Benchmarking Reinforcement Learning (RL) Algorithms for Portfolio Optimization

Preprint · August 2024

DOI: 10.13140/RG.2.2.20482.70085

2 authors:

Yassine Hachaïchi
University of Carthage
53 PUBLICATIONS 81 CITATIONS
SEE PROFILE

Amir Lanwer
Tunisia Polytechnic School
1 PUBLICATION 0 CITATIONS
SEE PROFILE

Benchmarking Reinforcement Learning (RL) Algorithms for Portfolio Optimization

Yassine Hachaïchi
EI&TIC Lab
ENICarthage, Carthage University
TUNISIA
https://orcid.org/0000-0002-3282-0877
Amir Lanwer
LEGI
Ecole polytechnique de Tunisie
TUNISIA
Amir.Lanwer@ept.ucar.tn

Introduction

The field of finance has seen significant advancements in recent years, with the introduction of new technologies and techniques for portfolio management. Portfolio managers play a crucial role in this process, as they are responsible for managing risk and seeking returns on behalf of their clients.

One technique that has shown promise in helping portfolio managers achieve these goals is reinforcement learning (RL). RL is a type of machine learning that focuses on training agents to make a sequence of decisions in an environment to achieve a goal. In the context of finance, RL algorithms can be used to make decisions about portfolio allocation based on market data and other relevant information.

The use of RL in portfolio management finance has been explored in various studies, with promising results. RL algorithms have been shown to be effective in optimizing portfolio allocation, achieving higher returns than traditional methods such as the Markowitz mean-variance portfolio (MVP). In addition, RL has been applied to other areas of finance, such as algorithmic trading and risk management.

In this report, we explore the use of RL algorithms, including DDPG, PPO, A2C, SAC, and TD3, for portfolio management in three different markets: CAC40, DJ30, and DAX30. These markets were chosen for their diversity and representativeness of different regions and economies. We compare the performance of these algorithms to traditional portfolio optimization techniques such as the MVP and the market index in each market. Our goal is to determine whether RL can outperform these benchmarks and to examine our results in light of the Efficient Market Hypothesis (EMH), which proposes that asset prices reflect all available information at any given time.

Portfolio optimization

In this section, we delve into the fascinating world of portfolio optimization, a crucial topic in finance that involves the selection of the best combination of assets to include in a portfolio in order to maximize returns while minimizing risk. We begin by exploring the basic concepts of asset return and asset volatility, which provide insight into the profitability and riskiness of individual assets. We then move on to discuss portfolio features and metrics such as portfolio return, portfolio variance, covariance matrix, and portfolio volatility, which are important for understanding the overall performance and risk of a portfolio.

Next, we introduce Modern Portfolio Theory (MPT), a widely utilized model for optimizing investment portfolios. MPT offers a mathematical framework where investors select portfolios based on risk and return. We discuss the key assumptions of MPT and explain important concepts such as the Efficient Frontier and Mean Variance Optimization.

Finally, we examine the Minimum Variance Portfolio (MVP), a portfolio that lies on the efficient frontier and has the lowest possible variance.

1. Basic Concepts: Asset return and Asset volatility:

1.1. Asset return:

Asset return refers to the financial gain or loss experienced by an investor from holding and trading an investment instrument, such as stocks, bonds, ETFs, or futures. It is calculated by considering changes in the asset's price over a specific period, offering insight into the profitability of the investment and its overall performance.

$$r_i = \frac{P_{t+1,i} - P_{t,i}}{P_{t,i}} \times 100$$

r_i : Asset return for asset i.
$P_{t+1,i}$: Close price of asset i at time t+1.
$P_{t,i}$: Close price of asset i at time t.

1.2. Asset volatility:

Asset volatility refers to the extent of fluctuation or variance in the returns of an individual investment instrument, such as stocks, bonds, or other financial assets. It quantifies how much the returns of the asset deviate from their average or expected value. A higher asset volatility indicates that the asset's returns show more significant deviations over time, implying both greater potential risks and rewards. In contrast, lower volatility suggests more stable and predictable returns.
Mathematically, asset volatility is commonly calculated using the standard deviation formula:

$$\sigma_i = \sqrt{\frac{1}{N-1}\sum_{t=1}^{N}(r_{i,t} - r_i)^2}$$

Where:
σ_i : Asset volatility for asset i.
N : Number of observations or time periods.
$r_{i,t}$: Return of asset i at time t.
r_i : Average or expected return of asset i.

2. Portfolio features and metrics:

2.1. Portfolio return:

Portfolio return is the overall financial gain or loss from a combination of multiple assets held within an investment portfolio. Investors rely primarily on the historical returns to draw expectations about the future performance, and hence predict the expected returns. The overall return of a portfolio can be computed by summing the individual returns of its assets, weighted according to their proportions within the portfolio. The formula for calculating the portfolio return is as follows:

$$r_P = \sum_{i=1}^{N} w_i \cdot r_i$$

Where:
r_P: Portfolio return.
N : Number of assets in the portfolio.
w_i: Weight of asset i in the portfolio.
r_i: Return of asset i.

2.2. Portfolio Variance:

Portfolio variance is a statistical measure that quantifies the degree of dispersion or variability in the returns of a portfolio of investments. It takes into account the individual variances of each asset within the portfolio as well as their correlations with each other. The formula for calculating portfolio variance is:

$$\sigma_P^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \cdot \sigma_{i,j}$$

Where:
$\sigma_{i,j} = \rho_{i,j} \cdot \sigma_i \cdot \sigma_j$
σ_P^2 : Portfolio variance.
N : Number of assets in the portfolio.

$\sigma_{i,j}$: Covariance between asset i and asset j.
$\rho_{i,j}$: Correlation coefficient between asset i and asset j.
σ_i : Standard deviation of asset i.
σ_j : Standard deviation of asset j.
This calculation captures both the individual risk of each asset and the relationships between them, providing insight into the overall risk of the portfolio.

2.3. Covariance matrix:

A covariance matrix (Ω) is a mathematical representation that captures the relationships between various assets within an investment portfolio. It provides insights into how these assets move together or diverge in terms of their returns. In this matrix, the diagonal elements signify the variances of individual assets, indicating their individual volatility. The off-diagonal elements represent the covariances between pairs of assets, reflecting how their returns co-move. This matrix is crucial for understanding the diversification potential and risk management within a portfolio.

$$\Omega = \begin{bmatrix} \sigma_1^2 & \sigma_{1,2} & \cdots & \sigma_{1,N} \\ \sigma_{2,1} & \sigma_2^2 & \cdots & \sigma_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{N,1} & \sigma_{N,2} & \cdots & \sigma_N^2 \end{bmatrix}$$

A positive covariance suggests that the assets making up the portfolio often move in a similar direction, while a negative correlation indicates that these assets tend to move in opposite directions. Generally, the lower the correlation among assets, the lower the overall volatility of the portfolio. This underlines why professional fund managers opt for diversified portfolios by selecting uncorrelated stocks. They recognize that the interplay between these assets significantly influences the overall riskiness of the portfolio.

2.4. Portfolio volatility:

Portfolio volatility, often referred to as the portfolio's risk, is measured by the value of its standard deviation. This is found by taking the square root of the variance. The terms standard deviation and variance are often used interchangeably to gauge risk.

$$\sigma_P = \sqrt{\sigma_P^2}$$

3. Modern portfolio theory:

Modern Portfolio Theory (MPT), introduced by Markowitz in 1952, is a widely utilized model for optimizing investment portfolios. It offers a mathematical framework where investors select portfolios based on risk and return. Markowitz's research involves computing the Minimum-Variance Frontier, a collection of portfolios that balance projected return and risk. This method utilizes portfolio variance and mean to establish the Efficient Frontier, a boundary that refines risk-return evaluation by categorizing portfolios into efficient and inefficient ones.
The expected return and variance of the portfolio are defined as follows:

$$E(r_P) = \mu^T w \qquad\qquad \sigma_P^2 = w^T \Omega w$$

μ represents the vector of expected returns of individual assets.

3.1. Assumptions:

Some of the key assumptions of MPT include:

- The investors are rational and risk averse.
- The investors have access to the same information.
- The investors base their decisions on expected return and variance.
- The market is frictionless, i.e., there are no taxes or transaction costs.
- All expected returns, variances and covariances of returns are known.

- We are not considering risk free assets.

3.2. The Efficient Frontier:

By leveraging the two notions of portfolio expected return and variance (risk), it becomes possible to create and visualize the mean-variance efficient portfolio for a desired expected return within the mean-variance frontier:

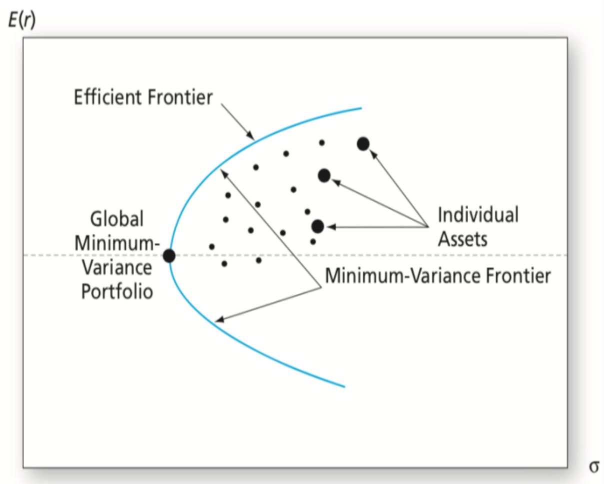


Figure 3.1: Mean-variance frontier when combining risky assets. (Bodie et al., 2014, p.220)

The portfolios situated along the minimum-variance frontier, starting from the global minimum-variance portfolio and extending upwards, offer optimal combinations of risk and return. As a result, the section of the frontier positioned above the global minimum-variance portfolio is recognized as the efficient frontier.

Figure 3.1 shows that all individual assets are positioned to the right of the efficient frontier. This observation signifies the presence of portfolios with equivalent risk but superior returns. Consequently, a portfolio consisting solely of a single risky asset is deemed inefficient.

On the other hand, for portfolios situated in the lower segment of the minimum-variance frontier, there exists a portfolio with the same risk but a higher expected return directly above it. Thus, the section of the frontier located below the Global minimum-variance portfolio is categorized as inefficient (Bodie et al. 2014).

3.3. Mean Variance Optimization:

The core principle of modern portfolio theory revolves around creating an optimal portfolio with a focus on minimizing variance. This is achieved while adhering to specific requirements: the portfolio weights summing up to one and the portfolio's return either matching or surpassing the target return (Markowitz, 1952). To address short-selling constraints, an additional optimization constraint is introduced, requiring that the weights of each asset within the portfolio remain positive. The procedure for constructing the optimal portfolio is articulated as a mathematical optimization problem:

$$\begin{array}{rl} \text{minimize} & \frac{1}{2} w^T \Omega w \\ \text{subject to} & \mu^T w \geq r_p \\ & \mathbf{1}^T w = 1 \\ & w \geq \mathbf{0} \end{array}$$

Here, w represents the portfolio weight vector and μ denotes the vector of expected returns for each asset in the portfolio. r_p signifies the target return, and 1 stands for a vector of ones. The covariance matrix for the asset returns in the portfolio is represented by **Ω**.

In summary, Modern Portfolio Theory (MPT) equips investors with tools to craft portfolios that align with their specific preferences. MPT enhances our comprehension of portfolio management, efficient portfolios, and the crucial notion of the minimum variance portfolio (MVP), which serves as a benchmark in this study.

3.4. Minimum Variance Portfolio:

The minimum variance portfolio (MVP) aims to decrease portfolio volatility by assigning weights to the least volatile assets. This portfolio is heuristic-based, not reliant on expected returns, and has the lowest risk among all mean-variance efficient portfolios. As a result, it resides at the far left end of the efficient frontier, as depicted in figure 3.1. The optimization setup can be expressed as follows:

$$\arg\min_{w}\left(\frac{1}{2}w^T\Omega w\right)$$

Reinforcement learning:

Reinforcement learning (RL) is a powerful approach to artificial intelligence that enables agents to learn from experience and make decisions in complex environments. In this section, we provide an overview of reinforcement learning, including its key components, approaches, and challenges. We begin by discussing the basics of machine learning and introducing the three main types of machine learning algorithms: supervised learning, unsupervised learning, and reinforcement learning.

