benefit of an action in a given state compared to the average action. The key advantage of this separation is that the value function can be learned more effectively, even in states where the choice of action has little impact on the reward.

The Q-value function is decomposed as

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s,a';\theta,\alpha) \right) \tag{A3}$$

This decomposition allows the agent to learn state values without needing to evaluate the impact of every possible action. This is particularly useful in environments where many actions lead to similar outcomes. The Dueling DQN often outperforms the standard DQN and Double DQN in terms of both learning speed and final performance.

Appendix A.2. DDPG

DDPG [25] is an actor–critic, model-free reinforcement learning algorithm designed for continuous action spaces. It extends the DQN's [24] capabilities to continuous actions, which are typical in real-world systems. Unlike the DQN, which is value-based and suitable for discrete actions, DDPG employs a deterministic policy gradient approach.

The algorithm uses two networks: an actor, which learns a policy that maps states to actions, and a critic, which evaluates the action-value function $Q(s,a)$. The critic's objective is to maximize the expected return, while the actor optimizes the policy based on feedback from the critic. The *critic* learns the action-value function $Q(s,a)$ using the Bellman equation:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}[r_t + \gamma Q^{\pi}(s_{t+1}, \mu(s_{t+1}))] \tag{A4}$$

The *actor* learns a deterministic policy $\mu(s)$, which directly maps states to actions: $\mu(s|\theta^{\mu}) = a$. The actor is trained by following the policy gradient:

$$\nabla_{\theta^{\mu}} J \approx \mathbb{E}\Big[\nabla_a Q(s,a|\theta^{Q}) \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})\Big] \tag{A5}$$

Similar to the DQN, DDPG employs a *replay buffer* to break the correlation between consecutive transitions, enhancing training stability and efficiency. Both the actor and critic networks have corresponding target networks, Q' and μ', which are slowly updated to provide stable target values during training:

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta' \tag{A6}$$

Since the actor outputs deterministic actions, exploration is induced by adding noise to the action selection process. DDPG typically uses an *Ornstein–Uhlenbeck* noise process for temporally correlated exploration; this noise helps in exploring continuous action spaces more effectively than simple uncorrelated Gaussian noise.

$$a_t = \mu(s_t|\theta^{\mu}) + \mathcal{N}_t \tag{A7}$$

DDPG's key advantage is its efficiency in continuous action spaces, directly outputting continuous actions without requiring discretization. This makes it ideal for tasks requiring precise control, such as real-time portfolio weight adjustments. Additionally, DDPG incorporates target networks for both the actor and critic, which significantly improves the stability of learning, especially in environments with high variance. Another strength of DDPG is that it is an off-policy algorithm, meaning it can learn from past experiences stored in the replay buffer, thus making better use of data and reducing the variance in policy updates.

However, DDPG also presents some disadvantages, especially in terms of exploration. Since it outputs deterministic actions, DDPG relies heavily on adding noise to ensure sufficient exploration. If the exploration is not well tuned, this can lead to suboptimal policies, particularly in highly stochastic environments such as financial markets with unpredictable price movements. Moreover, DDPG is sensitive to hyperparameter settings, such as the learning rates for the actor and critic, the type of noise process used, and the size of the replay buffer. Poor tuning of these parameters can result in slow learning or, in some cases, divergence of the model.

Appendix A.3. PPO

PPO [26] is a policy gradient method that improves the data efficiency and stability of traditional policy gradient algorithms. It belongs to the family of actor–critic methods, where the actor updates the policy while the critic evaluates it. PPO is designed to address some of the challenges faced by Trust Region Policy Optimization (TRPO), such as complexity and inefficiency in handling large models. PPO introduces a simpler objective with a clipped probability ratio, keeping updates within a safe range, avoiding excessively large updates that can destabilize learning. This method alternates between data collection via environmental interaction and multiple epochs of policy optimization using the collected data.

The core update in PPO uses a *clipped surrogate objective*, which is designed to limit policy updates without requiring the complexity of TRPO's constraints. The surrogate objective is defined as

$$L^{CLIP}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \quad \text{(A8)}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the ratio of the new to old policy probabilities. $\hat{A}_t$ is the estimated advantage at time t, and ϵ is a small hyperparameter, typically set to 0.1 or 0.2, which controls the range of allowed updates.

PPO limits the change in policy by clipping $r_t(\theta)$, preventing large deviations from the previous policy during optimization. This balances exploration and exploitation without the need for complex constraints like in TRPO.

