Facultad de Ciencias Económicas y Empresariales

Doble Grado en ADE y Business Analytics (E-2 Analytics)

Entropy Coefficients and portfolio management in the context of DRL

Author: Roberto Gozalo Brizuela
Director: Eduardo César Garrido Merchán

MADRID| April 2024

Resumen

En los últimos años, los académicos han investigado la aplicación de modelos de aprendizaje profundo por refuerzo (DRL, por sus siglas en inglés) en el ámbito de la gestión de carteras de inversión. Una de las principales razones por las que estos modelos demuestran un rendimiento superior es por la tecnología subyacente de aprendizaje profundo, que les permite identificar patrones en los datos que los modelos tradicionales a menudo pasan por alto. A pesar de la alta demanda computacional y la complejidad asociada con las redes neuronales, importantes instituciones financieras como JP Morgan han implementado con éxito estos modelos.

Dada la rápida evolución en este campo, ha habido un creciente interés en comprender cómo los diversos hiperparámetros afectan el rendimiento del modelo. En particular, hay poca investigación sobre el impacto del hiperparámetro del coeficiente de entropía, un elemento crítico en la configuración de modelos DRL. Este estudio tiene como objetivo resolver este problema al investigar el valor óptimo del hiperparámetro del coeficiente de entropía y analizar cómo los cambios en este parámetro influyen en el rendimiento del modelo. Esta investigación utiliza un conjunto de datos comprensivo de 15 años del índice Dow Jones, proporcionando un contexto robusto para nuestros experimentos y conclusiones. El objetivo final es ofrecer información que pueda guiar a los profesionales y académicos en la mejora de los modelos DRL para obtener mejores resultados en la gestión de carteras de inversión.

Palabras clave: Aprendizaje Profundo, Gestión de Carteras, Aprendizaje Profundo de Refuerzo, PPO, Ratio de Sharpe

Abstract

In recent years, scholars have dived deep into the application of deep reinforcement learning (DRL) models in portfolio management. One of the main reasons these models perform better is due to the underlying deep learning technology, which allows them to identify patterns in data that traditional models often overlook. Despite the high computational demands and complexity associated with neural networks, significant financial institutions like JP Morgan have successfully implemented these models.

Given the rapid advancements in this area, there has been a growing interest in understanding how various hyperparameters affect model performance. Notably, there is limited research on the impact of the entropy coefficient hyperparameter, a critical element in the configuration of DRL models. This study aims to solve that issue by investigating the optimal value of the entropy coefficient hyperparameter and analyzing how changes to this parameter influence the model's performance. This investigation uses a comprehensive 15-year dataset from the Dow Jones index, providing a robust context for our experiments and conclusions. The ultimate goal is to offer a clear analysis on this particular hyperparameter that could guide practitioners and researchers in refining DRL models for better portfolio management outcomes.

Keywords: Deep Learning, Portfolio Management, Deep Reinforcement Learning, PPO, Sharpe Ratio

Acknowledgments

Thanks to my family and friends. Also, thanks to my tutor who has helped me a lot throughout the completion of my work.

Contents

List of Figures

List of Tables

List of Algorithms

List of Equations

Chapter 1

Introduction

Portfolio management involves handling funds and obtaining returns while minimizing risk by investing them in different types of assets (Kapoor, 2014). The investment portfolio management industry has a size of five trillion dollars (Insights, 2023).

Traditional portfolio management approaches structured the allocation of weights for each asset through optimization problems. However, they suffer from a lot of rigidity both in achieving desired objectives and in adapting to changes over time (S. Yang, 2023). Other more traditional approaches that do not require optimization include Markowitz and the Capital Asset Pricing Model (CAPM). Despite these approaches having solid theoretical foundations, they make implicit assumptions such as assuming normal distributions in financial assets and assuming stable correlations (de Inversión, 2023; Ouyang, 2022). This leads to excessive rigidity. Machine Learning models, especially those using neural networks, offer much more flexibility in their predictions, alleviating the problems mentioned above (M. M. Fischer et al., 2015).

Quantitative finance and big data have made automated portfolio management imperative for investment firms (SUN & AN, 2020). The traditional approach with a fixed portfolio of stocks is no longer sufficient due to the advent of new technologies. Recently, artificial intelligence has been introduced into the Asset Management industry in companies such as JP Morgan, Morgan Stanley, and Vanguard Group (News, 2023). This is because Artificial Intelligence has increased efficiency, returns, and regulatory compliance (Bartram, 2020). There is great value in investigating how to maximize the efficiency of these investment portfolio creation processes. Lately, we have seen through ChatGPT how artificial intelligence can have comparable performance to humans in the

field of finance (Koubaa et al., 2023). Thus, it seems reasonable for us to test whether AI can also make better decisions in the field of portfolio management, where many psychological biases come into play that affect human judgement.

DRL eliminates the issue of having to predict future stock market prices and just focuses on allocating resources as efficiently as possible (Gu, Jiang, & Su, 2021). DRL systems learn through actions and rewards (François-Lavet et al., 2018) and they have been proven as useful through several past studies (Jiang, Xu, & Liang, 2017). Through this study, we will use this technology to create efficient asset allocation models.

Upon reviewing prior research on the application of Deep Reinforcement Learning (DRL) in portfolio management, we noted a consistent use of the default entropy coefficient across all studies. This observation sparked our curiosity regarding the adequacy of this conventional approach. Consequently, we recognized the significance of investigating whether modifying the entropy coefficient could potentially result in the development of a more effective model. Thus, our research tries to explore different values of the entropy coefficient, trying to undersand what the optimal value would be. As such, we intend to conduct experiments testing different values of the entropy coefficient for both the Proximal Policy Optimization (PPO) model. Several studies such as (Ahmed, Le Roux, Norouzi, & Schuurmans, 2019; Berner et al., 2019; Eimer, Lindauer, & Raileanu, 2023; Olsson, Malm, & Witt, 2022) suggest changes in model performance when optimizing the entropy coefficient. They make us believe that an experiment like ours may yield better results than the default entropy coefficient value.

Chapter 2

State of the Art Review

The landscape of trading strategies has seen a huge rise of Machine Learning techniques. Traditional approaches, from Meta-Learning strategies to Pattern Matching Algorithms, faced limitations in aligning with investor goals and leveraging valuable insights from data. In response, Reinforcement Learning (RL) has emerged as a promising paradigm to overcome these challenges (T. G. Fischer, 2018; Jiang et al., 2017). This section surveys the evolution of RL in trading, from works like Neuneier's critic-only approach in 1996 (Neuneier, 1995) to contemporary deep RL frameworks. These studies highlight the versatility of RL, from actor-only, actor-critic, and deep RL methodologies, as well as innovations like Adversarial RL and explainable DRL. From integrating market sentiment (Koratamaddi, Wadhwani, Gupta, & Sanjeevi, 2021) to dynamic risk-sensitive allocation (Yu, Lee, Kulyatin, Shi, & Dasgupta, 2019) and cost-sensitive optimization (Zhang et al., 2020), RL offers a comprehensive toolkit for portfolio management. Furthermore, ensemble methods (H. Yang, Liu, Zhong, & Walid, 2020) and multi-agent frameworks (Lin, Chen, Sang, & Huang, 2022) take in several models and improve performance. In figure 2.1, we can see an outlook of this state-of-the-art review.

Traditional trading strategies can be divided into Meta-Learning trading strategies that combine multiple strategies in order to achieve better performance and Pattern Matching Algorithms that predict the next market distribution of data through learning from historical data (Jiang et al., 2017). There are several limitations of the traditional trading strategies that the Reinforcement Learning approach seeks to overcome(T. G. Fischer, 2018). First, the objective of the optimization is not always in line with the investor's goals. Second, information from the feature space that draws insights can be

very valuable for the investing strategy. Third, exogenous constraints imposed by the environment such as lack of liquidity and transaction costs are normally not considered. Fourth, the Reinforcement Learning approach combines the portfolio construction and the prediction tasks of portfolio management. The two-step approach may lead to sub-optimal performance (Moody, Wu, Liao, & Saffell, 1998).

The first research project that used Reinforcement Learning came in 1996 with (Neuneier, 1995) that proposed a critic-only approach to reinforcement learning applied to portfolio management. Through Artificial FX data and the German Stock Index DAX, this actor-critic model outperformed supervised learning and other traditional models. It also found that volatility could be reduced through a penalty term. (Dempster, Payne, Romahi, & Thompson, 2001) found out that genetic algorithms can improve the performance of RL systems (T. G. Fischer, 2018). In this paper popular technical indicators are used as input, seeking a profitable trading rule using them. The paper compares a Markov Decision Process, Genetic Programming (GP), heuristic decision and reinforcement learning (RL) backtesting with a dataset of GBPUSD FX data. It finds out that the RL and the GP are the best-performing techniques when looking at the Sharpe Ratio. (J. W. Lee, Park, O, Lee, & Hong, 2007) created a divide and conquer strategy in which multiple agents were used for portfolio optimization. The trading tasks of the identification of opportunities and the determination of prices were divided amongst the agents (T. G. Fischer, 2018). The proposed framework uses multiple Q-learning agents allowing them to divide and conquer the trading problem by dividing roles and collaborating in the process of stock picking. Experiments on the korean stock trading market show the outperformance of this method in comparison to other alternative trading approaches. More recently, (Eilers, Dunis, von Mettenheim, & Breitner, 2014) demonstrated that Reinforcement Learning can be applied to trading strategies by taking advantage of seasonal effects. The study uses the Q-Learning algorithm, training the agent so it take advantage of seasonal events such as holidays that drive stocks up, upward biases at the end of the month, and pre-FOMC announcement changes. Backtesting with S&P 500 and DAX historical stock price data, the model clearly outperforms the baseline strategy.