We then delve into the details of reinforcement learning, covering topics such as the agent-environment interface, policies, rewards, value functions, and models. We also discuss Markov decision processes (MDPs), which provide a formal framework for modeling the interaction between a learning agent and its environment. Finally, we explore different approaches to reinforcement learning, including policy optimization, Q-learning, and mixed methods that combine elements from both classes. Throughout this section, we highlight also the challenges that arise in reinforcement learning.

1. Machine Learning: An Overview

Machine learning:

ML is a field of artificial intelligence that uses algorithms to enable machines to learn from data and make predictions or decisions without being explicitly programmed to do so. It is a way of teaching computers to learn from experience, much like humans do.

As illustrated in Figure 4.1, there are three main types of machine learning algorithms: supervised learning, unsupervised learning, and reinforcement learning.

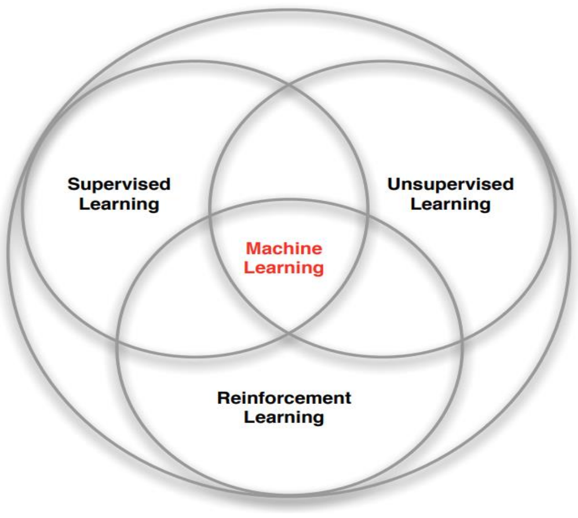


Figure 4.1: Machine learning branches

Supervised learning:

Supervised learning is a type of machine learning where the algorithm is trained on a labeled dataset, where the desired outcome variable (or “label”) is known. The algorithm uses this training data to learn a function that maps inputs to outputs, and can then use this function to make predictions on new, unseen data.

Unsupervised learning:

Unsupervised learning is a type of machine learning where the algorithm is trained on an unlabeled dataset, where the desired outcome variable (or “label”) is not known. The algorithm uses this training data to find patterns or relationships in the data, often by grouping similar data points together into clusters.

Reinforcement learning:

"Reinforcement learning is a subset of machine learning focused on learning what to do—how to map situations to actions—so as to maximize a numerical reward signal. The learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics—trial-and-error search and delayed reward—are the two most important distinguishing features of reinforcement learning." (Sutton & Barto)[]

2. RL components:

2.1. Agent-Environment interface:

In reinforcement learning (RL), the interaction between the agent and its environment forms the core components of the learning process. This interaction is characterized by a loop where the agent observes the current state of the environment and takes actions based on a policy. The agent's objective is to learn an optimal policy that maximizes the cumulative rewards over time.

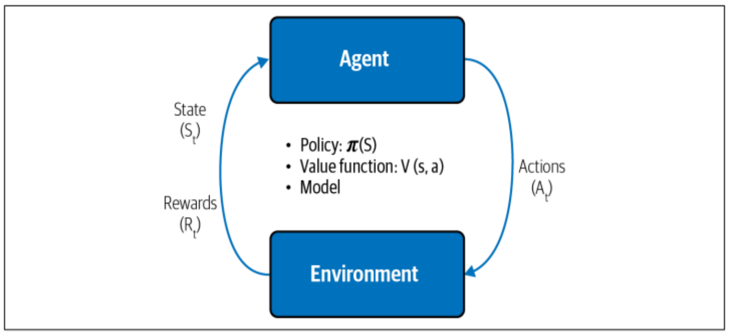


Figure 4.2: RL components

The loop is composed as follows:

at time t :

- the agent takes an action a(t).
- the agent obtains a (short-term) reward r(t).
- then the state of the world becomes s(t+1).

Agent: The agent is the entity that learns and makes decisions in the RL framework. It interacts with the environment, observes its current state s(t), and takes actions a(t) based on a policy (π).The agent's objective is to learn an optimal policy that maximizes the cumulative rewards over time.

Environment: The environment represents the world or system in which the agent operates. It can be any context or scenario in which the agent seeks to learn and improve its decision-making. The environment can be simulated, real-world, or even a combination of both.

State: The state refers to the current condition or configuration of the environment at a particular timestep. It contains all relevant information required for decision-making. The agent observes the state and uses it to determine its next action.

Action: An action is a specific decision or behavior that the agent can take in a given state. The set of available actions depends on the problem domain and the specific task the agent is trying to solve. Actions can be discrete (e.g., selecting a specific financial instrument, initiating a long or short position) or continuous (e.g., adjusting portfolio weights, setting risk thresholds).

2.2. Policy, reward, value function and model :

Beyond the agent and the environment, one can identify four main sub elements of a reinforcement learning system: a policy, a reward signal, a value function, and, optionally, a model of the environment.

Policy: Refers to the strategy or rule that an agent follows to make decisions in an environment. It is a mapping from states to actions, where the policy determines which action the agent should take in a given state. The policy is a fundamental component of an RL agent as it guides the agent's behavior and influences its learning process. Policies can be:

- **Deterministic**, where each state is associated with a specific action.

$$A_{t+1} = \pi(s_t)$$

- **Stochastic**, where probabilities are assigned to each possible action in a given state.

$$\pi(a|s) = P[a_t = a|s_t = s]$$

Reward signal: Serves as the guidance for the agent's behavior. It is a numeric value provided by the environment to the agent at each time step. The objective of the agent is to maximize the total reward received over the long run. The reward signal defines what events are favorable or unfavorable for the agent, and it forms the basis for updating the agent's policy. If an action leads to a low reward, the policy may be adjusted to choose a different action in similar situations in the future.

Value function: Whereas the reward signal indicates what is good in an immediate sense, a value function specifies what is good in the long run. The value of a state represents the total accumulated reward an agent can expect in the future, starting from that state. Values consider the likelihood of future states and the rewards associated with them. For instance, a state with a low immediate reward can still possess a high value if it consistently leads to subsequent states with high rewards. Action choices are guided by value assessments, as the goal is to select actions that lead to states of maximum value rather than immediate reward. Values must be estimated and re-estimated from the sequences of observations an agent makes over its entire lifetime.

Model: In reinforcement learning, an essential component in most algorithms is the ability to efficiently estimate values. This component, commonly known as a model. It captures the behavior of the environment and enables making inferences about its behavior. For instance, given a state and action, the model can predict the subsequent state and associated reward. Models are used for planning, by which we mean any way of deciding on a course of action by considering possible future situations before they are actually experienced.

There are two main types of methods for solving reinforcement learning problems: model-based methods and model-free methods.

- **Model-based methods:** Use models and planning to solve reinforcement learning problems.
- **Model-free methods:** Are explicitly trial-and-error learners and are viewed as almost the opposite of planning.

Overall, an RL agent aims to learn a policy or value function, as illustrated in Figure 4.3. The choice of learning approach for a policy depends on the type of RL model being used. In cases where the environment is fully known, optimal solutions can be achieved through model-based approaches. However, when the environment is unknown, a model-free approach is employed, where the agent explicitly learns the model as part of the algorithm. This allows for adaptive learning in the absence of complete knowledge about the environment.

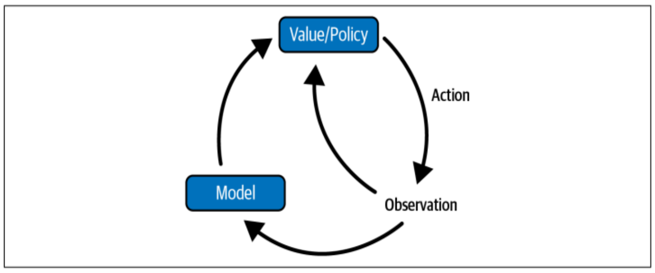


Figure 4.3: Model, value and policy

2.3. Markov Decision Processes:

Reinforcement learning operates within the framework of Markov decision processes (MDPs), providing a formal structure to capture the interaction between a learning agent and its environment. This framework involves defining the process as a sequential progression of states, where a state is considered Markov if the probability of transitioning to the next state relies solely on the current state and is independent of the past states. By leveraging this Markov property, reinforcement learning algorithms can effectively model and optimize decision-making processes in dynamic environments.

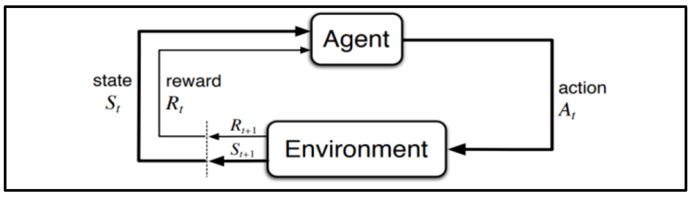


Figure 4.4: The agent-environment interaction in a Markov decision process

A Markov decision process is a tuple (S,A,P,γ,R),where:

- **S** is a finite set of states
- **A** is a finite set of actions
- **P** is the state transition probability matrix: $P[S_{t+1} = s' | S_t = s, A_t = a]$
- γ ∈ [0, 1] is called the discount factor.
- **R** : S × A → IR is a reward function

The goal of RL is to maximize the expected value of the return by choosing the optimal policy and value function. A policy, π, is the thought process behind picking an action. It is a function that maps the states to the actions (Littman & Szepesv´ari, 1996). If the policy is deterministic, then we have:

$$A_{t+1} = \pi(s_t)$$

The goal is to maximize the expected return where the return, G_t, is a summation of all the discounted rewards received by the agent from time step t onwards:

$$G_t = R_{t+1} + \gamma \cdot R_{t+2} + \gamma^2 \cdot R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

The value function, V_π(s), estimates the expected return starting from a specific state:

$$V_\pi(s) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s \right]$$

The action-value function $Q_\pi(s, a)$ is the expected return starting from state s, taking a specific action, a, and then following the policy π:

$$Q_\pi(s, a) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a \right]$$

With the help of the Bellman equation, the value function can be decomposed into two parts: the immediate reward, R_{t+1}, and the discounted value of the successor state, $\gamma V_\pi(s_{t+1})$:

$$\begin{aligned}
V_\pi(s) &= \mathbb{E}_\pi \left[G_t \mid S_t = s \right] \\
&= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s \right] \\
&= \mathbb{E}_\pi \left[R_{t+1} + \gamma \left(\sum_{k=1}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s \right) \right] \\
&= R_{t+1} + \gamma \mathbb{E}_\pi \left[\sum_{k=1}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s \right] \\
&= R_{t+1} + \gamma V_\pi(s_{t+1})
\end{aligned}$$

The Bellman Equation for the action-value function can similarly be written as:

$$Q_\pi(s, a) = R_{t+1} + \gamma Q_\pi(s_{t+1}, a_{t+1})$$

The reward can be maximized by finding the optimal Value function, a function that yields maximum value compared to all other value functions. The Bellman Optimality Equation expresses that the value of a state under an optimal policy must be equal to the expected return from the best action in that state:

$$V^*(s) = \max_a Q^{\pi^*}(s, a) = \max_a \mathbb{E} \left[R_{t+1} + \gamma V^*(s_{t+1}) \mid S_t = s, A_t = a \right]$$

$$Q^*(s,a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \mid S_t = s, A_t = a\right]$$

There are three major approaches to RL:

- **Critic-only approach**: In the critic-only approach, the agent makes decisions based solely on a value function, typically represented as Q. By leveraging this value function, the agent can analyze the current state of the environment and make decisions based on the expected outcomes associated with different actions. The agent's actions are guided by the assessment of the value function without explicitly considering a policy.

$$Q^*(s_t,a_t) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \mid s_t = s_t, a_t = a_t\right]$$

- **Actor-only method**: The actor-only method focuses exclusively on policy optimization. It entails training an actor network that directly maps states to actions, without explicitly estimating a value function. The actor network learns to maximize the cumulative reward by exploring the environment and updating its policy based on the observed rewards. The policy is typically optimized using techniques like policy gradients or evolutionary algorithms. In this approach, the policy is defined by a set of parameters (θ) that specify the actor's behavior.

$$a_t = \pi_\theta(s_t)$$

- **Actor-critic method**: The actor-critic method combines both policy optimization and value estimation. It involves training an actor network and a critic network simultaneously. The actor network learns to select actions based on the observed states, while the critic network estimates either the value function or the Q-function. The critic provides feedback to the actor by evaluating the chosen actions and estimating their quality in different states. This feedback loop allows the actor to update its policy and improve decision-making. The actor-critic approach benefits from the integration of both policy optimization and value estimation, making it often more efficient than actor-only or critic-only methods.