One of the key advantages of PPO is that it is easier to implement compared to TRPO, requiring fewer computational resources while still providing similar stability. The clipped objective prevents overly aggressive policy updates, ensuring that learning remains smooth and stable even during multiple epochs of optimization. This makes PPO particularly robust across different environments and tasks, requiring less hyperparameter tuning compared to other policy gradient methods. Moreover, the ability to perform multiple updates using the same data improves sample efficiency, a critical factor in environments where interactions are expensive, such as in portfolio optimization.

Despite its robustness, PPO is not immune to certain drawbacks. The algorithm's performance can be sensitive to the choice of clipping parameter ϵ, which requires careful tuning. If the clipping threshold is too tight, the algorithm may fail to make meaningful policy updates, slowing down learning. On the other hand, if the clipping is too loose, it can lead to unstable updates similar to standard policy gradient methods. Additionally, PPO, while more efficient than TRPO, may still be outperformed by algorithms specifically designed for high-dimensional continuous control tasks, such as SAC.

Appendix A.4. SAC

SAC [27] is an off-policy, actor–critic algorithm based on the maximum entropy reinforcement learning framework. SAC aims to maximize both the expected return and the entropy of the policy. The inclusion of entropy in the objective encourages exploration and prevents premature convergence to suboptimal policies by promoting stochasticity in action selection. SAC is particularly well suited for continuous control tasks due to its ability to handle high-dimensional action spaces efficiently. It combines stable, sample-efficient learning with strong exploration capabilities, making it a robust choice for real-world problems like portfolio optimization, where market conditions can be highly dynamic and uncertain.

The core of SAC revolves around maximizing a modified objective function that balances reward maximization and policy entropy. The objective is defined as

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))] \quad \text{(A9)}$$

where $\mathcal{H}(\pi(\cdot|s_t))$ is the entropy of the policy at state s_t, and α is a temperature parameter that controls the trade-off between reward maximization and entropy maximization. SAC maintains two Q-value functions $Q(s,a)$, a value function $V(s)$, and a policy $\pi(a|s)$. The update for each function is based on the following.

The Q-function is updated using soft Bellman backups:

$$Q_\theta(s_t,a_t) = r(s_t,a_t) + \gamma\mathbb{E}_{s_{t+1}\sim p}[V_\psi(s_{t+1})] \quad \text{(A10)}$$

The value function update is

$$V_\psi(s_t) = \mathbb{E}_{a_t \sim \pi}[Q_\theta(s_t, a_t) - \log \pi(a_t|s_t)] \tag{A11}$$

And, lastly, the policy is updated by minimizing the KL divergence between the policy and a Boltzmann distribution of the Q-function:

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim D}\left[\mathbb{E}_{a_t \sim \pi_\phi}\left[\alpha \log \pi_\phi(a_t|s_t) - Q_\theta(s_t, a_t)\right]\right] \tag{A12}$$

SAC's most significant advantage is its ability to balance exploration and exploitation through the entropy term. By promoting exploration, SAC avoids the risk of prematurely converging to suboptimal policies, a common issue in deterministic approaches like DDPG. SAC is also off-policy, meaning it can reuse past experiences stored in a replay buffer, making it sample-efficient compared to on-policy methods. This efficiency is critical in data-scarce environments like financial markets. Furthermore, SAC leverages a stochastic policy, which allows for better exploration and smoother learning compared to algorithms that rely on deterministic policies.

SAC's reliance on entropy maximization introduces a trade-off: while promoting exploration, the algorithm can occasionally over-explore, leading to slower convergence, especially if the temperature parameter α is not well tuned. Additionally, SAC can be more computationally intensive than simpler policy gradient methods due to the need for maintaining and updating multiple value functions and a policy network. The performance of SAC is also sensitive to the choice of hyperparameters, particularly the temperature α, which must be carefully adjusted for each environment to achieve an appropriate balance between exploration and exploitation.

Appendix A.5. Agent Comparison in the Context of Portfolio Optimization

The Double DQN mitigates this overestimation bias by decoupling action selection from evaluation, providing more reliable value estimates in noisy environments such as financial markets. This modification enhances the accuracy of portfolio allocations, making it a more stable choice for optimizing portfolios in unpredictable market conditions [38,50].

