Smart Quantitative Trading Strategies using Reinforcement Learning Framework

Submitted April 18, 2024, in partial fulfillment of
the conditions for the award of the degree **BSc (Hons) Computer Science with Artificial Intelligence.**

Zhongkai Yang
20318227
scyzy3@nottingham.edu.cn

Supervised by Dr. Tianxiang Cui

School of Computer Science, University of Nottingham Ningbo China

Acknowledgements

I would like to express profound gratitude to my supervisor, Dr. Tianxiang Cui, for his continuous support and expert guidance throughout my final year studies. I am also thankful to his PhD student Mr Nanjiang Du, who have generously shared his expertise and provided practical assistance when needed. Their contributions have significantly enriched my learning experience.

Additionally, I owe a special thanks to my parents for their unwavering financial support and constant encouragement. Their commitment has not only enabled me to complete my undergraduate degree but also to pursue my graduate studies. Their belief in my potential continues to be a powerful source of motivation throughout my academic journey.

I also extend my gratitude to all those around me who have offered their support through this journey. This journey would not have been possible without the support and patience of everyone mentioned, and I am deeply appreciative of their involvement.

Abstract

This thesis investigates the application of Deep Reinforcement Learning (DRL) to develop advanced quantitative trading strategies for the cryptocurrency market. The study utilizes DRL to optimize real-time trading decisions, aiming to outperform traditional strategies based on technical indicators. Through comprehensive back-testing, the research demonstrates that DRL effectively enhances profitability and risk management in trading cryptocurrencies like Bitcoin. The findings underscore the potential of DRL to transform financial trading, offering insights for future applications in various financial instruments.

Contents

Acknowledgments **i**

Abstract **ii**

List of Figures **v**

List of Tables **vi**

List of Abbreviations **vii**

1 Introduction **1**
1.1 Background 1
1.2 Motivation 2
1.3 Aims and Objectives 3
1.4 Study Design 4
1.5 Thesis Structure 5

2 Literature Review **6**
2.1 Fundamental Analysis and Technical Analysis 6
2.2 Algorithmic Trading 7
2.3 Reinforcement Learning 8
2.4 Deep Reinforcement Learning 9
2.4.1 Critic-only and Actor-only Deep Reinforcement Learning 9
2.4.2 Actor-Critic Deep Reinforcement Learning 10

3 Methodologies **12**
3.1 Project Structure 12
3.2 Data Processor 13
3.2.1 Technical Indicators 13
3.2.2 Data Splitting 14
3.3 Modeling 15
3.3.1 Neural Network 15
3.3.2 Observation Space 16
3.3.3 Reward Design 18
3.3.4 Overall Modelling 19
3.4 Evaluations 20
3.4.1 MA Strategy 21
3.4.2 MACD Strategy 22
3.4.3 KDJ Strategy 23
3.4.4 Mixed Strategy: MA, MACD, RSI 25
3.5 Continuous Adjustment 26

4 Results and Evaluations **28**
4.1 Dataset Overview 28
4.2 Experimental Settings 30
4.3 BackTest: Agents Validations 32
4.4 BackTest: Real Trading 34

5 Summary and Reflections **41**
5.1 Project Management 41
5.2 Reflections 43
5.2.1 Time Management 44
5.2.2 Risk Management 44
5.2.3 Learning Outcomes 45
5.2.4 Benefits of the Final Year Project 45
5.3 Conclusion and Future Work 46

References **47**

List of Figures

1.1 CRISP-DM Process Diagram 5

2.1 Agent and environment interaction 8

3.1 Project Structure 13
3.2 Data splitting 14
3.3 BTC Trends 14
3.4 Model Structure 15

4.1 Data sample 30
4.2 Descriptive statistics of the BTC/USD trading pair dataset I 30
4.3 Descriptive statistics of the BTC/USD trading pair dataset II 30
4.4 PPO BackTest Validation Trading Points 33
4.5 PPO BackTest Validation Trading Profit Rate 33
4.6 DQN BackTest Validation Trading Points 33
4.7 DQN BackTest Validation Trading Profit Rate 34
4.8 PPO BackTest Trading Points 35
4.9 PPO BackTest Trading Profit Rate 35
4.10 DQN BackTest Trading Points 35
4.11 DQN BackTest Trading Profit Rate 36
4.12 MA Strategy Buy and Sell Points 37
4.13 MACD Strategy Buy and Sell Points 37
4.14 KDJ Strategy Buy and Sell Points 38
4.15 Mixed Strategy Buy and Sell Points 39
4.16 Portfolio Value vs Market Value 40