In the line of actor-only agents, (Moody & Wu, 1997) proposed the first actor-only Reinforcement Learning approach for Portfolio Management. With a dataset of monthly S&P 500 data and a dataset of 30-minute FX data, it proved that Recurrent Reinforcement Learning can be used when you have a portfolio with only a risky asset and a risk-free asset. (T. G. Fischer, 2018) (Dempster & Leemans, 2006) puts its focus on

risk as it proves that actor-only systems can be embedded in a multi-layered risk management system. The paper introduces Adaptive Reinforcement Learning for portfolio management, in which hyperparameters are dynamic instead of fixed. Tested in foreign exchange markets, this framework enhances the performance of traditional trading systems. More recently, (Deng, Kong, Bao, & Dai, 2015) proved that the performance of actor-only models can be improved by introducing sparse-coded, task-specific feature representations. The model creates a very rich state representation with more than 80 variables including volume patterns, technical indicators as well as order-book level features. Backtesting experiments were performed on the high-frequency data of Shangai IF, the most liquid chinese derivative. The model proved itself to be very robust and to reduce noise in predictions in comparison to past trading strategies.

The actor-critic was first introduced into the portfolio management problem (Li, Dagli, & Enke, 2007). This paper found out that actor-critic models outperform actor-only models and the supervised learning benchmark.(T. G. Fischer, 2018) Specifically, it found out that actor-critic models were more succesful at short-term prediction than the actor-only model through three stock price historical datasets: SP500, NASDAQ and IBM.

Further research focuses more highly on Deep Reinforcement Learning, the combination of Deep Learning and Reinforcement Learning (François-Lavet et al., 2018). Deep Learrning models train train multiple neural networks, which can bring instability to the model (Jiang et al., 2017). In a stable model, performance metrics are similar when testing with two different validation sets of data (Ling, 2019). That is why when analyzing each of the state-of-the-art models in this field, it's worth taking a look at whether they proved to be robust.

In this line of research, (Jiang et al., 2017) explores a deep reinforcement learning framework for the financial portfolio management problem. The proposed framework in this paper consists of using "the Ensemble of Identical Independent Evaluators (EIIE) topology, a Portfolio-Vector Memory (PVM), an Online Stochastic Batch Learning (OSBL) scheme, and a fully exploiting and explicit reward function." With this framework, it does experiments using a Convolutional Neural Network (CNN), a basic Recurrent Neural Network (RNN) and a Long-Short Term Memory (LSTM). Using a dataset with cryptocurrency prices datasets, it performs backtesting in order to test out the model's accuracy. All three algorithms performed better than the other traditional trading strategies against which this was tested including UCRP and UBAH.

Moreover, this project (Liang, Chen, Zhu, Jiang, & Li, 2018) explores Adversarial Deep Reinforcement learning for financial management, implementing three algorithms: Policy Gradient (PG), Deep Deterministic Policy Gradient (DDPG), and Proximal Policy Approximation (PPO). While they were tested for game playing and robot control, they hadn't yet been tested in a portfolio management problem. Intensive experiments were conducted under a stock market prices dataset. Performance metrics show that the PG algorithm performed the best out of the three of them. The experiment also finds the effectiveness of DRL algorithms in capturing market patterns. This is a great indicator that our model will be successful at capturing market patterns, thus conducting a profitable market strategy.

Furthermore, this research (Yu et al., 2019) explores the performance of a DRL framework in Dynamic Portfolio Management problem, when wealth has to be sequentially allocated. The framework consists of an infused prediction module (IPM), a generative adversarial data augmentation module (DAM) and a behavior cloning module (BCM). Using a financial market dataset, the model is tested. The DRL model is robust, profitable and risk-sensitive when matched against other trading strategies and RL models. The paper also approaches this problem using a PPO algorithm, obtaining state-of-the-art performance. This is very much encouraging for the thesis a PPO algorithm in a financial markets dataset will be applied.

In addition, this line of research (Zhang et al., 2020) includes a cost component into the DRL algorithm in a portfolio management problem. This component includes both risk-related costs and transaction-related costs. A novel policy is designed in order to extract asset correlations and price series patterns. Also, a function is developed in order to maximize returns while constraining costs. This model is tested in a real-world financial markets dataset, finding out that this methodology provides profitability, cost-sensitivity and representation abilities. This proves that DRL methodologies can be used for managing the risk in a portfolio, something we will try to achieve through this thesis.

Also, this research project creates a DRL-based approach called DeepTrader (Wang, Huang, Tu, Zhang, & Xu, 2021) for Risk-Return balanced portfolio management. The proposed methodology includes macroeconomic variables to change the long-short proportion of stocks to lower risk, including the negative maximum drawdown as the reward function. Moreover, the model incorporates a unit to assess individual assets, using price rising rate as the reward function. Experiments conducted with past financial markets data demonstrate the effectiveness of this approach, being specially effective during the

great financial crisis and its recovery period. Through this study, we find proof that DRL-based algorithms are able to manage risk in portfolios effectively and specially during economic downturns.

Explainable approaches to DRL portfolio management have also been explored. This paper (Guan & Liu, 2021) models a DRL agent with integrated gradients for feature weights, testing its performance in a Dow Jones return dataset from 2009 to 2021. They do this for both single-step and multi-step prediction. When benchmarking the model with other traditional machine learning models, the DRL agent shows a stronger performance than traditional machine learning models in multi-step prediction.

This paper (Koratamaddi et al., 2021) researches including market sentiment for DRL agents in stock portfolio allocation. This project proposes a approach to train an intelligent automated trader that uses historical stock data and market sentiment for a portfolio of Dow Jones stocks. They do sentiment analysis examining market news through Twitter and Google News. Through the Sharpe Ratio, they prove that their approach performs better than a vanilla DDPG, an adaptive DDPG, a mean-variance analysis, and a minimum variance analysis.

This research project (Soleymani & Paquet, 2021) explores deep graph convolutional reinforcement learning for financial portfolio management with a model called DeepPocket. The model extracts low-dimensional features through a Restricted Stacked Autoencoder. They find correlation among the financial instruments using the deep graph convolutional network. An Actor Critic method is used to exploit the investment policy. Results show this model clearly outperforms financial indexes through five datasets over three distinct investment periods.

AlphaPortfolio (Cong, Tang, Wang, & Zhang, 2021) is a Deep Reinforcement Learning framework proposed through this research that employs a Transformer Encoder and Long-Short Term Memory for feature representation. It uses a novel approach to attention mechanisms that they call Cross-Asset Attention Network in order to obtain the asset's weights. It uses a dataset of the NYSE market from 1965 to 2016 because of the requirement for a large amount of data that Deep Learning models have. The agent is tested with different trading and economic restrictions, yielding superb returns, proving that the model is robust. Researchers did also learn very valuable insights about stocks, through the feature representation that these models produce.

Moreover, this research (Benhamou, Saltiel, Ungari, & Mukhopadhyay, 2020) project hedges risk with a DRL-based model creating a dependency between market informa-

tion and hedging strategies. The model adds contextual information related to current financial indicators in order to train the model. Through different reward functions including net profit and sortino ratio, this strategy obtains higher returns and lower risk than the Markowitz approach. This encourages us to think that an approach different to the traditional industry-wide Markowitz approach can be more profitable.

This research line (Lim, Cao, & Quek, 2022) tries to maximize returns for a portfolio through a DRL-based algorithm. It uses a Q network with reward function that includes dynamically calculating the Net Asset Value of the portfolio, the model gets rewarded for its actions. The paper tests four methods: an LSTM-based DRL agent with gradual rebalancing, an LSTM-based DRL agent with full rebalancing, gradual rebalancing without an LSTM-based DRL agent and full rebalancing without an LSTM-based DRL agent. The proposed agent shows the ability to adapt to the market conditions and shows good performance. The gradual rebalancing as well shows better performance than full rebalancing. In our model, we will use elements in order to assure a proper gradual rebalancing of the portfolio, thus this paper reassures us of our methodology.

In this line of papers try to optimize for a certain portfolio metric,(Zhao, Ma, Li, & Zhang, 2023) should be mentioned as it optimizes for the asset's correlation. This model includes nonlinearity relationships between the assets through an attention-based mechanism. The algorithm chosen for training is a deterministic policy gradient recurrent reinforcement learning method based on Monte Carlo sampling. The method obtains state-of-the-art performance in return compared to traditional and advanced methodologies.