The Dueling DQN architecture further improves performance by separating the value of a state from the advantage of specific actions. In portfolio management, where long-term strategic decisions are often more important than short-term fluctuations, this architecture allows for more effective state evaluation, facilitating the development of robust asset allocation strategies even when the impact of individual actions may be minimal [49,51].

DDPG excels in continuous action spaces, directly outputting actions through the actor network. This makes it particularly well suited for real-time decision-making in portfolio optimization, where continuous and precise adjustments to asset allocations are required. DDPG's ability to optimize portfolio weights continuously allows it to respond dynamically to evolving market conditions, providing a more flexible approach to maximizing returns or minimizing risk. However, the algorithm's reliance on noise for exploration necessitates careful tuning to avoid premature convergence to suboptimal strategies [23,25].

PPO offers a versatile solution for portfolio optimization by handling both discrete and continuous actions effectively. Its clipped objective function ensures that updates to portfolio weights remain stable, thereby reducing the risk of abrupt or erratic changes in asset allocation. This stability is crucial in financial environments, where large shifts in portfolio composition can incur significant transaction costs or expose the portfolio to increased risk. Additionally, PPO's improved sample efficiency enhances its performance in data-constrained environments, making it well suited for real-world financial markets, where access to frequent, high-quality data is often limited [3,26].

Finally, SAC stands out for its ability to manage continuous action spaces while incorporating an entropy term to encourage exploration. This feature enables SAC to explore a wide range of portfolio strategies, which is particularly valuable in highly volatile or uncertain market conditions. By balancing risk and return more effectively, SAC is able to uncover robust strategies that perform well across a variety of market scenarios. Its off-policy nature further enhances its efficiency by allowing the algorithm to learn from historical market data, making it particularly advantageous in environments with limited access to real-time data [27,32].

Appendix B. Market Behaviors and Technical Indicators Computation

Technical indicators are used to detect or anticipate behaviors in a financial asset. Below is a non-exhaustive list of the behaviors that technical indicators try to capture:

- Bullish: This term refers to a positive outlook on an asset or market, where traders or investors expect prices to rise. Bullish sentiment often leads to increased buying activity. Bullish signals in technical analysis include uptrends, bullish crossovers, and patterns that suggest continued upward momentum or a reversal from a previous downtrend.
- Bearish: This term describes a negative outlook on an asset or market, where traders or investors expect prices to fall. Bearish sentiment typically leads to increased selling activity. In technical analysis, bearish signals include downtrends, bearish crossovers, and patterns indicating a decline in momentum or the possibility of a trend reversal to the downside.
- Overbought Level: An overbought level occurs when a technical indicator suggests that an asset has experienced a prolonged upward price movement, pushing its value higher than what might be considered sustainable or justified by fundamentals. When an asset is overbought, it implies that buying pressure may be excessive, and a price correction or pullback could be imminent.
- Oversold Level: An oversold level indicates that an asset has experienced a significant and extended downward price movement, possibly pushing its price below its intrinsic value. When an asset is oversold, it may signal a buying opportunity, as the selling pressure could be overdone, leading to a potential price rebound.
- Crossover: A crossover occurs when one technical indicator crosses above or below another indicator or a specific level, triggering a potential buy or sell signal. Common examples include a moving-average crossover, where a short-term moving average crosses above (bullish signal) or below (bearish signal) a long-term moving average, or an oscillator like the Stochastic Oscillator crosses certain thresholds (e.g., 20 or 80).
- Divergence: Divergence happens when the price of an asset moves in the opposite direction of a technical indicator, suggesting a possible shift in the asset's momentum or trend. Bullish divergence occurs when the price makes lower lows, but the indicator makes higher lows, potentially signaling a reversal to the upside. Bearish divergence happens when the price makes higher highs, but the indicator makes lower highs, indicating a potential downward reversal.
- Momentum: Momentum measures the speed and strength of price movements in a given direction. A strong momentum suggests that an asset's price is likely to continue moving in the same direction, either up or down.
- Trend Confirmation: Trend confirmation refers to using additional technical indicators or signals to validate the existence and strength of a market trend, whether up or down. Confirming trends helps traders avoid false signals or premature entries. For example, a trend may be confirmed when both price and indicators like moving averages or momentum oscillators are aligned in the same direction.
- Trend Reversal: A trend reversal occurs when the direction of a prevailing market trend changes, either from an uptrend to a downtrend or vice versa. Reversals are identified by various technical patterns, such as head and shoulders, or through divergence between price and momentum indicators. Recognizing trend reversals is crucial for traders looking to capitalize on the shift in market direction.