5.1 Original First semester plan 43
5.2 Original Second semester plan 43
5.3 Actual First semester plan 43
5.4 Actual Second semester plan 43

List of Tables

4.1 Validation Trading Results 34
4.2 Trading Results 36
4.3 Trading Results Comparison 40

List of Abbreviations

A2C	Advantage Actor-Critic
AACHER	Assorted Actor-Critic Deep Reinforcement Learning with Hindsight Experience Replay
ATS	Automated Trading Systems
BTC	Bitcoin
CCI	Commodity Channel Index
CRISP-DM	Cross-Industry Standard Process for Data Mining
DDPG	Deep Deterministic Policy Gradient
DL	Deep Learning
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
EMA	Exponential Moving Average
ETH	Ethereum
LSTM	Long Short-Term Memory
MACD	Moving Average Convergence Divergence
MA	Moving Average
MDP	Markov Decision Process
MoPAC	Model Predictive Actor-Critic
OHLCV	Open, High, Low, Close, Volume
PPO	Proximal Policy Optimization
ROI	Return on Investment
RSI	Relative Strength Index

Chapter 1

Introduction

In the age of globalised economies and advances in digital technology, the generation and collection of financial data has grown exponentially, exceeding the ability of manual assessment [13]. Automated Trading Systems (ATS) (also known as algorithmic trading) offer several advantages over traditional human trading. They circumvent the emotional bias inherent in human decision-making and operate continuously, allowing traders to exploit opportunities outside of standard trading hours [37]. This dissertation explores the application of Deep Reinforcement Learning (DRL) in the development of intelligent quantitative trading strategies designed to navigate complex financial markets.

1.1 Background

The dynamic and nonlinear nature of financial markets has fuelled the quest for advanced trading strategies capable of adapting to market fluctuations and predicting future trends with a high degree of accuracy. Traditional quantitative trading strategies based on mathematical and statistical models often struggle to keep pace with these complexities. However, Deep Reinforcement Learning, which is able to process huge datasets through deep neural networks and adapt to new market conditions through reinforcement learning, offers a promising solution.

As of today, April 4^{th}, 2024, the global cryptocurrency market cap stands at \$2.71 trillion, and the total cryptocurrency trading volume in the last day (April 3^{rd}, 2024) has been reported at \$107.61 billion [20]. As a comparison, the market capitalisation for China's A-share market as of February 2024 is approximately \$10.52 trillion USD [12]. The trading price of Bitcoin is set at about \$66,815.20 per BTC/USD pair, with the market capitalisation now reaching approximately \$1,314.32 billion USD. In the 24 hours preceding this data access, the trading volume for Bitcoin amounted to $40.38 billion USD [7]. Cryptocurrencies, also known as digital currencies, function in an online environment free from centralized control [1]. These digital assets are popular despite their risks and experience substantial volatility in a market that is active 24/7. Recently, there's been an increased focus on creating AI-driven trading bots enhanced by machine learning [26].

1.2 Motivation

The advent of Deep Learning (DL) has created a new age of computational power, with models achieving unprecedented performance in controlled simulated environments. However, translating these successes into complex, unpredictable real-world markets presents significant challenges, largely due to the gap between simulation and reality. A study by Prata et al. (2023) highlights that the performance of DL models degrades significantly when dealing with new data, highlighting concerns about their applicability in real market scenarios [38]. The inherent limitations of DL models, mainly the lack of direct market interaction, further exacerbate their inefficiency in dynamic decision-making scenarios such as quantitative trading, where most of the DL-based research nowadays is only for stock price prediction and not end-to-end trading models.