3. Taxonomy of RL Approaches:

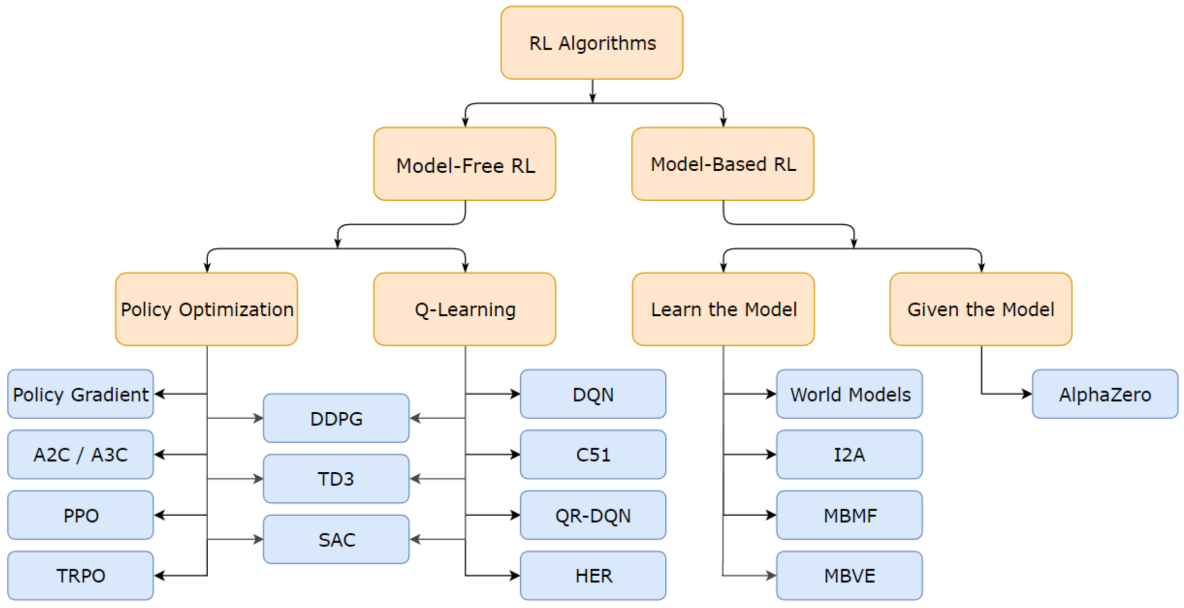


Figure 4.5: Reinforcement Learning taxonomy as defined by OpenAI

The first basic distinction that can be traced between RL approaches, separates model-based and model-free algorithms as shown in the previous section. Given that the algorithms adopted in this work are all model-free, a comprehensive discussion of model-based approaches is beyond the scope of this text. However, for a more detailed understanding of model-based methods, please refer to (14).

Within model-free algorithms, another distinction can be made between those that aim to learn the optimal policy (Policy optimization) and those that aim to approximate the optimal Q Function (Q-Learning).

Policy Optimization: Policy optimization algorithms directly optimize the policy of an agent by iteratively updating its parameters. The objective is to find the policy that maximizes the expected cumulative reward over time. These algorithms typically use techniques such as gradient ascent to improve the policy based on observed rewards. By iteratively adjusting the policy, the agent learns to select actions that lead to higher rewards and improved performance. Policy optimization methods offer stability and reliability, making them less prone to failures. However, they may require a larger amount of data and interactions with the environment for convergence. Examples of policy optimization algorithms include Proximal Policy Optimization (PPO) and Advantage Actor-Critic(A2C).

- **Advantage Actor-Critic**: A2C is a synchronous variant of the Advantage Actor-Critic algorithm. It combines the advantages of both policy-based methods (Actor) and value-based methods (Critic). The Actor network selects actions based on the current policy, while the Critic network estimates the value function to evaluate the quality of actions. By iteratively updating the policy parameters and the value function, A2C aims to find the policy that maximizes the expected cumulative reward over time. One advantage of A2C is its simplicity and lightweight framework, making it suitable for both small-scale and large-scale RL problems.
- **Proximal Policy Optimization:** PPO is a state-of-the-art Policy Optimization algorithm that addresses the issue of unstable policy updates. It utilizes a trust region approach to limit the policy update and ensure stability during training. PPO alternates between sampling data through interaction with the environment and optimizing a surrogate objective function using stochastic gradient ascent. This approach improves sample efficiency and provides a good balance between stability and performance. PPO is widely recognized for its ease of implementation and better sample complexity.

Q-Learning: Q-learning algorithms, including Deep Q-Network (**DQN**), focus on estimating the Q-values of state-action pairs. The Q-values represent the expected cumulative reward for taking a particular action in a given state. Q-learning algorithms iteratively update these Q-values based on observed rewards and estimated future rewards. The goal is to find the optimal Q-values that maximize the cumulative reward. The agent then selects actions based on the learned Q-values to make decisions. Q-Learning models provide higher sample efficiency and can handle both discrete and continuous action spaces. However, they may face challenges in finding the right exploration strategy and convergence issues in complex environments.

Mixed methods: Furthermore, it is possible to achieve a balance between these two approaches by utilizing mixed methods that combine elements from both classes. Three examples of such methods are Deep Deterministic Policy Gradient (DDPG), Soft Actor-Critic (SAC), and Twin Delayed Deep Deterministic Policy Gradient (TD3).

- **Deep Deterministic Policy Gradient**: DDPG is an algorithm that learns both an approximator for the Optimal Policy and the Optimal Q function. By combining elements of policy optimization and Q-learning, DDPG leverages the interaction between these two components to optimize and improve each other.
- **Twin Delayed Deep Deterministic Policy Gradient**: TD3 is an algorithm that improves upon DDPG by introducing several tricks to address the issue of overestimation bias in Q-value estimation. Overestimation bias refers to the tendency of the algorithm to overestimate the value of certain actions, which can lead to suboptimal behavior. TD3 addresses this issue by using two Q-functions instead of one, and taking the minimum of their estimates to form the target for the Bellman error loss functions. This helps to reduce overestimation bias and improve performance. Another trick used by TD3 is adding noise to the target policy. This means that when updating the policy, TD3 adds noise to the target action, which makes it harder for the policy to exploit Q-function errors by smoothing out Q along changes in action. This can help prevent the policy from prematurely converging to a bad local optimum. TD3 also updates the policy less frequently than the Q-functions. This means that the policy is updated only after a certain number of Q-function updates have been performed. This can help stabilize learning and improve performance.

- **Soft Actor-Critic**: SAC is an algorithm that encourages exploration by giving the agent a bonus reward for choosing actions that are uncertain. This bonus reward is proportional to the entropy of the policy, which measures how unpredictable the policy is. Entropy regularization is a central feature of SAC, and it helps to balance the trade-off between exploration and exploitation. By increasing entropy, SAC encourages more exploration, which can accelerate learning later on and prevent the policy from prematurely converging to a bad local optimum.

4. Challenges:

In the field of reinforcement learning, there exist several challenges that researchers and practitioners encounter. Two prominent challenges include the balance between exploration and exploitation, as well as the handling of high-dimensional state spaces.

- **Exploration vs Exploitation**: One of the challenges that arise in reinforcement learning, and not in other kinds of learning, is the trade-off between exploration and exploitation. To obtain a lot of reward, a reinforcement learning agent must prefer actions that it has tried in the past and found to be effective in producing reward. But to discover such actions, it has to try actions that it has not selected before. The agent has to exploit what it has already experienced in order to obtain reward, but it also has to explore in order to make better action selections in the future. The dilemma is that neither exploration nor exploitation can be pursued exclusively without failing at the task. The agent must try a variety of actions and progressively favor those that appear to be best. On a stochastic task, each action must be tried many times to gain a reliable estimate of its expected reward. The exploration–exploitation dilemma has been intensively studied by mathematicians for many decades, yet remains unresolved.
- **High-Dimensional State and Action Spaces**: Reinforcement learning becomes more challenging when dealing with high-dimensional state and action spaces. As the dimensionality of the problem increases, the agent needs to efficiently explore and exploit the vast space of possible actions and states.

Literature review

Portfolio optimization is a crucial topic in finance, as it involves the selection of the best combination of assets to include in a portfolio in order to maximize returns while minimizing risk. There are several approaches to portfolio optimization, including the classical approach, machine and deep learning approach, and reinforcement learning approach. These approaches use different techniques and algorithms to model the complex dynamics of financial markets and construct optimized portfolios. In this literature review, we will examine some papers on these approaches and their applications

1. Traditional approach:

The traditional approach to portfolio optimization involves the use of classical methods such as the Mean-Variance approach, the Maximum Sharpe ratio approach, and the Risk parity model. These approaches were introduced by Harry Markowitz, William F. Sharpe, and Edward Qian, respectively, and have been widely used in the field of finance for many years. They use expected returns and risks (volatility) as inputs to determine the optimized weights for each asset in the target portfolio.

Mean-Variance method: The Mean-Variance method, also known as the Markowitz model, is a classical approach to portfolio optimization that was first introduced by Harry Markowitz in his 1952 paper "Portfolio Selection". This approach uses expected returns and risks (volatility) as inputs to determine the optimized weights for each asset in the target portfolio. The goal of the Mean-Variance method is to find the portfolio that lies on the efficient frontier, which represents the set of portfolios that offer the highest expected return for a given level of risk [25].

Maximum Sharpe ratio: The Maximum Sharpe ratio approach is another classical approach to portfolio optimization that seeks to maximize the Sharpe ratio, a measure of risk-adjusted return. The Sharpe ratio is calculated by dividing the expected excess return of a portfolio by its standard deviation, which represents its risk. The Maximum Sharpe ratio approach aims to find the portfolio with the highest expected excess return per unit of risk. This approach was first introduced by William F. Sharpe in his 1966 paper “Mutual Fund Performance” [32].

Risk parity model: The Risk parity model is a more recent approach to portfolio optimization that aims to achieve diversification by allocating capital to assets in such a way that each asset contributes equally to the overall portfolio risk. This approach was first introduced by Edward Qian in his 2005 paper “Risk Parity Portfolios: Efficient Portfolios Through True Diversification”. The Risk parity model uses an iterative algorithm to construct portfolios that balance risk contributions from different asset classes [29].

These traditional approaches to portfolio optimization have been widely used in finance for many years and continue to be popular due to their simplicity and effectiveness. However, they do have some limitations, such as their reliance on historical data and assumptions about future market behavior. In recent years, there has been a growing interest in the use of machine learning, deep learning, and reinforcement learning techniques for portfolio optimization, which offer more advanced and flexible methods for modeling complex market dynamics

2. Machine learning and Deep learning approach:

Machine learning and deep learning techniques have been increasingly applied to the field of finance, particularly in portfolio optimization. These approaches use advanced algorithms to model the complex dynamics of financial markets and construct optimized portfolios. By incorporating machine learning and deep learning techniques, researchers aim to improve upon traditional portfolio optimization methods and achieve better performance. In this part, we will introduce three papers that discuss the application of these techniques to portfolio optimization.

Wang et al. (2019) present a novel approach to portfolio optimization in their paper “Portfolio formation with preselection using deep learning from long-term financial data” [41]. The authors propose a two-stage process that combines long short-term memory (LSTM) networks and the mean-variance model (MV). In the first stage, LSTM networks are used to forecast asset returns and select those with higher potential returns. The LSTM networks outperform several benchmark models by a very clear margin, including support vector machine (SVM), random forest (RF), deep neural networks (DNN), and autoregressive integrated moving average model (ARIMA). In the second stage, the MV model is applied to optimize the portfolio based on the selected assets. The proposed method is validated using data from the UK Stock Exchange 100 Index between March 1994 and March 2019 and outperforms several baseline strategies in terms of cumulative return per year, Sharpe ratio per triennium, and average return to risk per month of each triennium. Additionally, the use of LSTM networks allows for dynamic incorporation of new data, enabling the proposed approach to adapt to changing market conditions.