Appendix B.1. Normalized Average True Range

The Normalized Average True Range (NATR) is a volatility indicator derived from the Average True Range (ATR) [52] but normalized by dividing the ATR by the closing price at time t. It provides a relative measure of volatility by accounting for the price level of the security, making it easier to compare volatility across different assets with varying price ranges.

The NATR is based on the ATR, which measures the range of price movement for a given period and normalizes it using the closing price. The TR at time t is

$$TR = \max(\text{High}_t - \text{Low}_t, |\text{High}_t - \text{Close}_{t-1}|, |\text{Low}_t - \text{Close}_{t-1}|) \tag{A13}$$

The ATR is the moving average of the True Range over N periods (typically 14):

$$ATR = \text{SMA}_N(TR) \tag{A14}$$

The NATR is calculated by dividing the ATR by the current closing price:

$$NATR = \frac{ATR}{\text{Close}_t} \times 100 \tag{A15}$$

Appendix B.2. Aroon Oscillator

The Aroon Oscillator (AROONOSC) [53] is a technical indicator used to identify the strength and direction of a trend. It is derived from the Aroon Indicator, which measures the time since the highest high and the lowest low over a given period. The oscillator provides a single value that oscillates between -100 and 100.

The AROONOSC is calculated as the difference between the Aroon-Up and Aroon-Down indicators:

$$\begin{aligned} \text{Aroon-Up} &= \frac{N - \text{Number of periods since highest high}}{N} \times 100 \\ \text{Aroon-Down} &= \frac{N - \text{Number of periods since lowest low}}{N} \times 100 \\ \text{AROONOSC} &= \text{Aroon-Up} - \text{Aroon-Down} \end{aligned} \tag{A16}$$

Appendix B.3. Relative Strength Index

The Relative Strength Index (RSI) [52] is a momentum oscillator used to measure the speed of and change in price movements. It oscillates between 0 and 100. Values above 70 typically suggest that a security is overbought, while values below 30 indicate it is oversold.

The RSI is calculated by first computing the Relative Strength (RS):

$$\begin{aligned} RS &= \frac{\text{Average Gain over } n \text{ periods}}{\text{Average Loss over } n \text{ periods}} \\ RSI &= 100 - \left(\frac{100}{1 + RS}\right) \end{aligned} \tag{A17}$$

Appendix B.4. Commodity Channel Index

The Commodity Channel Index (CCI) [54] is a momentum-based oscillator that measures the deviation of a security's price from its statistical average. The CCI is calculated by comparing the current price to the moving average of the price over a specified period, adjusted by a mean deviation.

$$CCI = \frac{\text{Typical Price} - \text{SMA of Typical Price}}{0.015 \times \text{Mean Deviation}} \tag{A18}$$

Appendix B.5. Chande Momentum Oscillator

The Chande Momentum Oscillator (CMO) [55] measures the momentum of price movements by calculating the difference between recent gains and losses over a specified period. The oscillator moves between −100 and +100.

The CMO is calculated as

$$CMO = \frac{\sum(\text{Gains}) - \sum(\text{Losses})}{\sum(\text{Gains}) + \sum(\text{Losses})} \times 100 \tag{A19}$$

Appendix B.6. Money Flow Index

The Money Flow Index (MFI) [56] is a momentum indicator that uses both price and volume to measure buying and selling pressure. It oscillates between 0 and 100. An MFI above 80 generally indicates an overbought condition, while an MFI below 20 signals an oversold condition.

The MFI is calculated through

$$\begin{gathered} TP = \frac{\text{High} + \text{Low} + \text{Close}}{3} \\ RMF = TP \times \text{Volume} \end{gathered} \tag{A20}$$

$$MFR = \frac{\sum(\text{Positive Money Flow})}{\sum(\text{Negative Money Flow})} \tag{A21}$$

$$MFI = 100 - \left(\frac{100}{1 + MFR}\right) \tag{A22}$$

When the price increases in consecutive periods, the Raw Money Flow (RMF) is positive and added to the Positive Money Flow. When it decreases, the RMF is negative and added to the Negative Money Flow.