Even Morgan has published research using DRL for portfolio management. In a recent study (Sood, Papasotiriou, Vaiciulis, & Balch, 2023), they compare the performance of an DRL agent to a traditional Mean-Variance Optimization approach. This framework uses a PPO agent as its algorithm and Sharpe Ratio as the reward. Because of both appraoches algortihms optimizing for the Sharpe Ratio, we can now compare which model is more efficient. This approach shows better Annual Returns, Maximum Drawdown and Sharpe ratio than the traditional Mean-Variance optimization approach. As well, the DRL agent shows more consistent returns and a more stable portfolio.

More recent papers trying to optimize DRL-based models' performance include (S. Yang, 2023), which proposes a Task-Context Mutual Actor–Critic (TC-MAC) algorithm. Through a financial market setting, this algorithm shows superior performance to other traditional portfolio management algorithms. It as well shows transferability to

other problems.

Ensemble approaches (H. Yang et al., 2020) that combine more than one model for the final prediction have as well been researched. This paper combines three actor-critic based algorithms: Advantage Actor Critic (A2C), Proximal Policy Optimization (PPO) and Deep Deterministic Policy Gradient (DDPG). They are back-tested with historical stock price data of 30 Dow Jones companies. The performance of the ensemble is better than the three individual algorithms as well as two other baselines (the Min-Variance approach and the Dow Jones Index Return).

In this line of research, (Lin et al., 2022) proposes a multi-agent DRL approach for risk-shifting portfolio management. Each agent has a refined deep policy network and a special training method that makes the DRL agent to learn risk transfer behavior. This model outperforms other traditional strategies in the Sharpe Ratio.

Deep Reinforcement Learning has also been applied in the field of Economics (Charpentier, Elie, & Remlinger, 2021), a field very closely related to portfolio management. This article uses DRL for optimal control problems in economics, game theory, operations research, and finance. Despite not providing results for each selected task, they conclude that while advances are very promising in this field for making the models, computational capabilities are slowing progress down. They also note that these models assume that a lot of information is available, something that is not always true.

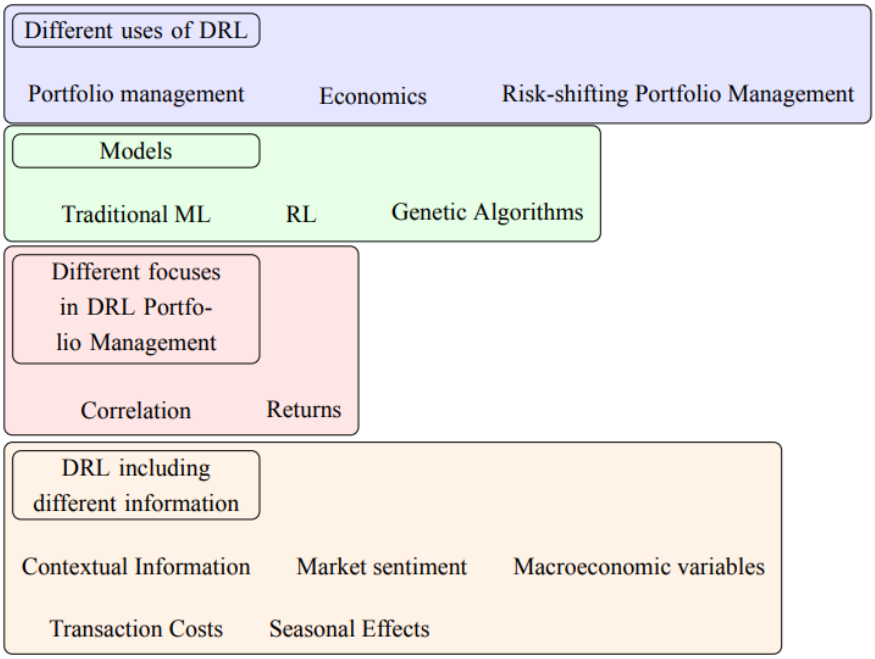

Figure 2.1: Organization of the State of the art review

Chapter 3

Thesis definition

This section outlines the scope and framework of the thesis, focusing on the development of a software code employing Deep Reinforcement Learning (DRL) for financial market analysis. Key objectives include modifying the open-source Stable Baselines algorithm to evaluate various entropy coefficients' effectiveness. Additionally, the constraints, hypothesis, and underlying assumptions crucial to the study are detailed, providing a comprehensive foundation for the research.

3.1 General Objective

The general objective of this work is the utilization of Deep Reinforcement Learning for the creation of diversified investment portfolios. Specifically, the Proximal Policy Optimization (PPO) technique will be employed. Additionally, a new methodology for quantifying portfolio diversification will be introduced and compared with the traditional approach. The work will conclude with hyperparameters that optimize portfolio behavior and determine if our methodology is more efficient than the previous one. The specific objectives are as follow:

3.2 Specific goals

1. Analyze the importance of using deep learning in the field of portfolio management, using reinforcement learning to verify its value.

2. Evaluate the performance of DRL algorithms testing for different entropy coefficients.

3. Provide a practical framework for the utilization of Deep Learning algorithms in the field of finance, especially in portfolio optimization.

4. Train a PPO model for the optimization of a portfolio with the assistance of stock market data from recent years.

3.3 Constraints

1. Time Limitation: The thesis must be completed within the academic year 2023-2024, affecting both the model training and its subsequent analysis.

2. Computational Limitation: The model will be executed using Google Collaboratory, utilizing Google's hosted computing resources, which offer more flexibility than personal computing but within Google's closed environment.

3. Data Collection: Due to computational and time constraints, the dataset used for testing will cover a limited period.

4. Measurement Errors: The dataset consists of close prices with a daily frequency, potentially introducing measurement inaccuracies.

3.3.1 Hypothesis

In this study, we provide a statistical analysis of the entropy regularizer for financial portfolio optimization problems. The hypothesis to be tested is that the entropy required for these financial problems obtains a better performance than the default value which is traditionally used. Because of this, we frame the problem as follows:

> **Null Hypothesis:** H_0 : entropy_coefficient $\leq$ default_entropy_coefficient **Alternative Hypothesis:** H_1 : entropy_coefficient $>$ default_entropy_coefficient

3.3.2 Assumptions

1. Transferability: Our model's behavior observed in our dataset is assumed to generalize to other financial market scenarios, making our findings relevant for practitioners applying DRL agents to portfolio optimization problems.

2. Model Assumptions: We assume past returns predict future returns, neglect transaction costs and taxes, assume trading doesn't impact pricing, consider investments infinitely divisible, and understand that investors are price takers. We also assume that trades can take place regardless of time, day, and share quantity.

3. Underlying Theory: We assume that the theory behind Deep Reinforcement Learning theory.

Chapter 4

Theoretical background

4.1 Deep Learning

The Deep Learning paradigm is the basis of the architecture of the PPO model, with neural networks at its core.Understanding the fundamental structure of a Deep Learning model is imperative.

The backpropagation algorithm discovers deep patterns in high dimensional data by adjusting parameters (LeCun, Bengio, & Hinton, 2015). Essentially, the structure of deep learning is composed of a multilayer stack of modules, with some modules learning information while others create non-linear input-output mappings of the data. Known as Multilayer Neural Networks, these multilayer stacks stack neural networks on top of one another. Using feedforward neural networks, these architectures involve a set of units computing a weighted sum of their inputs from the previous layer of neurons and passing the output through a non-linear function to traverse through the different layers.

Despite Deep Learning initially gaining traction in disparate fields such as speech recognition, object recognition, and object detection, it has made strides in Portfolio Management due to many of the same characteristics that made it so powerful in its initial domains of study. Forecasting models have a lot of substantial noise, and neural networks offer a potential solution as they have demonstrated an ability to discern underlying patterns in datasets. As an example, RL integrates both prediction and portfolio construction tasks into a unified process, aligning closely with investors' objectives. Because of this architecture, critical constraints such as transaction costs, market liquidity, and investor risk aversion can be effectively addressed (T. G. Fischer, 2018).

4.2 Deep Reinforcement Learning

(Sutton & Barto, 2018) highlights the similarities between reinforcement learning and human learning. Just as individuals acquire skills like bicycling through hands-on practice, reinforcement learning operates similarly, without actively considering principles like angular momentum, momentum, and force. Rather than addressing optimization challenges based on pre-established rules, an agent takes in the current state within a specified environment and aims to maximize rewards from available actions (Jang & Seong, 2023).

Q-learning isthe foundation of reinforcement learning (Hester et al., 2018). It is based on the Q-function, which represents the reward received by an agent when an action a is performed in a specific state s. In Q-learning, the Q function is updated by utilizing the greedy action $a(t+1)$ that maximizes the $Q(S_{t+1})$ function.

$$Q(s_t, a_t) = E[r(s_t, a_t, s_{t+1}) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})] \tag{4.1}$$

The Deep Q-Network (DQN), a reinforcement learning algorithm that incorporates a neural network and approximates the Q function. With this architecture, the state is encoded as a value function. However, the DQN approach becomes challenging to use as the amount of computation increases when the action space is large (Hester et al., 2018). In the stock market, the portfolio weight is not discrete and has an action space in the form of a continuous box, leading to infinite action spaces. Therefore, DQN is not deemed suitable for portfolio optimization. We can see the model's architecture through this 4.1 from (*Latex Figure for Deep Reinforcement Learning — tex.stackexchange.com*, n.d.).