In 2021, a paper titled “Portfolio optimization with return prediction using deep learning and machine learning” was published in the journal Expert Systems With Applications by authors Yilin Ma, Ruizhu Han, and Weizhong Wang [24]. The paper presents a novel approach to portfolio optimization by integrating return prediction of traditional time series models. The paper combines return prediction in portfolio formation with two machine learning models, i.e., random forest (RF) and support vector regression (SVR), and three deep learning models, i.e., LSTM neural network, deep multilayer perceptron (DMLP) and convolutional neural network. The paper applies these prediction models for stock preselection before portfolio formation and incorporates their predictive results in advancing mean–variance (MV) and omega portfolio (OF) optimization models. The evaluation is based on historical data of 9 years from 2007 to 2015 of component stocks of China securities 100 index. Experimental results show that MV and omega models with RF return prediction, i.e., RF+MVF and RF+OF, outperform the other models. Further, RF+MVF is superior to RF+OF.

The paper also discusses the impact of transaction costs on the performance of their models. When including transaction costs, the RF+MVF model still outperforms other models in terms of performance. This research provides valuable insights for investors and researchers interested in applying machine learning and deep learning techniques to portfolio optimization.

In the preprint paper “Portfolio Optimization with LSTM-Based Return and Risk Information” by Yong Zhang, Yongbin Su, Weilong Liu, and Xingyu Yang [43], the authors propose a deep learning-based mean-variance portfolio optimization that incorporates both prediction-based return and prediction-based risk information. The authors use a long short-term memory (LSTM) network-based predictor with inputs of technical indicators to predict stock returns and select a proportion of stocks with high predicted returns to construct the portfolio. Then they formulate a novel risk measure based on the prediction errors of the LSTM return and construct the covariance matrix based on those errors. This allows them to incorporate both predicted return and risk information into the portfolio optimization process, with the aim of improving the performance of their portfolio strategy. In each subsequent trading period, the prediction-based risk covariance matrix is updated on the basis of newly acquired information, allowing the proposed approach to adapt to changing market conditions. The proposed approach was evaluated using stock data from various markets, demonstrating that incorporating prediction-based return and risk information can enhance the profitability and performance of portfolio strategies. The authors compare the performance of their proposed LSTM+PPMV strategy to several other models using four datasets from different stock markets. These datasets include the Shanghai Stock Exchange 50 Index (SSE50), the China Securities 100 Index (CSI100), the Heng Seng Index (HSI), and the Dow Jones Industrial Average (DJIA). The other models used in this paper include the LSTM+PMV strategy, which is similar to the LSTM+PPMV strategy, but instead of incorporating both prediction-based return and prediction-based risk information, it only uses prediction-based return information; the LSTM+MVF strategy, which integrates predicted return with an improved mean-variance model, as reported in Ma et al. (2021); the RF+PPMV strategy, which examines the influence of a random forest (RF) model on performance, the Adaboost+PPMV strategy, which employs an Adaboost model to forecast future returns, where AdaBoost is one of the popular Boosting methods proposed by Freund & Schapire (1997); and the Markowitz mean-variance (MV) strategy, which determines optimal investment proportions by minimizing variance. The results demonstrate that the proposed LSTM+PPMV strategy performs favorably in comparison to these other models.The experimental results indicated that this strategy (LSTM+PPMV) achieved the highest performance, even when transaction costs were taken into account, provided they did not exceed 0.25%. This paper presents a valuable contribution to the literature on portfolio optimization using deep learning techniques.

In conclusion, machine learning and deep learning techniques have been increasingly applied to the field of finance, particularly in portfolio optimization. These approaches use advanced algorithms to model the complex dynamics of financial markets and construct optimized portfolios. By incorporating machine learning and deep learning techniques, researchers aim to improve upon traditional portfolio optimization methods and achieve better performance. The papers discussed in this section present novel approaches to portfolio optimization that integrate machine learning and deep learning techniques, such as LSTM networks, random forest, and support vector regression, with traditional portfolio optimization models, such as the mean-variance model and the omega model. These approaches have been shown to outperform traditional methods and adapt to changing market conditions.

3. Reinforcement learning approach:

Reinforcement learning (RL) has shown great potential in solving complex sequential decision-making problems, including portfolio optimization. Recent research has demonstrated the effectiveness of RL techniques in optimizing stock trading strategies, outperforming traditional finance methods such as Mean-Variance Optimization (MVO). This literature review summarizes the key findings of three papers that propose the use of Deep Reinforcement Learning (DRL) techniques for portfolio optimization.

The paper “FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance” published by Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang in 2020 [21] proposes the use of Deep Reinforcement Learning (DRL) techniques to optimize stock trading strategies. The authors demonstrate the effectiveness of their approach through empirical experiments.

According to the results, the authors were able to reproduce a portfolio allocation strategy that uses a DRL agent to allocate capital to a set of stocks and reallocate periodically. The training period was from 2009/01/01 to 2020/06/30 on a daily basis, and the testing period was from 2020/07/01 to 2021/06/30. The backtesting performance on Dow 30 constituent stocks shows that each DRL agent, namely A2C, TD3, PPO, and DDPG, outperforms the DJIA index and the min-variance strategy. A2C has the best performance with a Sharpe ratio of 2.36 and an annual return of 42.57%. In conclusion, the paper presents a valuable resource for researchers and practitioners interested in applying DRL techniques to portfolio optimization.
In 2022, Luthfianti et al. published a paper titled “Portfolio Allocation of Stocks in Index LQ45 using Deep Reinforcement Learning” [23] discusses the use of the Advantage Actor-Critic (A2C) algorithm, which is a type of Deep Reinforcement Learning (DRL), to construct a portfolio consisting of stocks in the LQ45 index in the Indonesian Stock Exchange. The data used is daily closing price data from January 15, 2014, to January 1, 2020. The experiment is conducted, including a combination of the number of shares in the portfolio 3, 5, 7, and 42 stocks. The results show that the portfolio value and the Sharpe Ratio of the DRL portfolio are better than the Equal Weight and Mean-Variance portfolio. Also, the performance of the DRL portfolio is much better for a small number of stocks.
Sood et al (2023) present a novel approach to portfolio optimization in their paper titled “Deep Reinforcement Learning for Optimal Portfolio Allocation: A Comparative Study with Mean-Variance Optimization” [34], which compares a Deep Reinforcement Learning (DRL) framework that uses the Proximal Policy Optimization (PPO) algorithm with Mean-Variance Optimization (MVO), a traditional finance method for portfolio optimization. The DRL framework aims to maximize the differential Sharpe ratio, which is used as the reward. The authors train the model on a multi-asset trading environment that simulates the US Equities market, using daily adjusted close price data of the S&P500 sector indices, the VIX index, and the S&P500 index between 2006 and 2021.
According to the results of the study, the Deep Reinforcement Learning (DRL) framework outperforms Mean-Variance Optimization (MVO) in several metrics, achieving higher Annual Returns and Sharpe ratio, as well as the smallest Maximum Drawdown. In fact, the annual returns and Sharpe ratio of the DRL framework were 1.85 times higher than those of the MVO portfolio.
In conclusion, the papers discussed in this section demonstrate the potential of Deep Reinforcement Learning (DRL) techniques for portfolio optimization. These techniques have been shown to outperform traditional finance methods several metrics, achieving higher returns and Sharpe ratios. The papers propose the use of various DRL algorithms, including Proximal Policy Optimization (PPO) and Advantage Actor-Critic (A2C), to construct portfolios that adapt to changing market conditions. These approaches have been validated using data from various markets, demonstrating the effectiveness of DRL techniques for portfolio optimization.

Methodology

In this section, we present the methodology used in our research. The process is illustrated in the graphic below (figure 5.1), which shows the main steps involved.

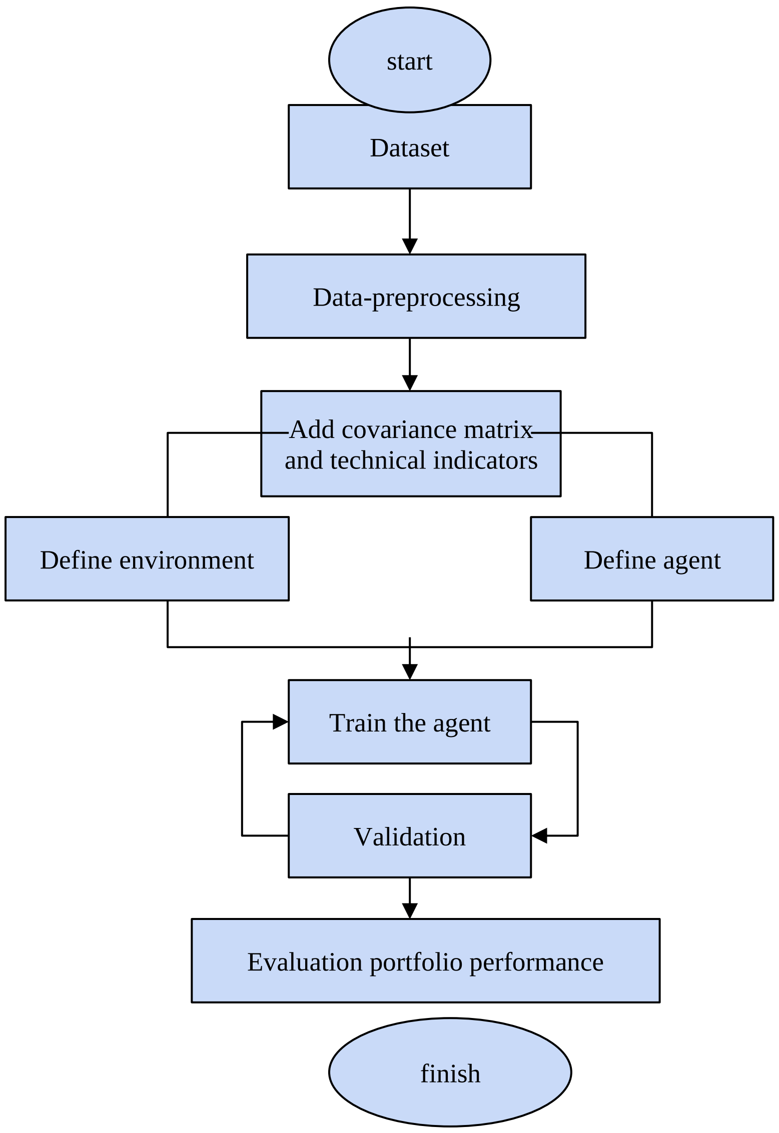


Figure 5.1: Methodology Overview

1. Input Data:

1.1. OHLCV data:

The historical price data for the four markets, namely DAX 30, DJIA 30 and CAC 40, spanning from December 27, 2013, to May 31, 2023, was obtained using the Yahoo Finance library. The dataset includes Open, High, Low, Close, and Volume (OHLCV) values.

German Stock Index(DAX 30): A stock market index consisting of the 30 largest publicly traded companies on the Frankfurt Stock Exchange in Germany, providing insights into the overall economic performance of the country.

Dow Jones Industrial Average (DJIA30): Made up of 30 major publicly traded companies in the United States, representing various industries. It's one of the oldest and most recognized stock market indices.

Continuous Assisted Quotation (CAC 40): A benchmark index comprising the top 40 companies listed on the Euronext Paris stock exchange, representing the French stock market's performance.

It's important to note that during this time interval, certain stocks might not have been available in the market. For instance, the CAC40, which ideally consists of 40 stocks, may not have the complete 40 stocks during certain periods. This could be due to stocks being added or removed from the index, resulting in a varying number of constituents over time.