Appendix B.7. Williams %R

Williams %R [57] compares the current closing price to the highest high over a specific period, typically 14 periods, and is expressed on a scale from 0 to −100.

$$\%R = \frac{\text{Highest High} - \text{Current Close}}{\text{Highest High} - \text{Lowest Low}} \times -100 \tag{A23}$$

Appendix B.8. Stochastic Oscillator

The Stochastic Oscillator (STOCHF) [58] is a momentum indicator that compares a security's closing price to its price range over a specified period. It ranges between 0 and 100. The %K and %D lines are calculated as follows:

$$\%K = \frac{\text{Current Close} - \text{Lowest Low}}{\text{Highest High} - \text{Lowest Low}} \times 100 \tag{A24}$$

$$\%D = \text{SMA}_3(\%K) \tag{A25}$$

Appendix C. RL Agent Portfolio Performance Metrics

Table A1 provides the metrics for all the RL agent portfolios, excluding the SAC experiments.

Table A1. Performance metrics for all RL agent configurations.

Ann. Return	Ann. Volatility	Calmar	Sharpe	Sortino	Max Drawdown	Agent	Feat.Ext.	Lookback	Reb. Freq.	Indicators
0.26	0.38	0.56	0.80	1.12	−0.47	DQN	CNN	16	1	No
0.26	0.46	0.45	0.74	1.02	−0.57	DQN	CNN	28	1	Yes
0.26	0.38	0.55	0.80	1.12	−0.47	DQN	MLP	28	1	No
0.19	0.37	0.34	0.66	0.91	−0.57	DDPG	MLP	16	10	No
0.19	0.31	0.39	0.72	1.00	−0.49	DDPG	MLP	28	1	Yes
0.19	0.32	0.37	0.70	0.96	−0.50	DDPG	CNN	28	10	Yes
0.19	0.25	0.55	0.80	1.18	−0.33	PPO	MLP	16	1	Yes
0.18	0.35	0.31	0.66	0.91	−0.59	DQN	MLP	16	1	No
0.18	0.34	0.50	0.66	0.95	−0.36	DQN	MLP	16	1	Yes
0.18	0.29	0.44	0.70	0.97	−0.39	DDPG	MLP	16	1	Yes
0.17	0.31	0.36	0.67	0.94	−0.49	DDPG	MLP	16	10	Yes
0.17	0.33	0.36	0.65	0.90	−0.49	DDPG	CNN	16	1	Yes
0.16	0.32	0.31	0.63	0.86	−0.52	DDPG	CNN	28	1	Yes
0.16	0.27	0.37	0.68	0.94	−0.42	DDPG	CNN	16	10	No