4.3 PPO model

As outlined in the original paper by (Schulman et al., 2017), the objective of PPO was to achieve the data efficiency and reliability similar to TRPO, applying only first-order optimization techniques. A unique objective function with clipped probability ratios was introduced by PPO, fin order toi get a consertive setting of results. PPO represents an innovative category of policy gradient methodologies where data is once and again acquired through interactions with the environment, and a "surrogate" objective function

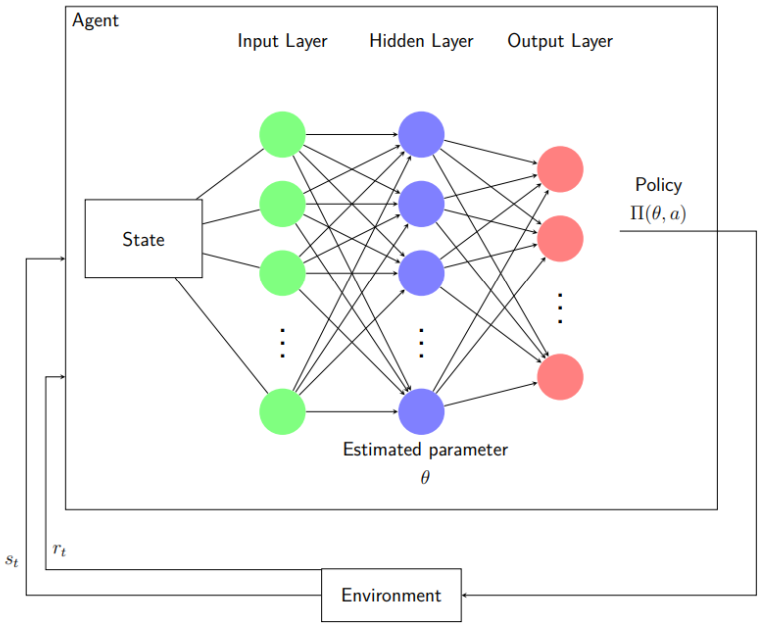

Figure 4.1: Deep Q-Network Architecture

is improved via stochastic gradient ascent. Unlike traditional policy gradient methods, where gradients are updated once per data sample, PPO uses an updated objective function that allows for several epochs of mini-batch updates (Schulman et al., 2017). This objective function of PPO can be represented as:

$$L(s, a, \theta_k, \theta) = \min \left(\pi_\theta(a|s), \frac{\pi_{\theta_{\text{old}}}(a|s)}{\pi_{\theta_k}(a|s)} \right) A^{\pi_{\theta_k}}(s, a) \tag{4.2}$$

$$g(\cdot, A) = \begin{cases} (1+\epsilon)A & \text{if } A \geq 0 \\ (1-\epsilon)A & \text{if } A < 0 \end{cases} \tag{4.3}$$

In its implementation, PPO keeps track of two policy networks. The initial one represents the current policy in need of refinement (Schulman et al., 2017).

$$\pi_\theta(a_t|s_t) \tag{4.4}$$

The policy used for sample collection previously:

$$\pi_{\theta_k}(a_t|s_t) \tag{4.5}$$

Switching between sampling data and interacting with the environment, PPO increases sample efficiency by evaluating a new policy using samples obtained from an earlier policy, employing the concept of significance sampling (Schulman et al., 2017).

Our goal is to maximize the expected surrogate objective function in order to optimize

$$_t \left[\pi_\theta(a_t|s_t)\frac{\pi_{\theta_{\text{old}}}(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}\hat{A}^{\pi_k}(s,a)\right] \quad (6.37)(4.6)$$

As the current policy is developed, the gap between it and the previous policy widens, leading to increased estimation variance and resulting in poor judgments due to inaccuracy. Consequently, every four iterations, the second network is remade with the revised policy (Schulman et al., 2017). Using a clipped objective, a ratio between the new policy and the old policy is computed (Schulman et al., 2017).

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} \quad (6.38) \tag{4.7}$$

The comparison of the two policies is represented by this ratio. When the new policy significantly deviates from the previous one, a modified objective function is generated to constrain the estimated advantage function. The resulting objective function is now as follows (Schulman et al., 2017):

$$\mathcal{L}^{\theta_k}_{\text{CLIP}}(\theta) = E_{\tau\sim\pi_k}\left[\sum_{t=0}^{T}\min\left(r_t(\theta)\hat{A}_t^{\pi_k}, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^{\pi_k}\right)\right] \quad (6.39) \tag{4.8}$$

If the probability ratio between the new and old policies exceeds the range $(1-\epsilon)$ to $(1+\epsilon)$, adjustments are made to the advantage function. Clipping is applied to limit the extent of effective modifications permitted at each stage, thereby enhancing stability. This discourages significant alterations to the policy if they fall beyond our acceptable threshold (Schulman et al., 2017). The method is illustrated in Equation 1. We can observe the model's architecture in Figure 4.2(Schulman et al., 2017)

Algorithm 1: PPO with Clipped Objective from Schulman et al. (2017)

Input: initial policy parameters θ_0, clipping threshold

for $k = 0, 1, 2, ...$ **do**

Collect set of partial trajectories D_k on policy $\pi_k = \pi(\theta_k)$

Estimate advantages $\hat{A}_t^{\pi_k}$

Compute policy update $\theta_{k+1} = \arg\max_\theta \mathcal{L}_{\text{CLIP}}^{\theta_k}(\theta)$ by taking K steps of minibatch SGD (via Adam), where

$\mathcal{L}_{\text{CLIP}}^{\theta_k}(\theta) = E_{\tau \sim \pi_k}\left[\sum_{t=0}^{T} \min\left(r_t(\theta)\hat{A}_t^{\pi_k}, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^{\pi_k}\right)\right]$

end

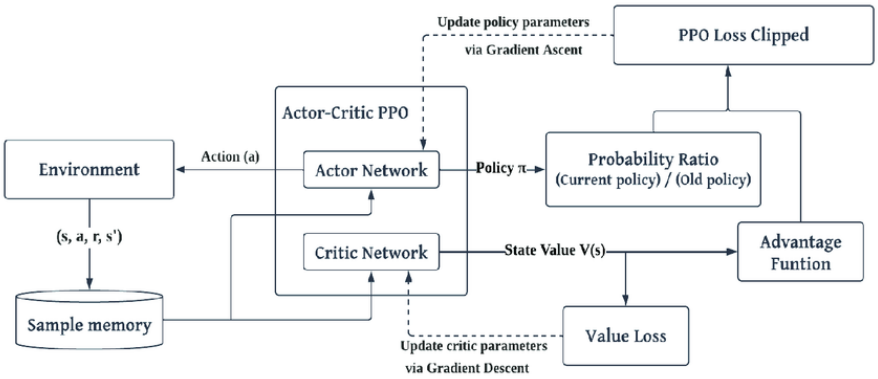

Figure 4.2: PPO model architecture

4.4 Technical Indicators in Trading

Traders often rely on indicators such as the moving average, relative strength index, moving average convergence divergence, and stochastic oscillators to identify buy and sell signals. However, making profits in a market where numerous investors trade against each other presents a challenge. According to the efficient market hypothesis, any advantages gained by an investor are susceptible to being nullified by others who possess the same market information (C. I. Lee, Pan, & Liu, 2001). Consequently, investors seek additional insights to inform their trading decisions, often turning to historical data for guidance on future price movements. Trading strategies rooted in technical analysis have garnered significant attention from investors, particularly high-frequency traders(Fyfe, Marney, & Tarbert, 1999). These strategies typically involve parameters

$$\text{EMA}_{n1}, \text{EMA}_{n2}, \text{EMA}_{n3} \tag{4.9}$$

$$\text{MACD} = \text{EMA}_{n1} - \text{EMA}_{n2} \tag{4.10}$$

$$\text{MACD Signal} = \text{EMA}_{n3} \text{ of MACD} \tag{4.11}$$

Figure 4.3: MACD Formula and its Signal Line

borrowed from conventional trading practices, such as selecting durations for the moving average and relative strength index. Recent research has focused on refining and optimizing these trading strategies (Faijareon & Sornil, 2019).