Index	Number of stocks available
DJIA30	29
CAC40	34
DAX30	25

Table 5.1: number of stocks available for each index

1.2. Technical indicators:

Technical indicators are essential tools that provide insights into market trends and potential opportunities. In this study, these indicators were utilized as inputs for the reinforcement learning (RL) models. Here are the technical indicators used:

Simple Moving Average (SMA (n)): SMA (n) calculates the average price of an asset over the past n days, providing a smoothed representation of its price trend.

$$\text{SMA (n)} = \frac{1}{n}\sum_{t=1}^{n} \text{Price}_t$$

Moving Average Convergence Divergence (MACD): identifies changes in momentum, calculated as the difference between the 26-day and 12-day exponential moving averages (EMAs).

$$\text{MACD} = \text{EMA}_{12} - \text{EMA}_{26}$$

Directional Movement Index (DX (n)): measures the strength of a trend and indicates whether it's upward or downward.

$$\text{DX (n)} = 100\times\frac{\text{Smoothed Positive Directional Index (DX+)} - \text{Smoothed Negative Directional Index (DX-)}}{\text{Sum of DX+ and DX-}}$$

Commodity Channel Index (CCI (n)): gauges the asset's current price level relative to its historical average, highlighting potential overbought or oversold conditions.

$$\text{CCI (n)} = \frac{1}{0.015}\times\frac{\text{Typical Price} - \text{SMA (n) of Typical Price}}{\text{Mean Deviation}}$$

Relative Strength Index (RSI (n)): assesses the asset's recent gains and losses, indicating potential overbought or oversold conditions.

$$\text{RSI-n} = 100 - \frac{100}{1+\frac{\text{Average Gain}}{\text{Average Loss}}}$$

Bollinger Bands (BollUB and BollLB): Bollinger Bands consist of an upper (BollUB) and lower (BollLB) band that track price volatility and potential reversal points.

$$\text{BollUB} = \text{SMA (n)} + (2\times\text{Standard Deviation of Price})$$
$$\text{BollLB} = \text{SMA (n)} - (2\times\text{Standard Deviation of Price})$$

1.3. Rolling covariance matrix:

The covariance matrix served as a dual-purpose input for both the Minimum Variance Portfolio (MVP) optimization and the Reinforcement Learning (RL) framework. In the MVP approach, it guided the optimization process to achieve minimal portfolio variance. For RL, the covariance matrix was utilized as part of the state representation, capturing historical relationships among assets' returns. This matrix was updated using a rolling window of the previous 252 trading days to ensure relevance and adaptability.

2. Data splitting:

The dataset was split into training, validation, and testing sets to facilitate model development and evaluation. We used the data from December 27, 2013, to December 31, 2020, for training purposes.

The data from January 4, 2021, to December 31, 2021, was used as a validation set to fine-tune the model's hyperparameters and prevent overfitting. After choosing the best hyperparameters for each model, we retrained the model on the combined training and validation sets. The data from January 3, 2022, to May 31, 2023, was exclusively reserved for testing the model's performance.
This approach allowed for a thorough evaluation of the model's performance under varying timeframes, ensuring its robustness across different market conditions.

Training data		Validation data		Test data	
Date range	steps	Date range	steps	Date range	steps
27/12/2013 to 31/12/2020	1794	04/01/2021 to 31/12/2021	258	03/01/2022 to 31/05/2023	614

Table 5.2: Train, validation and test set

3. Assumptions:

No slippage: Slippage is the difference between the expected price at which an order is placed and the actual price at which the trade occurs. It's common in volatile markets or when trading assets with limited market liquidity. We assume that our portfolio's traded assets possess significant liquidity, allowing for instantaneous trades at market prices, thereby reducing the impact of slippage.
Liquidity: All traded assets in our portfolio are easily convertible to cash with minimal loss due to sufficient liquidity.
No Market Impact: Despite trading volumes potentially influencing market prices, we assume our agents' trading volume is insignificant in impacting the market. As a result, our state-space excludes market impact information.
Null Risk-Free rate: In the context of this study, the risk-free rate is considered to be equal to zero. This assumption simplifies the analysis and aligns with the premise that no risk-free return is factored into the portfolio optimization and evaluation processes.

4. Rebalancing:

In this study, all portfolios managed by the DRL agents (DDPG, A2C, PPO, SAC and TD3) are rebalanced daily, involving the adjustment of the asset weights within each portfolio. To simulate real-world trading conditions, certain practical constraints are incorporated. Specifically, each DRL agent is limited to trading a maximum of 100 shares per asset per day. Additionally, a transaction cost percentage of 0.1% is applied to every trade executed by the DRL agents. Furthermore, the initial investment for each DRL agent is set at $1,000,000, providing a consistent starting point for evaluating their performance and optimization strategies. These measures are taken to ensure that our analysis accurately captures real-world trading complexities and enhances the applicability of our findings.
We should note that both the MVP and the market capitalization-weighted (Market Cap) portfolios are rebalanced daily. This iterative rebalancing strategy dynamically maintains optimal allocations for the MVP and ensures that the assets of the Market Cap portfolios dynamically maintain their allocation based on market capitalization. Notably, the rebalancing for the MVP portfolio is conducted without incurring any transaction fees.

5. The DRL models:

5.1. State space, action space, reward function and environment:

State space: The state space describes the observations that the agent receives from the environment. Just as a human trader needs to analyze various information before executing a trade, so our trading agent observes many different features to better learn in an interactive environment.

This data includes:

- Balance: the amount of money left in the account at the current time step t.
- Shares: current shares for each stock.
- OHLC prices: used to track stock price changes.
- Trading volume : total quantity of shares traded during a trading slot.
- Technical indicators: MACD, CCI, RSI, DX.
- Rolling Covariance matrix.

Action space: The portfolio weight of each stock is in the interval [0, 1].

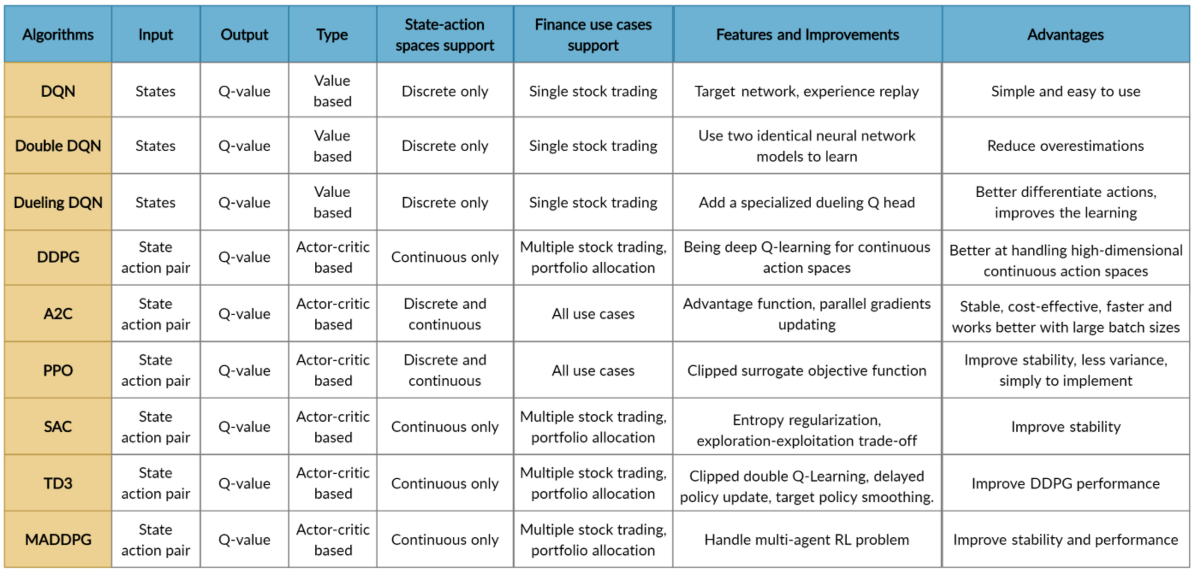
Reward function: The reward function r(s, a, s′) is the incentive mechanism for an agent to learn a better action. In our study the reward represents the change of the portfolio value when action a is taken at state s and arriving at new state s':

$r(s, a, s') = v' - v$, where v' and v represent the portfolio values at state s' and s, respectively.

Environment: The environment that the agent interacts with is the stock market. By receiving observations from the stock market, which represent the state of the market, and analyzing various features, the agent is able to learn and make informed decisions. The specific markets used in this study are DJIA30, DAX30, and CAC40 constituents.

5.2. DRL agents:

The Deep Reinforcement Learning (DRL) agent uses algorithms from the FinRL library, including DDPG, PPO, SAC, A2C, and TD3. These algorithms are implemented using the OpenAI Baselines and Stable Baselines frameworks. A comparison of these DRL algorithms can be found in Figure 5.1.



Algorithms	Input	Output	Type	State-action spaces support	Finance use cases support	Features and Improvements	Advantages
DQN	States	Q-value	Value based	Discrete only	Single stock trading	Target network, experience replay	Simple and easy to use
Double DQN	States	Q-value	Value based	Discrete only	Single stock trading	Use two identical neural network models to learn	Reduce overestimations
Dueling DQN	States	Q-value	Value based	Discrete only	Single stock trading	Add a specialized dueling Q head	Better differentiate actions, improves the learning
DDPG	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Being deep Q-learning for continuous action spaces	Better at handling high-dimensional continuous action spaces
A2C	State action pair	Q-value	Actor-critic based	Discrete and continuous	All use cases	Advantage function, parallel gradients updating	Stable, cost-effective, faster and works better with large batch sizes
PPO	State action pair	Q-value	Actor-critic based	Discrete and continuous	All use cases	Clipped surrogate objective function	Improve stability, less variance, simply to implement
SAC	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Entropy regularization, exploration-exploitation trade-off	Improve stability
TD3	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Clipped double Q-Learning, delayed policy update, target policy smoothing.	Improve DDPG performance
MADDPG	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Handle multi-agent RL problem	Improve stability and performance

Figure 5.1: Comparison of DRL algorithms

6. Hyperparameter Tuning:

In machine learning, hyperparameters are parameters that are set before the learning process begins. These parameters control various aspects of the learning process and can have a significant impact on the performance of the algorithm. In this study, we performed hyperparameter tuning for each of the reinforcement learning algorithms we used: A2C, PPO, DDPG, TD3, and SAC. We focused on tuning the most important hyperparameters for each algorithm.

A2C hyperparameters: For the A2C algorithm, we tried different values for the learning rate, number of steps, discount factor (gamma), and entropy coefficient. The learning rate controls the step size of the gradient descent algorithm used to update the model’s parameters. The number of steps determines how many steps of experience to collect before updating the model. The discount factor (gamma) determines how much future rewards are taken into account when calculating the value of a state or action. The entropy coefficient controls the trade-off between exploration and exploitation.

Hyperparameter	Values
discount factor (gamma)	0.99, 0.9
learning rate	0.001, 0.0007, 0.0005
entropy coefficient	0.001, 0.0001
number of steps	5000, 1000

Table 5.3: A2C hyperparameters

PPO hyperparameters: For the PPO algorithm, we tried different values for the learning rate, number of steps, discount factor (gamma), and clip range. The clip range controls how much the policy can change in a single update.

Hyperparameter	Values
discount factor (gamma)	0.99, 0.9
learning rate	0.001, 0.0007, 0.0003
clip ranges	0.2, 0.3
number of steps	2048, 8192

Table 5.4: PPO hyperparameters

DDPG hyperparameters: For the DDPG algorithm, we tried different values for the learning rate, buffer size, soft update factor (tau), discount factor (gamma), and number of learning steps (ls). The buffer size determines how many experiences to store in the replay buffer. The soft update factor (tau) controls how quickly the target network is updated towards the main network.