0.15	0.27	0.43	0.66	0.97	−0.36	PPO	MLP	28	10	Yes
0.14	0.31	0.28	0.59	0.81	−0.50	DDPG	MLP	16	1	No
0.14	0.29	0.28	0.61	0.84	−0.52	DDPG	CNN	28	10	No
0.14	0.19	0.57	0.80	1.12	−0.25	^GSPC	-	-	1	-
0.14	0.25	0.35	0.66	0.91	−0.40	DDPG	CNN	16	1	No
0.13	0.24	0.32	0.63	0.88	−0.42	DDPG	CNN	16	10	Yes
0.13	0.22	0.34	0.65	0.90	−0.37	DDPG	MLP	28	10	No
0.12	0.39	0.21	0.50	0.69	−0.59	DQN	MLP	28	1	Yes
0.12	0.24	0.39	0.59	0.89	−0.31	PPO	MLP	16	10	No
0.11	0.23	0.27	0.59	0.81	−0.42	DDPG	CNN	28	1	No
0.11	0.28	0.25	0.52	0.70	−0.43	PPO	CNN	28	1	Yes
0.11	0.32	0.25	0.49	0.68	−0.44	PPO	CNN	16	10	No
0.09	0.40	0.17	0.43	0.58	−0.54	PPO	CNN	28	10	No
0.09	0.23	0.24	0.49	0.72	−0.38	PPO	MLP	28	1	Yes
0.09	0.30	0.19	0.44	0.62	−0.46	PPO	MLP	28	10	No
0.09	0.14	0.35	0.69	0.94	−0.25	EWP	-	-	1	-
0.07	0.30	0.16	0.38	0.54	−0.46	DQN	MLP	28	10	No
0.07	0.32	0.13	0.38	0.52	−0.54	DQN	CNN	28	1	No
0.07	0.26	0.20	0.39	0.54	−0.35	PPO	CNN	16	10	Yes
0.07	0.21	0.17	0.41	0.57	−0.38	DDPG	MLP	28	10	Yes
0.06	0.32	0.10	0.33	0.46	−0.54	DQN	CNN	16	10	No
0.05	0.34	0.09	0.32	0.44	−0.59	DQN	MLP	16	10	Yes
0.05	0.38	0.11	0.32	0.45	−0.47	PPO	CNN	16	1	Yes
0.05	0.29	0.11	0.31	0.41	−0.44	PPO	CNN	28	1	No
0.05	0.18	0.13	0.35	0.46	−0.35	DQN	MLP	16	10	No
0.05	0.19	0.13	0.33	0.47	−0.34	DDPG	MLP	28	1	No
0.04	0.31	0.09	0.29	0.40	−0.50	PPO	MLP	16	10	Yes
0.03	0.30	0.05	0.23	0.32	−0.52	DQN	CNN	28	10	No
0.02	0.31	0.02	0.21	0.28	−0.69	PPO	CNN	16	1	No
0.01	0.26	0.02	0.19	0.25	−0.59	DQN	MLP	28	10	Yes
0.01	0.03	0.27	0.36	0.63	−0.04	DQN	CNN	28	10	Yes
0.00	0.02	0.11	0.18	0.25	−0.03	DQN	CNN	16	10	Yes
0.00	0.25	−0.01	0.12	0.14	−0.62	PPO	MLP	28	1	No
−0.01	0.39	−0.02	0.17	0.24	−0.51	DQN	CNN	16	1	Yes
−0.01	0.24	−0.02	0.07	0.09	−0.51	PPO	MLP	16	1	No
−0.04	0.28	−0.06	0.00	0.00	−0.66	PPO	CNN	28	10	Yes