4.4.1 Turbulence Index

The Turbulence indicator functions as a tool for evaluating risk. (Chow, Jacquier, Kritzman, & Lowry, 1999) were the first to introduce a measure of financial turbulence, originally developed by (Mahalanobis, 1925). Their methodology addresses the volatility of risk parameters within the realm of portfolio allocation. A central part of their study involves detecting multivariate outliers and employing them to calculate a revised covariance matrix. The authors argue that their approach provides a more precise assessment of a portfolio's risk level during periods of market turbulence. However, their method is designed specifically for portfolio allocation and does not prioritize the identification of market turbulence periods. (Dumitrescu, 2015)

4.4.2 MACD

Gerald Appel developed the Moving Average Convergent Divergent (MACD) in the late 1970s. It is employed to monitor stock direction and assess stock price momentum through the utilization of two moving averages:

A sample trading rule based on MACD is: buy if MACD is greater than signal. Parameters of the MACD trading rules are fast length $n1$, slow length $n2$, and signal length $n3$. (Faijareon & Sornil, 2019)

4.4.3 RSI

The relative strength index (RSI) is a tool for monitoring stock direction and quantifying the rate of price change over a specified timeframe. It ranges from 0 to 100 and can be

computed as follows:

$$\text{RSI} = 100 - \frac{100}{1 + \text{RS}} \tag{4.12}$$

where RS is the average gain of up periods during a time frame divided by average loss of down periods during the time frame.(Faijareon & Sornil, 2019)

4.4.4 Sharpe Ratio

The Sharpe ratio, widely recognized in the industry for measuring risk-adjusted returns, will be used to evaluate performance. The numerator of the ratio represents the expected portfolio return minus the risk-free rate, while the denominator is the expected volatility of the portfolio or the standard deviation of returns (adjusted by subtracting the risk-free asset's standard deviation, which is zero) (*deborahkidd.com*, n.d.). The resulting ratio isolates the expected excess return that the portfolio could generate per unit of portfolio return variability. Originally, Sharpe's version assumed that borrowing at the risk-free rate would fund the investment in the risky asset, implying a zero-investment strategy. Over time, the metric has been referred to by users as the Sharpe measure or Sharpe ratio, and it has been used to evaluate investment decisions ex-post. The ex-post, or historical, Sharpe ratio employs actual rather than expected returns and is calculated as follows:

$$\text{Sharpe Ratio} = \frac{R_p - R_f}{\sigma_p} \tag{4.13}$$

In the Sharpe ratio calculation, R_p is the average return of the portfolio, R_f is the average return of the risk-free rate for the period being assessed, and σ_p is the average standard deviation of the portfolio. An investor is informed by the Sharpe ratio about the proportion of a portfolio's performance linked with undertaking risk. It quantifies the incremental value of a portfolio compared to its overall risk. A portfolio comprising risk-free assets or one yielding zero excess return would register a Sharpe ratio of zero (*deborahkidd.com*, n.d.).

4.5 Entropy Coefficient Value

The entropy coefficient functions as a regularizer. When all actions have equal probabilities, the policy achieves maximum entropy, whereas it minimizes when one action's probability becomes dominant. This coefficient is multiplied by the maximum possible

entropy and integrated into the loss function. Its role is to prevent premature dominance of one action's probability within the policy, thereby encouraging exploration (AurelianTactics, 2018). Serving as a hyperparameter in the PPO model, it steers exploration. Adjustments to this coefficient are aimed at broadening the model's exploration capacity, particularly in tackling the CAT problem, which involves a wide array of potential moves. Insufficient exploration could impede the model's ability to learn optimal strategies (Corecco, Adorni, & Gambardella, 2023)

This research demonstrates the benefits of adjusting the entropy coefficient in enhancing performance. It indicates that integrating entropy into the reward system, along with adopting a more stochastic policy, alters the optimization goal. Additionally, it suggests that entropy and stochasticity tend to promote a more favorable objective for optimization. Through a secondary experimental setup, the study explores the optimization effects of stochastic policies, revealing that high entropy policies can accelerate learning and improve the quality of final solutions. (Ahmed et al., 2019) . This means that the agent would explore states as it is learning and that the model would also be better equipped to deal with abnormal situations, which would make it more robust.(Olsson et al., 2022)

This paper also proves how a change in the entropy coefficient can greatly affect performance (Eimer et al., 2023) OpenAI has published an article that discusses varying performance levels associated with different entropy coefficients. In a DRL context, they demonstrate that lower entropy tends to result in poorer performance because the model struggles more with exploration, while higher entropy leads to even worse outcomes due to excessively random actions.(Berner et al., 2019). Through these articles, we can observe that different entropy coefficient values can drive performance, thus making us comfortable that our experiment will yield some positive and significant results for this research field.

Chapter 5

Methodology and results analysis

This section is divided into a methodology part and a results part. In the methology part, I will detail how the experiment was executed sequentially, detailing the acquisition and data pre-processing, the descriptive data analysis, the environment design, the application of the DRL algorithms and the evaluation of results.Afterwards, in the experiment evaluation part, we will evaluate the shift in the model's performance when changing the entropy coefficient. If when running experiments the sharpe ratio improves, we will conclude that said entropy coefficient is the most efficient for this context.

5.1 Methodology

The primary aim of this study is create a model that can improve stock price forecasting by adjusting the entropy coefficient. Specifically, this involves altering the entropy coefficient values within the PPO DRL model.

To accomplish this objective, the FiNRL library will be employed. As the initial open-source framework tailored for financial reinforcement learning, it has three layers: market environments, agents, and applications. In the context of trading, this means making sequential decisions as an agent interacts with a market environment. Additionally, several open-source Python packages will be integrated, including as Yahoo Finance API, pandas, numpy, matplotlib, stockstats, OpenAI gym, stable-baselines, TensorFlow, and pyfolio. (FinRL, 2023a)

This thesis will examine the trading data of Dow 30 constituents from 2008 to 2023, employing the final year for testing and the rest of the years for training purposes. The

Dow Hones is composed by 30 notable U.S. publicly traded companies.(*What Is the Dow 30, Companies In It, Significance — investopedia.com*, 2023) Serving as an indicator for the U.S. stock market and economy, it reflects the combined share price performance of significant entities listed on the New York Stock Exchange (NYSE) and NASDAQ, excluding transportation and utility firms.

- **Acquisition and data pre-processing:** We will acquire data from Yahoo Finance and we will apply several pre-processing techniques to better apply our model.
- **Descriptive data analysis:** Analysis of data was carried out to understand the structure of our dataset. We examined distribution, seasonality, and evolution to gain an initial understanding of the dataset.
- **Environment Design** Financial tasks are modeled as a Markov Decision Process (MDP) challenge. The training process involves monitoring stock price fluctuations, executing actions, and assessing rewards to facilitate the agent's strategy adjustments. Via reinforcement learning, the trading agent will formulate a strategy geared towards maximizing rewards over time. Our trading environments, built upon the OpenAI Gym framework, emulates real stock markets by employing time-driven simulation principles with actual market data.(FinRL, 2023b)
- **Application of DRL Algorithms** We will experiment with various entropy coefficient values for PPO. We have an initial capital of $1,000,000 on January 1st, 2023 and we will trade Dow Jones 30 stocks with that capital. (FinRL, 2023b)

We will now examine different entropy coefficients to determine whether altering this parameter significantly impacts model accuracy. Our focus will be on the Sharpe ratio, a key metric, calculated over a fixed trading period of one year following training on previous years' data from the DOW JONES 30 index ETF.(FinRL, 2023b)

Actions define defining stock weight allocations, rewards quantify changes in portfolio value, and states take in observed features. The environment consists of the Dow 30 stocks.(FinRL, 2023b)

5.2 Data pre-processing

Yahoo Finance provides us with the stock informationat no cost. FinRL utilizes a YahooDownloader class to download data from the Yahoo Finance API, which has a call

limit of 2,000 requests per hour per IP or up to 48,000 requests per day. Subsequently, this data is stored in a pandas library DataFrame. This data is used in the Python file for further data pre-processing. Technical indicators are incorporated in the dataset from the start. In real-world trading, various factors have to be taken in, such as historical stock prices, current holdings, and technical indicators. In this context, we illustrate two trend-following technical indicators: MACD and RSI. Additionally, a turbulence index is introduced.(FinRL, 2023b)

Risk aversion plays a crucial role in determining whether an investor will prioritize capital preservation. It also influences the trader's trading strategy in response to market volatility. To manage risk in highly volatile markets, such as during the Great Financial Crisis, FinRL employs a financial turbulence index for assessment. We manually include the covariance matrix as states.(FinRL, 2023b)

5.3 Descriptive data analysis

Here we can observe a table containing key statistics of the daily changes and a histogram. From both representations, it is evident that average returns are predominantly clustered around zero, which makes sense taking into account classic financial theory that says that financial returns are zero in the long-term. Additionally, the distribution appears to be symmetrical, as the absolute maximum and minimum returns in the distribution are almost equal.

Statistic	**Value**
Count	4027.000000
Mean	0.000339
Standard Deviation	0.012231
Minimum	-0.129265
25% Quartile	-0.004229
50% Quartile	0.000554
75% Quartile	0.005597
Maximum	0.113650

Table 5.1: Summary Statistics

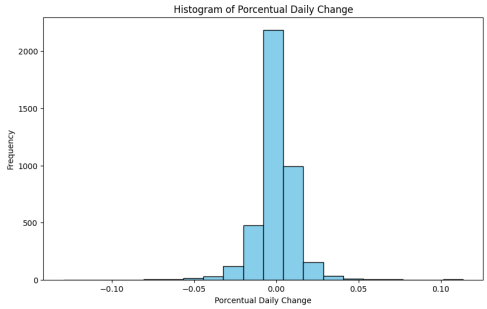

Table 5.2: Histogram

Through this seasonality decomposition, we can see the big seasonality that stock's prices suffer. We can also observe that markets tend to move around the economic cycles

which makes them have this type of coherent trend.