Hyperparameter	Values
discount factor (gamma)	0.99, 0.9
learning rate	0.001, 0.0001, 0.00001
buffer size	100000, 250000
soft update factor (tau)	0.001, 0.005

Table 5.5: DDPG hyperparameters

TD3 hyperparameters:

Hyperparameter	Values
discount factor (gamma)	0.99, 0.9
learning rate	0.001, 0.0007, 0.0005
buffer size	100000, 250000

soft update factor (tau)	0.001, 0.005

Table 5.6: TD3 hyperparameters

SAC hyperparameters:

Hyperparameter	Values
discount factor (gamma)	0.99, 0.9
learning rate	0.001, 0.0007, 0.0003
buffer size	100000, 250000
soft update factor (tau)	0.001, 0.005

Table 5.6: SAC hyperparameters

7. Benchmark portfolios:

Market-Capitalization-Weighted Portfolios: For the creation of market-capitalization-weighted portfolios, data and weights of each constituent are gathered from the respective capitalization-weighted indices CAC40, DJIA30, HSI50, and MDAX50. The market capitalization data for individual stocks on each rebalancing date is sourced from Yahoo Finance.

$$\text{weight}_i = \frac{p_i \cdot n_i}{\sum_{i=1}^{N}(p_i \cdot n_i)}$$

Where:

weight_i the weight of the ith stock in the portfolio.
p_i is the price of the ith stock.
n_i is the number of stocks of the ith stock.
N is the total number of stocks in the portfolio.

Minimum Variance Portfolio: In this study, we utilized the Minimum Variance Portfolio (MVP) as a benchmark to evaluate the performance of our Reinforcement Learning (RL) based portfolio optimization strategies in comparison to traditional optimization approaches. The mathematical equations for constructing the MVP can be found in Section 2.

We also set the weight values for each asset in the MVP between 0 and 0.1 to reduce the risk of overexposure to a single asset or a group of assets, as well as the risk of extreme short selling. This also increases the diversification and stability of the portfolio, as well as its robustness to estimation errors.

$$w_i \in [0, 0.1] \text{ for } i = 1, 2, \ldots, N \quad \text{and} \quad \sum_{i=1}^{N} w_i = 1$$

8. Evaluation metrics:

To comprehensively evaluate the effectiveness of our portfolio optimization strategies and their comparisons with benchmark portfolios, we employ a range of performance measures. These metrics provide insights into the cumulative return, annual volatility, maximum drawdown, Sharpe ratio, Sortino ratio, and Calmar ratio of the portfolios. Each performance measure offers a unique perspective on the risk and return profile of the portfolios, enabling us to gauge their relative strengths and weaknesses.

Cumulative return (CR): The cumulative return calculates the total percentage change in the portfolio value over the entire investment period.

$$CR = \frac{\text{Portfolio Value at the End of Period}-\text{Initial Portfolio Value}}{\text{Initial Portfolio Value}} \times 100\%$$

Annual volatility (AV): Annual volatility measures the dispersion of portfolio returns over a year.

$$\text{AV} = \sqrt{252} \times \text{Standard Deviation of Daily Returns}$$

Sharpe ratio (SR): The Sharpe ratio, proposed by William F. Sharpe in 1966, is a measure of risk-adjusted return that considers the excess return of an investment (above the risk-free rate) per unit of its volatility. It serves as a metric for evaluating the risk-adjusted performance of investment portfolios. A higher Sharpe ratio indicates a better risk-adjusted performance, as it quantifies how well the portfolio compensates the investor for the level of risk taken.

$$\text{SR} = \frac{\text{E}(r_p)-\text{Risk-Free Rate}}{\sigma_p}$$

Maximum drawdown (MDD): MDD measures the largest decline from the peak in the whole trading period to show the worst case. The formal definition is:

$$\text{Maximum Drawdown (MDD)} = \max_{t\in[0,T]} \left(\frac{\text{Peak Portfolio Value}-\text{Portfolio Value at } t}{\text{Peak Portfolio Value}}\right)$$

Where T represents the total number of time periods.

Sortino ratio (SR): The Sortino ratio evaluates the risk-adjusted return, considering only downside risk, using the downside deviation (DD) which refers only to the standard deviation of negative returns.

$$\text{SR} = \frac{\text{E}(r_p) - \text{Risk-Free Rate}}{\text{DD}}$$

9. The efficient market hypothesis:

The Efficient Market Hypothesis (EMH) is a theory in financial economics that gained prominence after a paper published by Fama in 1970, which proposes that the prices of assets, such as stocks, bonds, and commodities, reflect all available information at any given time. This means that all publicly available information, such as news, financial reports, and economic indicators, is already incorporated into the price of an asset. As a result, it is not possible to consistently outperform the market by using this information, as any new information that becomes available will be quickly reflected in the price of the asset.

The EMH is presented in three forms: weak, semi-strong, and strong.

The weak form efficiency: EMH suggests that all past price and volume information is already reflected in current prices, so technical analysis cannot be used to generate excess returns. Technical analysis is a method of evaluating securities by analyzing statistics generated by market activity, such as past prices and volume. The weak form of the EMH suggests that this information is already incorporated into current prices, so using technical analysis to make trading decisions will not result in higher returns.

The semi-strong form efficiency: EMH suggests that all publicly available information is already reflected in current prices, so fundamental analysis cannot be used to generate excess returns. Fundamental analysis is a method of evaluating securities by attempting to measure their intrinsic value by examining related economic, financial, and other qualitative and quantitative factors. The semi-strong form of the EMH suggests that this information is already incorporated into current prices, so using fundamental analysis or technical analysis to make trading decisions will not result in higher returns.

The strong form efficiency: EMH suggests that all information, both public and private, is already reflected in current prices, so even insider information cannot be used to generate excess returns. Insider information is non-public information about a company that could provide an advantage to investors who have access to it. The strong form of the EMH suggests that even this information is already incorporated into current prices, so using insider information to make trading decisions will not result in higher returns

Results and discussion

In this section, we present the results of our analysis of the performance of different Deep Reinforcement Learning (DRL) algorithms in three different markets: DJIA30, CAC40, and DAX30.

We have identified the best hyperparameters for each reinforcement learning model (A2C, PPO, DDPG, SAC, TD3) in each market through fine-tuning on a validation set. The performance of the DRL algorithms is

evaluated using various metrics such as cumulative returns, annual volatility, Sharpe ratio, maximum drawdown, and Sortino ratio.
We compare the performance of the DRL algorithms with two baselines: the market index and the Minimum Variance Portfolio (MVP).
Finally, we discuss the results of our analysis and provide insights into the potential of DRL algorithms to improve portfolio management and investment strategies. We also consider our results in light of the Efficient Market Hypothesis (EMH).

1. Hyperparameter Selection:

Tables 1, 2, and 3 summarize the best hyperparameters for each reinforcement learning model (A2C, PPO, DDPG, SAC, TD3) in each market (DJIA30, DAX30, CAC40). These hyperparameters were identified through fine-tuning on a validation set to achieve the best performance for each model based on the cumulative return and Sharpe ratio metrics.

DJIA30	
Model	Best hyperparameters
A2C	learning_rate=0.001, n_steps=1000, gamma=0.9, ent_coef=0.0001
PPO	learning_rate=0.0003, n_steps=2048, gamma=0.9, clip_range=0.2
DDPG	learning_rate=0.0001, buffer_size=250000, tau=0.005, gamma=0.9
SAC	learning_rate=0.001, buffer_size=100000, tau=0.001, gamma=0.99
TD3	learning_rate=0.001, buffer_size=100000, tau=0.001, gamma=0.99

Table 6.1: Optimal hyperparameters summary (DJIA30)

CAC40	
Model	Best hyperparameters
A2C	learning_rate=0.0007, n_steps=1000, gamma=0.99, ent_coef=0.001
PPO	learning_rate=0.001, n_steps=8192, gamma=0.9, clip_range=0.3
DDPG	learning_rate=1e-05, buffer_size=250000, tau=0.005, gamma=0.9
SAC	learning_rate=0.0007, buffer_size=250000, tau=0.001, gamma=0.9
TD3	learning_rate=0.001, buffer_size=100000, tau=0.005, gamma=0.99

Table 6.2: Optimal hyperparameters summary (CAC40)

DAX30	
Model	Best hyperparameters
A2C	learning_rate=0.001, n_steps=1000, gamma=0.9, ent_coef=0.0001

PPO	learning_rate=0.0003, n_steps=8192, gamma=0.99, clip_range=0.2
DDPG	learning_rate=1e-05, buffer_size=100000, tau=0.005, gamma=0.99
SAC	learning_rate=0.001, buffer_size=250000, tau=0.001, gamma=0.9
TD3	learning_rate=0.001, buffer_size=250000, tau=0.005, gamma=0.9

Table 6.3: Optimal hyperparameters summary (DAX30)

2. Results:

The results are evaluated using various metrics, such as cumulative returns, annual volatility, Sharpe ratio, maximum drawdown and Sortino ratio. These metrics measure the profitability, risk, and efficiency of the portfolio.

Tables 6.4, 6.5 and 6.6 show the performance of different DRL algorithms in the 3 different markets. The results are based on the best hyperparameters for each model, which are shown in Tables 6.1, 6.2 and 6.3. The results are compared with two baselines: the market index and MVP.

DJIA:

	A2C	**TD3**	**DDPG**	**PPO**	**SAC**	**DJIA**	**MVP**
Cumulative returns	14.75%	14.46%	14.81%	14.58%	**17.59%**	9.33%	11.13%
Annual volatility	16.10%	15.30%	16.00%	15.93%	16.50%	16.00%	**12.71%**
Sharpe ratio	0.44	0.44	0.44	0.44	**0.49**	0.31	0.41
Max drawdown	-21.97%	-17.98%	-21.39%	-21.56%	-20.88%	-21.94%	**-14.48%**
Sortino ratio	0.62	0.63	0.63	0.62	**0.70**	0.44	0.58

Table 6.4: Performance of DRL algorithms compared to DJIA

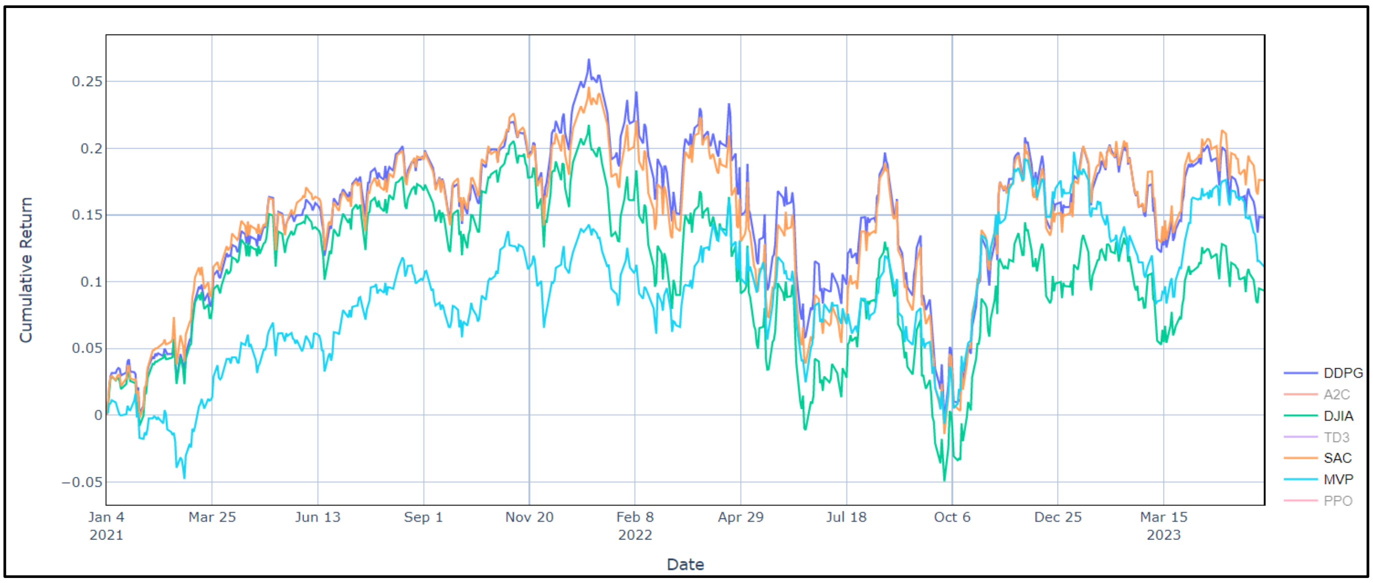


Figure 6.1: Cumulative Returns of DRL Strategies (DDPG and SAC) vs Baselines.

According to table 1, the different Deep Reinforcement Learning (DRL) algorithms performed well in the DJIA market, as measured by various metrics such as cumulative returns, annual volatility, Sharpe ratio, maximum drawdown, and Sortino ratio. These metrics measure the profitability, risk, and efficiency of the portfolio.