Experimental results results by annualized return compared to the benchmark portfolios (in grey).

References

1. Markowitz, H. Portfolio Selection. *J. Financ.* **1952**, *7*, 77–91.
2. Benhamou, E.; Saltiel, D.; Ungari, S.; Mukhopadhyay, A. Bridging the gap between Markowitz planning and deep reinforcement learning. *arXiv* **2020**, arXiv:2010.09108. [CrossRef]
3. Halperin, I.; Liu, J.; Zhang, X. Combining Reinforcement Learning and Inverse Reinforcement Learning for Asset Allocation Recommendations. *arXiv* **2022**, arXiv:2201.01874. [CrossRef]
4. Markowitz, H. The optimization of a quadratic function subject to linear constraints. *Nav. Res. Logist. Q.* **1956**, *3*, 111–133. [CrossRef]
5. Benhamou, E.; Saltiel, D.; Ohana, J.; Atif, J.; Laraki, R. Deep Reinforcement Learning (DRL) for Portfolio Allocation. In *Machine Learning and Knowledge Discovery in Databases: Applied Data Science and Demo Track*; Springer: Cham, Switzerland, 2021; pp. 527–531. [CrossRef]
6. Merton, R.C.; Samuelson, P.A. Fallacy of the log-normal approximation to optimal portfolio decision-making over many periods. *J. Financ. Econ.* **1974**, *1*, 67–94. [CrossRef]
7. Odermatt, L.; Beqiraj, J.; Osterrieder, J. Deep Reinforcement Learning for Finance and the Efficient Market Hypothesis. *SSRN Electron. J.* **2021**. [CrossRef]
8. Li, G. Enhancing Portfolio Performances through LSTM and Covariance Shrinkage. *Adv. Econ. Manag. Political Sci.* **2023**, *26*, 187–198. [CrossRef]
9. Hochreiter, S.; Schmidhuber, J. Long Short-term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
10. Sen, J.; Dasgupta, S. Portfolio Optimization: A Comparative Study. *arXiv* **2023**, arXiv:2307.05048. [CrossRef]
11. Pun, C.S.; Wong, H.Y. Robust investment–reinsurance optimization with multiscale stochastic volatility. *Insur. Math. Econ.* **2015**, *62*, 245–256. [CrossRef]
12. Supandi, E.; Rosadi, D.; Abdurakhman, A. Improved robust portfolio optimization. *Malays. J. Math. Sci.* **2017**, *11*, 239–260.
13. Liu, X.Y.; Xiong, Z.; Zhong, S.; Yang, H.; Walid, A. Practical Deep Reinforcement Learning Approach for Stock Trading. *arXiv* **2022**, arXiv:1811.07522. [CrossRef]
14. Wang, Z.; Jin, S.; Li, W. Research on Portfolio Optimization Based on Deep Reinforcement Learning. In Proceedings of the 2022 4th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Shanghai, China, 28–30 October 2022; pp. 391–395. [CrossRef]
15. Yang, S. Deep reinforcement learning for portfolio management. *Knowl.-Based Syst.* **2023**, *278*, 110905. [CrossRef]
16. Wei, L.; Weiwei, Z. Research on Portfolio Optimization Models Using Deep Deterministic Policy Gradient. In Proceedings of the 2020 International Conference on Robots & Intelligent System (ICRIS), Sanya, China, 7–8 November 2020; pp. 698–701. [CrossRef]
17. Harnpadungkij, T.; Chaisangmongkon, W.; Phunchongharn, P. Risk-Sensitive Portfolio Management by using Distributional Reinforcement Learning. In Proceedings of the 2019 IEEE 10th International Conference on Awareness Science and Technology (iCAST), Morioka, Japan, 23–25 October 2019; pp. 1–6. [CrossRef]
18. Hakansson, N.H. Multi-Period Mean-Variance Analysis: Toward A General Theory of Portfolio Choice. *J. Financ.* **1971**, *26*, 857–884.
19. Sefiane, S.; Benbouziane, M. Portfolio Selection Using Genetic Algorithm. *J. Appl. Financ. Bank.* **2012**, *2*, 143–154.
20. Hochreiter, R. An Evolutionary Optimization Approach to Risk Parity Portfolio Selection. In *Applications of Evolutionary Computation*; Springer International Publishing: Cham, Switzerland, 2015; pp. 279–288. [CrossRef]
21. Cen, S.; Cheng, C.; Chen, Y.; Wei, Y.; Chi, Y. Fast Global Convergence of Natural Policy Gradient Methods with Entropy Regularization. *arXiv* **2021**, arXiv:2007.06558. [CrossRef]
22. LeCun, Y.; Bengio, Y. Convolutional networks for images, speech, and time series. In *The Handbook of Brain Theory and Neural Networks*; MIT Press: Cambridge, MA, USA, 1998; pp. 255–258.
23. Jiang, Z.; Xu, D.; Liang, J. A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem. *arXiv* **2017**, arXiv:1706.10059. [CrossRef]
24. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjel, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef]
25. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2019**, arXiv:1509.02971. [CrossRef]
26. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347. [CrossRef]
27. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *arXiv* **2018**, arXiv:1801.01290. [CrossRef]
28. Pendharkar, P.; Cusatis, P. Trading Financial Indices with Reinforcement Learning Agents. *Expert Syst. Appl.* **2018**, *103*, 1–13. [CrossRef]
29. Yu, P.; Lee, J.S.; Kulyatin, I.; Shi, Z.; Dasgupta, S. Model-based Deep Reinforcement Learning for Dynamic Portfolio Optimization. *arXiv* **2019**, arXiv:1901.08740. [CrossRef]
30. Lattimore, T.; Hutter, M. PAC Bounds for Discounted MDPs. *arXiv* **2012**, arXiv:1202.3890. [CrossRef]
31. Longstaff, F.; Schwartz, E. Valuing American Options by Simulation: A Simple Least-Squares Approach. *Rev. Financ. Stud.* **2001**, *14*, 113–147. [CrossRef]