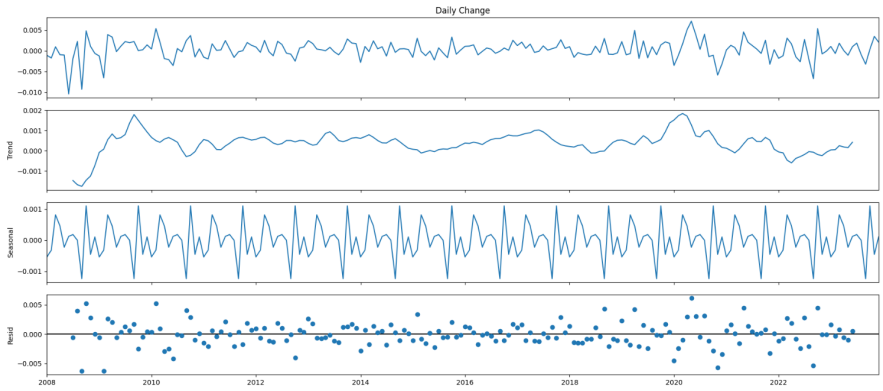

Figure 5.1: LOESS Decomposition

We are as well able to observe the decomposition of returns through this scatterplot in 5.2, where we can observe how stock returns moved over the observed period. As an example of volatility, we can clearly observe the covid crisis as one of the most volatile times of the last 15 years for the stock market.

One of the primary assumptions in classical portfolio management models is that returns follow a normal distribution. We will now examine whether this assumption holds true for our dataset. To assess this, a statistical Shapiro test was conducted to determine the normality of the distribution. The results indicate that we cannot confirm a normal distribution with a 90% confidence level. In order to get further confirmation,a plot was generated to compare it with the distribution of points if the model were entirely normal. From the visual representation in 5.3, it becomes apparent that our distribution is not normal, as it is more concentrated around the mean compared to a typical normal distribution.

5.4 Environment Design

5.4.1 Train-Test Split

In time series forecasting, randomly dividing the dataset into training and testing sets does not make sense since observations are dependent in time series analysis. For this project, the period between 2008 and 2022 was selected as training and the test was 2023. This resulted in a training test split of 95/5.(FinRL, 2023b)

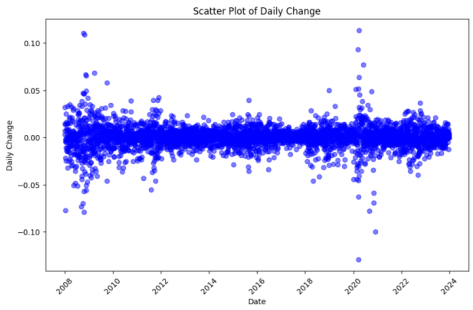

Figure 5.2: Scatterplot of the Dow Jones returns from 2008 to 2023

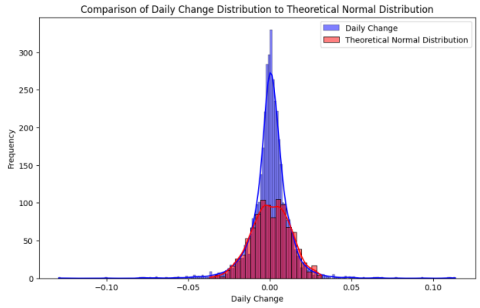

Figure 5.3: Comparison between how the dataset would be shaped if it had a perfect Normal Distribution and how it's shaped

5.4.2 Stock trading environemnt

This framework uses a DataFrame (`df`) to represent input data and integrates crucial parameters such as stock_dim, indicating the number of unique stocks; `hmax`, setting the maximum number of shares for trading; initial_amount, meaning the starting capital; transaction_cost_pct, representing the percentage of transaction costs per trade. Furthermore, it also uses reward_scaling to fine-tune reward magnitudes, state_space to establish input feature dimensions, and action_space corresponding to the dimensionality of stocks. Additionally, tech_indicator_list contains the names of technical indicators guiding trading decisions, while turbulence_threshold regulates risk aversion. Lastly, `day` functions for the dates. (FinRL, 2023b)

In terms of actions undertaken during training, the model has various actions. These actions include `sell_stock()` and `buy_stock()`, for selling and buying actions respectively based on the sign of the action. The `step()` method organizes the trading process by returning actions, computing rewards, and providing subsequent observations at each step. Moreover, the `reset()` function is utilized to reset the environment, while `render()` returns additional functions for visualization . Additionally, `save_asset_memory()` and `save_action_memory()` are employed to document values and actions/positions at each time step, facilitating comprehensive data collection and analysis within the trading environment. (FinRL, 2023b)

5.5 DRL Algorithm Implementation

We'll evaluate different entropy coefficient values for PPO. Starting with an initial investment of $1,000,000 on January 1, 2023, we'll use trained PPO models to trade Dow Jones 30 stocks. To account for transaction costs, we set the transaction cost at 0.1%. We will use the Sharpe Ratio to examine each entropy coefficient's performance in this dataset.(FinRL, 2023b)

5.5.1 PPO Model Hyperparameters

- **Horizon range** Horizon refers to the number of time steps into the future that a current reward can be linked to a past action (or how far back a past action can influence the current reward). For consistency with DRL models, we will consistently employ a horizon of 2048 steps, a widely accepted practice in the field. (Unity, 2020)

- **Entropy coefficient** The entropy coefficient acts as a regularizer in reinforcement learning. It measures the level of uncertainty in the policy, with maximum entropy occurring when all actions are equally probable and minimum entropy when one action dominates. During training, the entropy coefficient is multiplied by the maximum possible entropy and added to the loss function. We will experiment with various entropy coefficients, including 0.0, 0.00001, 0.0001, 0.001, 0.01, 0.1, 0.5, and 0.9 to determine the most effective option. (AurelianTactics, 2018)

- **Learning Rate** Deep learning neural networks are trained through the stochastic gradient descent algorithm. This optimization technique estimates the error gradient for the current model state using examples from the training dataset. It then updates the model's weights using the backpropagation of errors algorithm. The magnitude of these weight updates during training is referred to as the step size or the "learning rate." The learning rate is a tunable hyperparameter crucial in neural network training, typically set to a small positive value between 0.0 and 1.0. (Brownlee, 2019) In this study, we will adopt a stable learning rate of 0.0001.

- **Batch size** This hyperparameter determines the number of samples processed before updating the internal model parameters. (Devansh, 2024) We will use a selected batch size of 128.

5.6 Evaluation of results

In this academic study, we aim to estimate the mean of the Sharpe ratio through a repeated experiment conducted 25 times with the different entropy coefficients, employing different random seeds. This experiment will experiment with both the default entropy coefficient and higher entropy coefficients. We think that higher values of the entropy coefficient will lead to more robust policies for the agent, attributed to a more exploratory learning process of the policy distribution. However, we also anticipate a potential degradation in performance with higher entropy coefficients.

Subsequently, we will conduct a statistical hypothesis contrast of pairs of means and a linear regression analysis of the Sharpe ratio. This analysis will be based on the outcomes derived from various values of the entropy regularizer. The objective is to assess what the relationship between the Sharpe ratio and the entropy regularizer is.(FinRL, 2023b)

In Table 5.3 presented below, we present a comparative analysis of performance across various entropy coefficients utilized, with consideration of five distinct seeds. Each seed and entropy coefficient pairing showcases the model's performance, quantified by its Sharpe ratio. Notably, there is consistent performance observed across the array of entropy coefficients. Among our experiments, the lowest recorded Sharpe ratio is 1.35368135 for Seed 2 and a coefficient of 0.0, while the highest Sharpe ratio is 1.35566438 for Seed 4 utilizing a coefficient of 0.00001. The difference between the best and worst performing experiments amounts to a just 0.15% change in model performance. Through entropy coefficient values ranging from 0 to 1, for this specific dataset, alterations in the entropy coefficient hyper-parameter, while holding all other variables constant, impact performance very little.

In order to further illustrate the lack of performance variance between the different entropy coefficients, we took the average performance of a entropy coefficient through its different seeds (a contrast of pairs of means), which is represented in 5.4. In this graph, we once again observe almost no change in performance between the different entropy coefficients, thus proving no change in performance. As noted in the methodology part of this thesis, it needs to be stated that each experiment was run 25 times, thus proving that the change in entropy coefficient is not significant in this context and that the lack of change in performance is not just due to chance.

It's important to note that this experiment was conducted with very limited resources,

including very limited computational capacity as well as very limited time to conduct this experiment. In order to get more assurance over the results, these experiments would need to be conducted with more computational capacity.

Entropy Coefficient	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5
0.0	1.35368433	1.35368135	1.35369012	1.35368806	1.35369305
0.00001	1.35565479	1.35565737	1.3556624	1.35566438	1.35566236
0.0001	1.35565479	1.35565737	1.35566241	1.35566438	1.35566235
0.001	1.35565479	1.35565737	1.35566241	1.35566438	1.35566236
0.01	1.35565479	1.35565737	1.3556624	1.35566438	1.35566235
0.1	1.3556548	1.35565738	1.35566237	1.35566437	1.35566235
0.5	1.35565479	1.35565738	1.35566235	1.35566428	1.35566232
0.9	1.35565477	1.3556574	1.35566235	1.35566426	1.35566234

Table 5.3: Sharpe ratios of the different entropy coefficients and seeds.