In terms of profitability, as measured by cumulative returns, all the DRL algorithms (A2C, TD3, DDPG, PPO, and SAC) achieved higher returns than the market index (DJIA) and the Minimum Variance Portfolio (MVP). This indicates that the DRL algorithms were able to generate higher profits. Among the DRL algorithms, SAC achieved the highest cumulative return at 17.59%, followed by DDPG at 14.81%.

In terms of risk, as measured by annual volatility and maximum drawdown, the DRL algorithms had similar levels of risk to the market index. However, TD3 had slightly lower risk as indicated by its lower annual volatility of 15.30% and lower maximum drawdown of -17.98%. The MVP had the lowest level of risk overall, with the lowest annual volatility at 12.71% and lowest maximum drawdown at -14.48%.

In terms of efficiency, as measured by the Sharpe and Sortino ratios, all the DRL algorithms were more efficient than the market index and MVP. Among the DRL algorithms, SAC achieved the highest Sharpe ratio at 0.49 and highest Sortino ratio at 0.70.

Overall, these results suggest that the DRL algorithms were able to achieve higher returns with similar levels of risk compared to the market index and MVP in the DJIA market. Among the DRL algorithms, SAC performed the best in terms of both returns and efficiency.

Figure 6.1 displays the cumulative returns of DJIA, DDPG, SAC, and MVP. According to the results, the DRL strategies (DDPG and SAC) clearly outperformed the two baselines (DJIA 30 and MVP) for the entire period from January 4th 2021 to May 31st, 2023.

From the beginning of January 4th, 2021 to mid-February, the MVP performed poorly compared to the two DRL strategies and DJIA 30, but then its performance became more similar to DJIA 30. In October 2022, during a period of maximum drawdown, the MVP recovered very rapidly compared to the others and then approached the two DRL strategies and clearly beat DJIA 30 until the end.

Additionally, it can be observed that in all drawdowns, the MVP had the lowest drawdown and recovered very rapidly. It was also less volatile compared to SAC, DDPG, and DJIA which were more volatile.

CAC40:

	A2C	**TD3**	**DDPG**	**PPO**	**SAC**	**CAC40**	**MVP**
Cumulative returns	31.57%	27.15%	**33.12%**	29.63%	28.19%	29.00%	25.18%
Annual volatility	18.49%	18.10%	17.78%	18.72%	19.62%	18.10%	**13.76%**
Sharpe ratio	0.70	0.63	**0.74**	0.66	0.61	0.66	0.73
Max drawdown	-24.25%	-24.71%	-21.13%	-24.51%	-26.80%	-23.04%	**-19.83%**
Sortino ratio	0.99	0.89	**01.05**	0.93	0.87	0.94	01.03

Table 6.4: Performance of DRL algorithms compared to CAC40

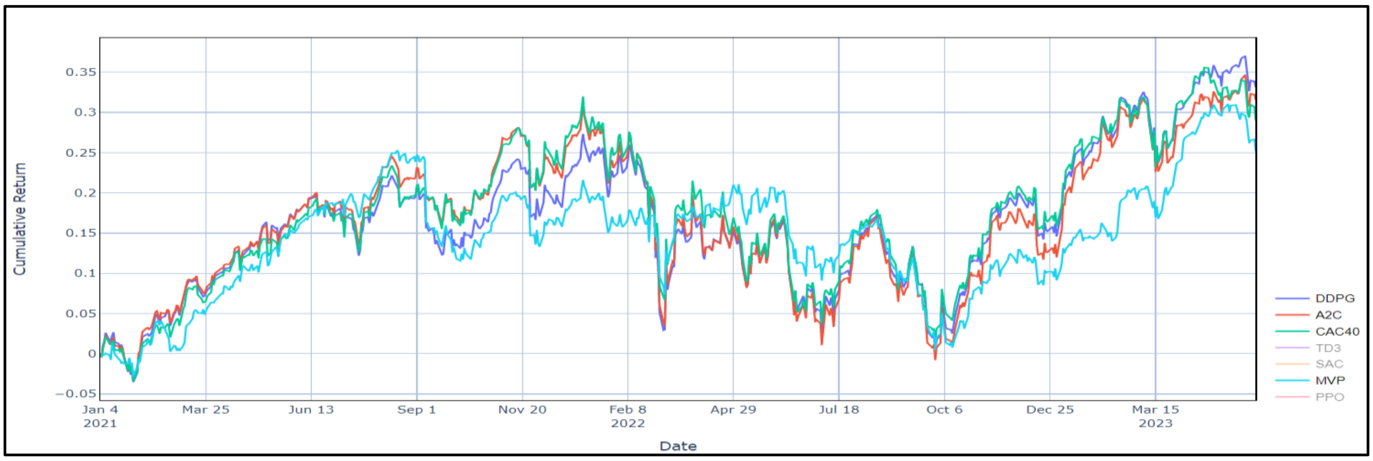


Figure 6.2: Cumulative Returns of DRL Strategies (DDPG and A2C) vs Baselines

According to the table 2, the Deep Reinforcement Learning (DRL) algorithms, demonstrated strong performance in the french market.

In terms of cumulative returns, all DRL algorithms surpassed the Minimum Variance Portfolio (MVP) by at least 2.37%, with the exception of SAC which had a slightly lower return of 28.19%. DDPG achieved the highest cumulative return of 33.12%, indicating its exceptional ability to maximize profits.

When considering annual volatility and maximum drawdown, the DRL algorithms showed risk levels similar to the market index. However, SAC has a slightly higher risk profile with an annual volatility of 19.62% and a maximum drawdown of -26.80%. On the other hand, MVP demonstrated the lowest risk profile with an annual volatility of 13.76% and a maximum drawdown of -19.83%.

In terms of portfolio efficiency, evaluated using Sharpe and Sortino ratios, the DRL algorithms showed varied performance. DDPG achieved the highest Sharpe ratio of 0.74 and Sortino ratio of 1.05, suggesting that it was most effective in providing risk-adjusted returns and handling downside risk.

In summary, while all DRL algorithms demonstrated promising results in the CAC40 market with higher returns and comparable risk to the market index and MVP, DDPG emerged as the leading performer in terms of both returns and efficiency. Figure 6.2 displays the cumulative returns of DRL strategies (DDPG and A2C) and the baselines (MVP and CAC40).

According to the results, all algorithms had similar cumulative returns until February 1st, 2022, when a maximum drawdown occurred for all strategies. During this drawdown, MVP had the lowest drawdown of -19.83% while A2C and CAC40 had larger drawdowns. MVP recovered very rapidly compared to the others and then began to outperform them for a short period until August when another drawdown occurred. During this second drawdown, the other strategies recovered rapidly compared to MVP and MVP began to perform the worst in terms of cumulative returns. The other strategies (A2C, DDPG and CAC40) had similar cumulative returns until mid-May when DDPG and A2C slightly outperformed the CAC40 index and clearly outperformed the MVP baseline. Additionally, it can be observed that MVP was clearly less volatile compared to the other strategies.

DAX:

	A2C	**TD3**	**DDPG**	**PPO**	**SAC**	**DAX**	**MVP**
Cumulative returns	16.43%	**26.36%**	17.60%	14.05%	12.51%	15.90%	6.63%
Annual volatility	18.36%	18.55%	17.24%	17.71%	17.92%	18.69%	**14.13%**
Sharpe ratio	0.43	**0.61**	0.47	0.39	0.36	0.42	0.26
Max drawdown	-25.06%	-23.88%	**-20.71%**	-26.46%	-27.44%	-26.40%	-26.20%
Sortino ratio	0.62	**0.87**	0.68	0.56	0.51	0.59	0.37

Table 6.4: Performance of DRL algorithms compared to DAX

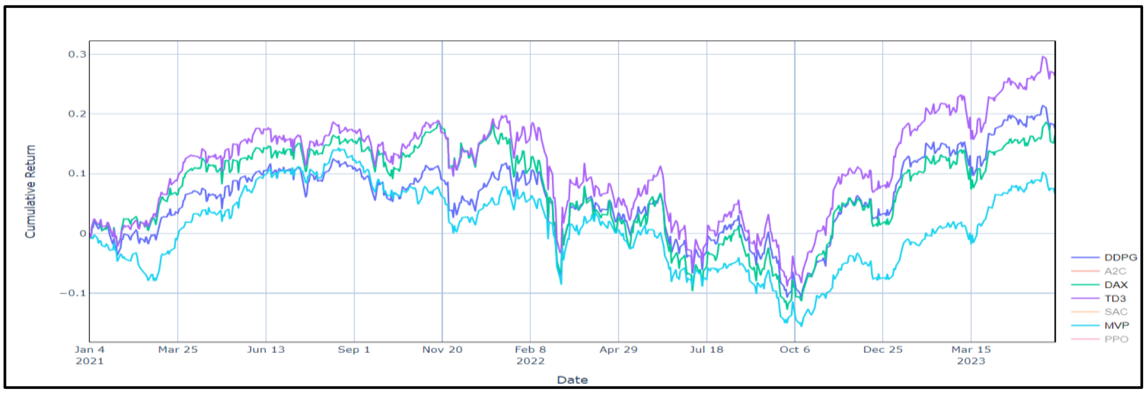


Figure 6.3: Cumulative Returns of DRL Strategies (DDPG and TD3) vs Baselines

Table 3, shows that the DRL algorithms achieved notable results in terms of cumulative returns, risk management, and portfolio efficiency when compared to the market index represented by the DAX.

In terms of cumulative returns, all DRL algorithms, with the exception of SAC, surpassed the market index (DAX) and the Minimum Variance Portfolio (MVP). TD3 achieved the highest cumulative return of 26.36%, indicating its exceptional ability to maximize profits. This represents a significant increase over the market index return of 15.90% and the MVP return of 6.63%.

When evaluating risk through annual volatility and maximum drawdown, the Deep Reinforcement Learning (DRL) algorithms showed risk levels that were comparable to the market index. However, DDPG stood out with a slightly lower risk profile, as indicated by its annual volatility of 17.24% and a maximum drawdown of -20.71%. This suggests that DDPG was able to effectively manage risk while still achieving high returns. In contrast, the Minimum Variance Portfolio (MVP) exhibited the lowest overall risk profile with an annual volatility of 14.13%.

In terms of portfolio efficiency, the DRL algorithms showed varied performance. TD3 achieved the highest Sharpe ratio of 0.61 and Sortino ratio of 0.87, suggesting that it was most effective in providing risk-adjusted returns and handling downside risk. This represents a significant improvement over the market index Sharpe ratio of 0.42 and Sortino ratio of 0.59.

In summary, while all DRL algorithms demonstrated promising results in the DAX market with higher returns and comparable risk to the market index and MVP, TD3 emerged as the leading performer in terms of both returns and efficiency. Its ability to achieve high returns while effectively managing risk makes it a strong choice for portfolio optimization in the DAX market. Figure 3 displays the cumulative returns of DAX, TD3, PPO, and MVP.

Until March 2022, DAX, TD3, and PPO had similar performance while MVP lagged behind. After that, the cumulative returns of all four algorithms were comparable and they all experienced a significant drawdown. After this drawdown, they slowly recovered and had similar cumulative returns until the end of December when TD3 slightly outperformed DAX and clearly outperformed PPO and MVP. The drawdown in March 2022 highlights the potential risks associated with using these algorithms for portfolio optimization. It is also worth noting that MVP demonstrated remarkable stability, exhibiting less volatility than the other algorithms throughout the entire period.

Figure 6.3 displays the cumulative returns of DRL strategies (DDPG and TD3) and the baselines (DAX and MVP). According to the results, TD3 outperformed all other strategies by a very clear margin for the entire period. MVP had almost the lowest cumulative return for the entire period.

From January 4th to February 8th, 2022, DAX outperformed DDPG, but then they had similar cumulative returns. In February 2022, during a period of maximum drawdown for all strategies, MVP and DAX had the maximum drawdown of -26% while DDPG had the lowest and recovered rapidly along with TD3 compared to the two baselines.

In summary, TD3 outperformed all other strategies in terms of cumulative returns, while MVP performed very poorly in comparison to the DAX index.