32. Soleymani, F.; Paquet, E. Financial Portfolio Optimization with Online Deep Reinforcement Learning and Restricted Stacked Autoencoder—DeepBreath. *Expert Syst. Appl.* **2020**, *156*, 113456. [CrossRef]
33. Fama, E.F. Efficient Capital Markets: A Review of Theory and Empirical Work. *J. Financ.* **1970**, *2*, 383–417. [CrossRef]
34. Carver, R. *Systematic Trading: A Unique New Method for Designing Trading and Investing Systems*; EBL-Schweitzer, Harriman House: Hampshire, UK, 2015; p. 48.
35. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **1989**, *2*, 359–366. [CrossRef]
36. Shi, Z.; Wei, J.; Liang, Y. A Theoretical Analysis on Feature Learning in Neural Networks: Emergence from Inputs and Advantage over Fixed Features. *arXiv* **2022**, arXiv:2206.01717. [CrossRef]
37. Yang, G.; Hu, E.J. Feature Learning in Infinite-Width Neural Networks. *arXiv* **2022**, arXiv:2011.14522. [CrossRef]
38. van Hasselt, H.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-learning. *arXiv* **2015**, arXiv:1509.06461. [CrossRef]
39. de Prado, M.L. *Advances in Financial Machine Learning*, 1st ed.; Wiley Publishing: Hoboken, NJ, USA, 2018; Chapter 2.
40. Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *J. Mach. Learn. Res.* **2021**, *22*, 1–8.
41. Vigna, E. *On Time Consistency for Mean-Variance Portfolio Selection*; Carlo Alberto Notebooks 476; Collegio Carlo Alberto: Torino, Italy, 2016.
42. Espiga, F. Portfolio Optimization. 2024. Available online: https://figshare.com/collections/PORTFOLIO_OPTIMIZATION/7467934 (accessed on 8 December 2024).
43. Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; Zaremba, W. OpenAI Gym. *arXiv* **2016**, arXiv:1606.01540. [CrossRef]
44. Towers, M.; Terry, J.K.; Kwiatkowski, A.; Balis, J.U.; de Cola, G.; Deleu, T.; Goulão, M.; Kallinteris, A.; KG, A.; Krimmel, M.; et al. Gymnasium. 2023. Available online: https://zenodo.org/records/8127026 (accessed on 8 December 2024).
45. Sharpe, W. Mutual Fund Performance. *J. Bus.* **1965**, *39*, 119–138. [CrossRef]
46. Sortino, F.A.; van der Meer, R. Downside Risk. *J. Portf. Manag.* **1991**, *17*, 27–31. [CrossRef]
47. Young, T. Calmar ratio: A smoother tool. *Futures* **1991**, *20*, 40–41.
48. Bellman, R. *Dynamic Programming*; Dover Publications: Mineola, NY, USA, 1957.
49. Wang, Z.; Schaul, T.; Hessel, M.; van Hasselt, H.; Lanctot, M.; de Freitas, N. Dueling Network Architectures for Deep Reinforcement Learning. *arXiv* **2016**, arXiv:1511.06581. [CrossRef]
50. Fujimoto, S.; Hoof, H.; Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. *arXiv* **2018**, arXiv:1802.09477. [CrossRef]
51. Gao, Z.; Gao, Y.; Hu, Y.; Jiang, Z.; Su, J. Application of Deep Q-Network in Portfolio Management. *arXiv* **2020**, arXiv:2003.06365. [CrossRef]
52. Wilder, J. *New Concepts in Technical Trading Systems*; Trend Research: Abu Dhabi, United Arab Emirates, 1978; pp. 21–23. 63–70.
53. Chande, T. The Time Price Oscillator. 1995. Available online: https://store.traders.com/-v13-c09-thetime-pdf.html?srsltid=AfmBOooJTN6RVoFN_XXqhHegrxp0fXdJP3cTxfuPJ2j_4Uw2-oHy9SAG (accessed on 8 December 2024).
54. Lambert, D. Commodity Channel Index: Tool for Trading Cyclic Trends. 1982. Available online: https://store.traders.com/-v01-c05-comm-pdf.html?srsltid=AfmBOoqNzNIaJtdSg7OwR3FazvfXfbfpjZQzcRaXYCvwpPH98gGv9B2M (accessed on 8 December 2024).
55. Chande, T.; Kroll, S. *The New Technical Trader: Boost Your Profit by Plugging into the Latest Indicators*; Wiley Finance, Wiley: Hoboken, NJ, USA, 1994; pp. 94–118.
56. Quong, G.; Soudack, A. Volume-Weighted RSI: Money Flow. 1989. Available online: https://store.traders.com/-v07-c03-volumew-pdf.html?srsltid=AfmBOortDQy0rOdmnMXX-QVdXorrh_q0h4DZt3HgP-8seBHmcA76z8CV (accessed on 8 December 2024).
57. Williams, L. *How I Made One Million Dollars… Last Year… Trading Commodities*; Windsor Books: Brightwaters, NY, USA, 1979; pp. 94–118.
58. Lane, G.C. Lane's Stochastics. 1984. Available online: https://store.traders.com/-v02-c03-lane-pdf.html?srsltid=AfmBOoqpO80_Wrs0Lx1ezAu1T3OEIb8X0Ztkx5OJsakotBnc-Lm4sLqg (accessed on 8 December 2024).