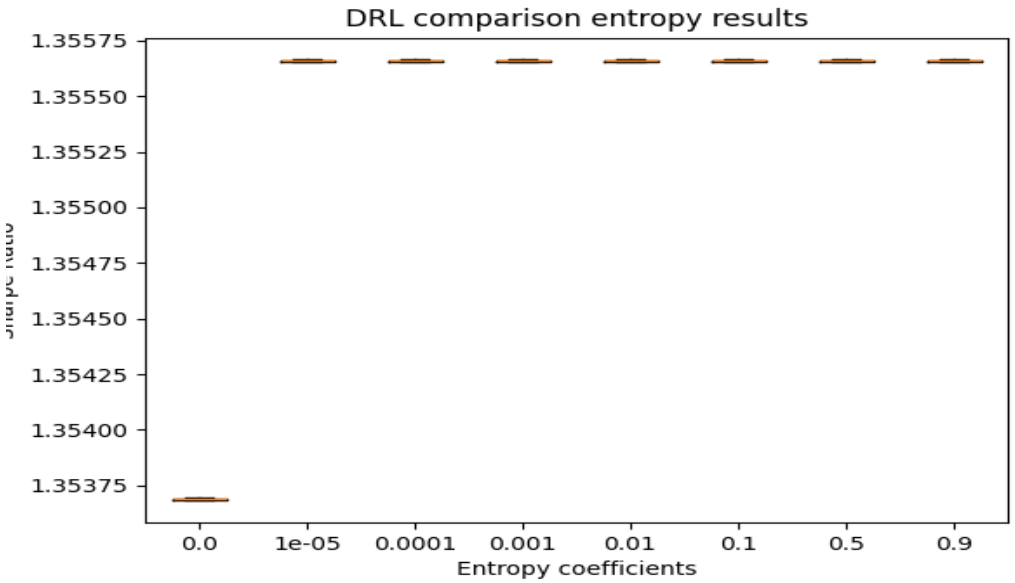

Figure 5.4: Contrasts of pairs of means between the different entropy coefficients showing no significant difference in performance between the different entropy coefficients

Chapter 6

Conclusions

Our research project analyses the importance of fine-tuning the entropy coefficient hyperparameter in Portfolio Management using Deep Reinforcement models. We examined the utility of incorporating deep learning methodologies in portfolio management by analyzing a real-world dataset from the Dow Jones 30 index. Our study tries to determine whether adjustments to the entropy coefficient would have a substantial impact on the model's effectiveness, as assessed through the Sharpe ratio. Our results indicate that altering the entropy coefficient does not significantly influence the model's performance.

We start the paper by analyzing the development of quantitative finance in portfolio management and highlighting the potential of Deep Reinforcement Learning (DRL) as a cutting-edge technology in this sphere. Through a comprehensive literature review, we identified crucial methodologies and recent advancements in DRL Portfolio Management. This helped us select Proximal Policy Optimization (PPO) model for experimentation. Additionally, we explored various aspects of DRL models, including their applications, model selections, and available resources.

After understanding DRL Portfolio Management research, we formulated hypotheses and research objectives. Subsequently, we highlight the foundational technology behind DRL models, emphasizing the transformative influence of deep learning and reinforcement learning on financial analysis. We provided a deep explanation behind the theoretical background of the PPO model, along with the technical indicators integrated into our analysis.

In our methodology, we implemented the PPO model and applied descriptive data

analysis to understand the dataset. Through multiple experiment with various entropy coefficients, we observed no significant deviations in performance, underscoring the model's consistency across different scenarios.

Looking forward, we recommend exploring other hyperparameters such as learning rate and batch size . Furthermore, we encourage other scholars to research other DRL models for Portfolio Management, because of their proven efficacy.

Declaración de Uso de Herramientas de Inteligencia Artificial Generativa en Trabajos Fin de Grado

ADVERTENCIA: Desde la Universidad consideramos que ChatGPT u otras herramientas similares son herramientas muy útiles en la vida académica, aunque su uso queda siempre bajo la responsabilidad del alumno, puesto que las respuestas que proporciona pueden no ser veraces. En este sentido, NO está permitido su uso en la elaboración del Trabajo fin de Grado para generar código porque estas herramientas no son fiables en esa tarea. Aunque el código funcione, no hay garantías de que metodológicamente sea correcto, y es altamente probable que no lo sea.

Por la presente, yo, [Nombre completo del estudiante], estudiante de [nombre del título] de la Universidad Pontificia Comillas al presentar mi Trabajo Fin de Grado titulado "[Título del trabajo]", declaro que he utilizado la herramienta de Inteligencia Artificial Generativa ChatGPT u otras similares de IAG de código sólo en el contexto de las actividades descritas a continuación [el alumno debe mantener solo aquellas en las que se ha usado ChatGPT o similares y borrar el resto. Si no se ha usado ninguna, borrar todas y escribir “no he usado ninguna”]:

1. **Brainstorming de ideas de investigación:** Utilizado para idear y esbozar posibles áreas de investigación.
2. **Crítico:** Para encontrar contra-argumentos a una tesis específica que pretendo defender.
3. **Referencias:** Usado conjuntamente con otras herramientas, como Science, para identificar referencias preliminares que luego he contrastado y validado.
4. **Metodólogo:** Para descubrir métodos aplicables a problemas específicos de investigación.
5. **Interpretador de código:** Para realizar análisis de datos preliminares.
6. **Estudios multidisciplinares:** Para comprender perspectivas de otras comunidades sobre temas de naturaleza multidisciplinar.
7. **Constructor de plantillas:** Para diseñar formatos específicos para secciones del trabajo.
8. **Corrector de estilo literario y de lenguaje:** Para mejorar la calidad lingüística y estilística del texto.
9. **Generador previo de diagramas de flujo y contenido:** Para esbozar diagramas iniciales.
10. **Sintetizador y divulgador de libros complicados:** Para resumir y comprender literatura compleja.
11. **Generador de datos sintéticos de prueba:** Para la creación de conjuntos de datos ficticios.
12. **Generador de problemas de ejemplo:** Para ilustrar conceptos y técnicas.
13. **Revisor:** Para recibir sugerencias sobre cómo mejorar y perfeccionar el trabajo con diferentes niveles de exigencia.
14. **Generador de encuestas:** Para diseñar cuestionarios preliminares.
15. **Traductor:** Para traducir textos de un lenguaje a otro.

Afirmo que toda la información y contenido presentados en este trabajo son producto de mi investigación y esfuerzo individual, excepto donde se ha indicado lo contrario y se han dado los créditos correspondientes (he incluido las referencias adecuadas en el TFG y he explicitado para que se ha usado ChatGPT u otras herramientas similares). Soy consciente de las implicaciones académicas y éticas de presentar un trabajo no original y acepto las consecuencias de cualquier violación a esta declaración.

Fecha: 23/04/2024

Firma: Roberto Gozalo Brizuela

References

Ahmed, Z., Le Roux, N., Norouzi, M., & Schuurmans, D. (2019). Understanding the impact of entropy on policy optimization. In *International conference on machine learning* (pp. 151–160).

AurelianTactics. (2018). *PPO Hyperparameters and Ranges*. `https://medium.com/aureliantactics/ppo-hyperparameters-and-ranges-6fc2d29bccbe`. ([Accessed 07-04-2024])

Bartram, S. M. (2020). *Artificial intelligence in asset management* (WBS Finance Group Research Paper). Retrieved from `http://hdl.handle.net/11159/421820` (10.2139/ssrn.3510343; 10.2139/ssrn.3510343)

Benhamou, E., Saltiel, D., Ungari, S., & Mukhopadhyay, A. (2020). *Time your hedge with deep reinforcement learning.*

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., … others (2019). Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Brownlee, J. (2019). *How to configure the learning rate.* `https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/`. ([Accessed 07-04-2024])

Charpentier, A., Elie, R., & Remlinger, C. (2021). Reinforcement learning in economics and finance. *Computational Economics*, 1–38.

Chow, G., Jacquier, E., Kritzman, M., & Lowry, K. (1999). Optimal portfolios in good times and bad. *Financial Analysts Journal*, *55*(3), 65–73.

Cong, L. W., Tang, K., Wang, J., & Zhang, Y. (2021). Alphaportfolio: Direct construction through deep reinforcement learning and interpretable ai. *Available at SSRN 3554486*.

Corecco, S., Adorni, G., & Gambardella, L. M. (2023). Proximal policy optimization-based reinforcement learning and hybrid approaches to explore the cross array task

optimal solution. *Machine learning and knowledge extraction*, *5*(4), 1660–1679.

deborahkidd.com. (n.d.). https://deborahkidd.com/wp-content/uploads/The-Sharpe-Ratio-and-the-Information-Ratio-1.pdf. ([Accessed 03-04-2024])

de Inversión, E. (2023). *Modelo de markowitz.* https://www.estrategiasdeinversion.com/herramientas/diccionario/mercados/modelo-de-markowitz-t-240. ([Accessed 13-November-2023])

Dempster, M. A., & Leemans, V. (2006). An automated fx trading system using adaptive reinforcement learning. *Expert systems with applications*, *30*(3), 543–552.