3. Discussion:

This study aims to detect whether Deep Reinforcement Learning (DRL) algorithms can pick up market trends and make profits on these. The analysis of the performance of different DRL algorithms in the DJIA, CAC40, and DAX markets suggests that these algorithms were able to achieve higher returns with similar levels of risk compared to traditional investment strategies such as the market index and the Minimum Variance Portfolio (MVP). This indicates that these algorithms were able to effectively incorporate available information and make profitable trading decisions.

When considered in the context of the Efficient Market Hypothesis (EMH), which proposes that asset prices reflect all available information at any given time, these results are particularly interesting. According to the EMH, it should not be possible to consistently outperform the market by using publicly available information.

However, the analysis suggests that the DRL algorithms were able to outperform traditional investment strategies, potentially indicating that these algorithms were able to process and incorporate information more effectively.
Based on the performance of various DRL algorithms in managing portfolios, it seems that these models are capable of identifying and exploiting market inefficiencies that are not accounted for by the EMH. This contradicts the EMH presented in this thesis. The fact that these models use historical closing prices and other relevant factors to learn suggests that they even conflict with the weakest form of the hypothesis.
It is worth noting that this approach is not widely used in practice. As such, the above-average performance of these models could be attributed to their limited use by investors. If more investors begin to adopt these models, their excess returns over the market portfolio may be arbitraged away, which would be consistent with the EMH.
However, the approach implemented in this study has some **limitations** that can affect its performance and impact its ability to accurately reflect real-world conditions:
The testing period could be extended to assess the long-term value of the portfolio more robustly: This means that by extending the testing period, the study could more accurately evaluate the long-term performance of the portfolio constructed using the approach.
The frequent rebalancing of market capitalization-weighted portfolios (DAX, CAC40 and DJIA30) can lead to high transaction costs, which is not implemented in the presented models. This means that the models do not take into account the costs associated with buying and selling securities when rebalancing the portfolio. Similarly, the minimum variance portfolio (MVP) does not consider transaction costs, which can impact its performance in the real world.
Large-scale buying or selling of stocks can affect the market: When a large number of stocks are bought or sold, it can create demand or supply for those stocks, which can drive up or down their prices.
Overall, this analysis provides valuable insights into the potential of Deep Reinforcement Learning (DRL) algorithms to improve portfolio management and investment strategies. Further research could explore how these algorithms are able to outperform traditional investment strategies and whether their performance is consistent with the predictions of the Efficient Market Hypothesis (EMH). Additionally, it would be also interesting to compare the performance of different RL algorithms in this context, to determine which algorithm is best suited for portfolio management and investment strategies.

Conclusion

This report has explored the use of reinforcement learning (RL) algorithms for portfolio optimization in four different markets: CAC40, DJ30 and DAX30. The performance of five RL algorithms: DDPG, PPO, A2C, SAC, and TD3, was compared to traditional portfolio optimization techniques such as the MVP and the market index in each market. Historical price data, technical indicators, and rolling covariance matrix were used as inputs for the RL models. The models were trained to maximize the portfolio value as the reward function.
The main findings of this thesis are that the RL models outperformed traditional methods in terms of cumulative return, Sharpe ratio, and maximum drawdown. The best performing RL model was TD3 in the DAX market, DDPG in the CAC40 market and SAC in the DJIA market. The RL models were able to adapt to changing market conditions and capture complex dynamics of financial markets. They were also able to handle high-dimensional state and action spaces.
The main contributions of this thesis are that it provides a comprehensive and comparative study of RL algorithms for portfolio optimization in different markets. It introduces a novel state representation that incorporates both technical indicators and risk information using rolling covariance matrix. It also demonstrates the effectiveness and robustness of RL techniques for portfolio optimization. Additionally, this thesis compares the performance of DRL algorithms against the market index in relation to the Efficient Market Hypothesis (EMH), providing valuable insights into the potential limitations of this hypothesis.

The main limitations and future directions of this thesis are that it assumes a zero risk-free rate and no market impact, which may not reflect realistic scenarios. It does not consider other factors that may affect portfolio performance, such as transaction costs, slippage, liquidity, etc. It also does not account for market efficiency and its implications for portfolio optimization.
Future research can extend the RL framework to include more realistic assumptions and constraints, as well as other reward functions and performance metrics. Future research can also explore other RL algorithms and techniques.

References

[1] Bodie, Zvi, Alex Kane, and Alan Marcus. EBOOK: Investments-Global edition. McGraw Hill, 2014.
[2] Cagliero, L., Fior, J., & Mangia, F. (2022). Financial data analysis by means of Reinforcement Learning techniques.
[3] Cartanyà Caro, P. (2022). A deep learning approach to portfolio optimization (Bachelor's thesis, Universitat Politècnica de Catalunya).
[4] Charpentier, A., Elie, R., & Remlinger, C. (2021). Reinforcement learning in economics and finance. Computational Economics, 1-38.
[5] de Prado, M. M. L. (2020). Machine learning for asset managers. Cambridge University Press.
[6] Del Nobile, F. (2021). Efficient portfolio allocation: reinforcement learning methods applied to modern finance.
[7] Faturohman, T., & Nugraha, T. (2022). Islamic Stock Portfolio Optimization Using Deep Reinforcement Learning. Journal of Islamic Monetary Economics and Finance, 8(2), 181-200.
[8] Filos, A. (2019). Reinforcement learning for portfolio management. arXiv preprint arXiv:1909.09571.
[9] Fontana Miyoshi Bianchi, R. (2022). Reinforcement learning for portfolio optmization (Master's thesis, Universitat Politècnica de Catalunya).
[10] Frend, Yoav, and Robert E. Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting." Journal of computer and system sciences 55.1 (1997): 119-139.
[11] Fama, Eugene F. "Efficient capital markets: A review of theory and empirical work." The journal of Finance 25.2 (1970): 383-417.
[12] Guan, M., & Liu, X. Y. (2021, November). Explainable deep reinforcement learning for portfolio management: an empirical approach. In Proceedings of the Second ACM International Conference on AI in Finance (pp. 1-9).
[13] Gunjan, A., & Bhattacharyya, S. (2023). A brief review of portfolio optimization techniques. Artificial Intelligence Review, 56(5), 3847-3886.
[14] Haarnoja, Tuomas, et al. "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor." International conference on machine learning. PMLR, 2018.
[15] Hieu, L. T. (2020). Deep reinforcement learning for stock portfolio optimization. arXiv preprint arXiv:2012.06325.
[16] Hongjoong, K. I. M. (2021). MEAN-VARIANCE PORTFOLIO OPTIMIZATION WITH STOCK RETURN PREDICTION USING XGBOOST. Economic Computation & Economic Cybernetics Studies & Research, 55(4).
[17] Kobets, V., & Savchenko, S. Building an Optimal Investment Portfolio with Python Machine Learning Tools.
[18] Lavko, M., Klein, T., & Walther, T. (2023). Reinforcement Learning and Portfolio Allocation: Challenging Traditional Allocation Methods. Queen's Management School Working Paper, 1.
[19] Li, X., Li, Y., Zhan, Y., & Liu, X. Y. (2019). Optimistic bull or pessimistic bear: Adaptive deep reinforcement learning for stock portfolio allocation. arXiv preprint arXiv:1907.01503.
[20] LILLO, F. Deep Learning methods for Portfolio Optimization.
[21] Liu, X. Y., Yang, H., Chen, Q., Zhang, R., Yang, L., Xiao, B., & Wang, C. D. (2020).

FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance. arXiv preprint arXiv:2011.09607.

[22] Liu, X. Y., Yang, H., Gao, J., & Wang, C. D. (2021, November). FinRL: Deep reinforcement learning framework to automate trading in quantitative finance. In Proceedings of the second ACM international conference on AI in finance (pp. 1-9).

[23]Luthfianti, Bellatris Aprila, Deni Saepudin, and Aditya Firman Ihsan. "Portfolio Allocation of Stocks in Index LQ45 using Deep Reinforcement Learning." *2022 10th International Conference on Information and Communication Technology (ICoICT)*. IEEE, 2022.

[24] Ma, Y., Han, R., & Wang, W. (2021). Portfolio optimization with return prediction using deep learning and machine learning. Expert Systems with Applications, 165, 113973.

[25] Markowits, H. M. (1952). Portfolio selection. Journal of finance, 7(1), 71-91.

[26] Moghar, A., & Hamiche, M. (2020). Stock market prediction using LSTM recurrent neural network. Procedia Computer Science, 170, 1168-1173.

[27] Obeidat, S., Shapiro, D., Lemay, M., MacPherson, M. K., & Bolic, M. (2018). Adaptive portfolio asset allocation optimization with deep learning. International Journal on Advances in Intelligent Systems, 11(1), 25-34.

[28] Pretorius, R., & van Zyl, T. (2022). Deep reinforcement learning and convex mean-variance optimisation for portfolio management. arXiv preprint arXiv:2203.11318.

[29] Qian, Edward. "Risk parity and diversification." The Journal of Investing 20.1 (2011): 119-127.

[29] Sadriu, L. (2022). Deep Reinforcement Learning Approach to Portfolio Optimization.

[30] Schulman, John, et al. "Proximal policy optimization algorithms." arXiv preprint arXiv:1707.06347 (2017).

[31] Sen, J., Dutta, A., & Mehtab, S. (2021, October). Stock portfolio optimization using a deep learning LSTM model. In 2021 IEEE Mysore Sub Section International Conference (MysuruCon) (pp. 263-271). IEEE.

[32] Sharpe, William F. "Mutual fund performance." The Journal of business 39.1 (1966): 119-138.

[33] Silver, David. "Lectures on reinforcement learning." URL: https://www. davidsilver. uk/teaching 471 (2015).

[34] Sood, S., Papasotiriou, K., Vaiciulis, M., & Balch, T. (2023). Deep Reinforcement Learning for Optimal Portfolio Allocation: A Comparative Study with Mean-Variance Optimization. FinPlan 2023, 21.

[35] Sun, G. Portfolio Optimization Based on Deep Reinforcement Learning.

[36] Sun, S., Wang, R., & An, B. (2023). Reinforcement learning for quantitative trading. ACM Transactions on Intelligent Systems and Technology, 14(3), 1-29.

[37] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

[38] Szepesvári, C. (2022). Algorithms for reinforcement learning. Springer Nature.

[39] Szepesvári, Csaba, and Michael L. Littman. "Generalized markov decision processes: Dynamic-programming and reinforcement-learning algorithms." Proceedings of International Conference of Machine Learning. Vol. 96. 1996.

[40] Tatsat, Hariom, Sahil Puri, and Brad Lookabaugh. Machine Learning and Data Science Blueprints for Finance. O'Reilly Media, 2020.

[41] Wang, Wuyu, et al. "Portfolio formation with preselection using deep learning from long-term financial data." *Expert Systems with Applications* 143 (2020): 113042.

[41] Wu, W., & Hargreaves, C. Deep Reinforcement Learning Approach to Portfolio Optimization in the Australian Stock Market. Available at SSRN 4429448.

[42] Yu, X., Wu, W., Liao, X., & Han, Y. (2023). Dynamic stock-decision ensemble strategy based on deep reinforcement learning. Applied Intelligence, 53(2), 2452-2470.

[43] Zhang, Y., Su, Y., Liu, W., & Yang, X. Portfolio Optimization with Lstm-Based Return and Risk Information. Available at SSRN 4215299.

[44] Zhang, Y., Su, Y., Liu, W., & Yang, X. Portfolio Optimization with Lstm-Based Return and Risk Information. Available at SSRN 4215299.
[45] Zrara, L. (2020). PORTFOLIO OPTIMIZATION USING DEEP LEARNING FOR THE MOROCCAN MARKET.
[46] Sen, J., Dutta, A., & Mehtab, S. (2021, October). Stock portfolio optimization using a deep learning LSTM model. In 2021 IEEE Mysore Sub Section International Conference (MysuruCon) (pp. 263-271). IEEE.
[47] Sahu, S. K., Mokhade, A., & Bokde, N. D. (2023). An Overview of Machine Learning, Deep Learning, and Reinforcement Learning-Based Techniques in Quantitative Finance: Recent Progress and Challenges. Applied Sciences, 13(3), 1956.