Dempster, M. A., Payne, T. W., Romahi, Y., & Thompson, G. W. (2001). Computational learning techniques for intraday fx trading using popular technical indicators. *IEEE Transactions on neural networks*, *12*(4), 744–754.

Deng, Y., Kong, Y., Bao, F., & Dai, Q. (2015). Sparse coding-inspired optimal trading system for hft industry. *IEEE Transactions on Industrial Informatics*, *11*(2), 467–475.

Devansh. (2024). *How does batch size impact your model learning.* https://medium.com/geekculture/how-does-batch-size-impact-your-model-learning-2dd34d9fb1fa. ([Accessed 07-04-2024])

Dumitrescu, S. (2015). Turbulence and systemic risk in the european union financial system. *Financial Studies*, *19*(2A), 41–71.

Eilers, D., Dunis, C. L., von Mettenheim, H.-J., & Breitner, M. H. (2014). Intelligent trading of seasonal effects: A decision support algorithm based on reinforcement learning. *Decision support systems*, *64*, 100–108.

Eimer, T., Lindauer, M., & Raileanu, R. (2023). Hyperparameters in reinforcement learning and how to tune them. In *International conference on machine learning* (pp. 9104–9149).

Faijareon, C., & Sornil, O. (2019). Evolving and combining technical indicators to generate trading strategies. In *Journal of physics: Conference series* (Vol. 1195, p. 012010).

FinRL. (2023a). *Ai for finance.* https://github.com/AI4Finance-Foundation/FinRL. ([Accessed 08-04-2024])

FinRL. (2023b). *Deep reinforcement learning for stock trading from scratch: Multiple stock trading.* https://github.com/AI4Finance-Foundation/FinRL/blob/master/examples/Stock_NeurIPS2018_SB3.ipynb. ([Accessed 23-

04-2024])

Fischer, M. M., et al. (2015). Neural networks. a class of flexible non-linear models for regression and classification. *Handbook of Research Methods and Applications in Economic Geography*, 172–192.

Fischer, T. G. (2018). *Reinforcement learning in financial markets-a survey* (Tech. Rep.). FAU discussion papers in economics.

François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., Pineau, J., et al. (2018). An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, *11*(3-4), 219–354.

Fyfe, C., Marney, J. P., & Tarbert, H. F. (1999). Technical analysis versus market efficiency-a genetic programming approach. *Applied Financial Economics*, *9*(2), 183–191.

Gu, F., Jiang, Z., & Su, J. (2021). Application of features and neural network to enhance the performance of deep reinforcement learning in portfolio management. In *2021 ieee 6th international conference on big data analytics (icbda)* (pp. 92–97).

Guan, M., & Liu, X.-Y. (2021). *Explainable deep reinforcement learning for portfolio management: An empirical approach.*

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., … others (2018). Deep q-learning from demonstrations. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 32).

Insights, G. (2023). *Portfolio management market industry analysis.* `https://www.gminsights.com/industry-analysis/project-portfolio-management-market`. ([Accessed 23-October-2023])

Jang, J., & Seong, N. (2023). Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory. *Expert Systems with Applications*, *218*, 119556.

Jiang, Z., Xu, D., & Liang, J. (2017). *A deep reinforcement learning framework for the financial portfolio management problem.*

Kapoor, N. (2014). Financial portfolio management: Overview and decision making in investment process. *International Journal of Research (IJR)*, *1*(10), 1362–1369.

Koratamaddi, P., Wadhwani, K., Gupta, M., & Sanjeevi, S. G. (2021). Market sentiment-aware deep reinforcement learning approach for stock portfolio allocation. *Engineering Science and Technology, an International Journal*, *24*(4), 848-859. Retrieved from `https://www.sciencedirect.com/science/article/pii/`

S2215098621000070 doi: https://doi.org/10.1016/j.jestch.2021.01.007

Koubaa, A., Qureshi, B., Ammar, A., Khan, Z., Boulila, W., & Ghouti, L. (2023). Humans are still better than chatgpt: Case of the ieeextreme competition. *Heliyon*, *9*(11).

Latex Figure for Deep Reinforcement Learning — tex.stackexchange.com. (n.d.). https://tex.stackexchange.com/questions/523603/latex-figure-for-deep-reinforcement-learning. ([Accessed 14-04-2024])

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

Lee, C. I., Pan, M.-S., & Liu, Y. A. (2001). On market efficiency of asian foreign exchange rates: Evidence from a joint variance ratio test and technical trading rules. *Journal of International Financial Markets, Institutions and Money*, *11*(2), 199–214.

Lee, J. W., Park, J., O, J., Lee, J., & Hong, E. (2007). A multiagent approach to q-learning for daily stock trading. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, *37*(6), 864-877. doi: 10.1109/TSMCA.2007.904825

Li, H., Dagli, C. H., & Enke, D. (2007). Short-term stock market timing prediction under reinforcement learning schemes. In *2007 ieee international symposium on approximate dynamic programming and reinforcement learning* (pp. 233–240).

Liang, Z., Chen, H., Zhu, J., Jiang, K., & Li, Y. (2018). *Adversarial deep reinforcement learning in portfolio management.*

Lim, Q. Y. E., Cao, Q., & Quek, C. (2022). Dynamic portfolio rebalancing through reinforcement learning. *Neural Computing and Applications*, *34*(9), 7125–7139.

Lin, Y.-C., Chen, C.-T., Sang, C.-Y., & Huang, S.-H. (2022). Multiagent-based deep reinforcement learning for risk-shifting portfolio management. *Applied Soft Computing*, *123*, 108894.

Ling, D. (2019). Measuring model stability. *SAS Resource Papers*(3568).

Mahalanobis, P. C. (1925). Analysis of race-mixture in bengal.

Moody, J., & Wu, L. (1997). Optimization of trading systems and portfolios. In *Proceedings of the ieee/iafe 1997 computational intelligence for financial engineering (cifer)* (p. 300-307). doi: 10.1109/CIFER.1997.618952

Moody, J., Wu, L., Liao, Y., & Saffell, M. (1998). Performance functions and reinforce-

ment learning for trading systems and portfolios. *Journal of forecasting*, *17*(5-6), 441–470.

Neuneier, R. (1995). Optimal asset allocation using adaptive dynamic programming. *Advances in neural information processing systems*, *8*.

News, U. (2023). *7 top investment firms using ai for asset management.* `https://money.usnews.com/investing/articles/7-top-investment-firms-using-ai-for-asset-management`. ([Accessed 23-October-2023])

Olsson, M., Malm, S., & Witt, K. (2022). *Evaluating the effects of hyperparameter optimization in vizdoom.*

Ouyang, Z. (2022). A study of stock portfolio strategy based on machine learning. In *2022 7th international conference on financial innovation and economic development (icfied 2022)* (pp. 79–87).

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Soleymani, F., & Paquet, E. (2021). Deep graph convolutional reinforcement learning for financial portfolio management – deeppocket. *Expert Systems with Applications*, *182*, 115127. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0957417421005686` doi: https://doi.org/10.1016/j.eswa.2021.115127

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.

SUN, Z., & AN, Y. (2020). *Portfolio management with reinforcement learning.* Retrieved from `https://openreview.net/forum?id=YdJuGLgMo4H`

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

Unity. (2020). *Time horizon explanation.* `https://forum.unity.com/threads/in-depth-explanation-for-time-horizon-hyperparameter.818169/`. ([Accessed 08-04-2024])

Wang, Z., Huang, B., Tu, S., Zhang, K., & Xu, L. (2021). Deeptrader: a deep reinforcement learning approach for risk-return balanced portfolio management with market conditions embedding. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 35, pp. 643–650).

What Is the Dow 30, Companies In It, Significance — investopedia.com. (2023).

https://www.investopedia.com/terms/d/dow-30.asp. ([Accessed 07-04-2024])

Yang, H., Liu, X.-Y., Zhong, S., & Walid, A. (2020). Deep reinforcement learning for automated stock trading: An ensemble strategy. In *Proceedings of the first acm international conference on ai in finance* (pp. 1–8).

Yang, S. (2023). Deep reinforcement learning for portfolio management. *Knowledge-Based Systems*, *278*, 110905. Retrieved from https://www.sciencedirect.com/science/article/pii/S095070512300655X doi: https://doi.org/10.1016/j.knosys.2023.110905

Yu, P., Lee, J. S., Kulyatin, I., Shi, Z., & Dasgupta, S. (2019). *Model-based deep reinforcement learning for dynamic portfolio optimization.*

Zhang, Y., Zhao, P., Wu, Q., Li, B., Huang, J., & Tan, M. (2020). Cost-sensitive portfolio selection via deep reinforcement learning. *IEEE Transactions on Knowledge and Data Engineering*, *34*(1), 236–248.

Zhao, T., Ma, X., Li, X., & Zhang, C. (2023). Asset correlation based deep reinforcement learning for the portfolio selection. *Expert Systems with Applications*, *221*, 119707. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417423002087 doi: https://doi.org/10.1016/j.eswa.2023.119707