On the contrary, reinforcement learning (RL), especially deep reinforcement learning (DRL) formed through DL, has shown a strong ability in sequential decision-making scenarios with complex sensory inputs, like Atari games [33]. The cryptocurrency trading environment is a similar continuous decision-making environment full of various signals, this similarity makes DRL a powerful approach for revenue maximisation and risk minimisation in financial markets.

DRL has four main advantages in creating quantitative trading strategies: translating market data directly into actionable trading decisions; optimising profits directly, avoiding the need for financial forecasts; incorporating specific task constraints and technical indicators into the training process; and adapting to different market conditions [39]. Despite these advantages, most of the existing research remains at the conceptual level, lacking practical applications in real-time trading and failing to provide comprehensive comparisons with human traders or between different DRL technologies [4]. This gap in application and evaluation motivates me to focus on the cryptocurrency market, whose unique characteristics, such as high liquidity and availability of real-time data, make it a prime candidate for implementing and testing DRL-based trading strategies in a real-time environment.

Furthermore, the potential for generating significant revenue through algorithmic trading, as well as the booming opportunities in the emerging Bitcoin market, underline the motivation for the project. This research explores the potential of DRL to revolutionise trading strategies, aiming to bridge the gap between theoretical advances and practical applications, and contributing to the development of intelligent trading systems.

1.3 Aims and Objectives

The main goal of the project is to leverage the combined strengths of DL and RL within a DRL framework, with the aim of creating advanced quantitative trading strategies designed specifically for the cryptocurrency market. Focusing on major cryptocurrencies such as Bitcoin (BTC) and Ethereum (ETH), the project aims to explore the capabilities of DRL in making efficient, real-time trading decisions. The goal is to develop a set of successful trading strategies, compare them to traditional strategies that rely on technical indicators, and ultimately validate them in a real-world trading system.

The main objective is divided into several sub-aspects to achieve:

1. Foundational Trading Knowledge: The first step is to conduct basic research on trading principles, trading processes, and specific aspects of cryptocurrency trading. This initial step ensures that our DRL model is built on a solid understanding of trading mechanisms and the markets.

2. Literature Analysis: Conduct a critical review of the current state of quantitative trading research, identifying existing methodologies and their application in the cryptocurrency market, and pointing out prevalent challenges.

3. Historical Data Examination: Analyse historical price data of key cryptocurrencies to understand market volatility and identify trend patterns, informing the strategy development phase.

4. Development of DRL Frameworks: Train a novel DRL-based trading strategy that adapts

to the volatility of the cryptocurrency market and achieves objective profitability. The approach will focus on efficiently handling high-dimensional data and designing reward functions that accurately respond to changes in the market environment.

5. Benchmark Strategy Comparisons: Evaluate the performance of our DRL model against established trading strategies based on technical indicators, assessing efficiency and profitability.

6. Real-World Trading Implementation: Deploy the developed DRL strategy in live cryptocurrency trading environments to test its practical applicability and performance under real market conditions.

1.4 Study Design

The development of machine learning models in the field of data science and analytics often employs the CRISP-DM methodology, a standard recognised for its systematic and comprehensive approach to data mining projects [41]. In this project, the CRISP-DM methodology will be used to outline the overall structure of the research, ensuring a rigorous and structured progression from conceptualisation to deployment. The CRISP-DM framework is shown in the Figure 1.1 [2], it provides a powerful template for addressing the complexity of deploying DRL strategies in the cryptocurrency trading domain [31].

The project started with a thorough business understanding phase, where the project goals and requirements were defined from a business perspective and translated into a data mining problem. Next, the data understanding phase involved the initial collection and exploration of data to identify quality issues and gain initial insights. The subsequent data preparation phase focuses on building the final dataset through a selection, cleaning and transformation process. Modelling then entails the application and calibration of various DRL models, followed by an evaluation phase where these models are assessed against business objectives and predefined benchmarks. Finally, the deployment phase implements the refined DRL strategy into a live trading environment, complete with performance monitoring and iterative improvement mechanisms.

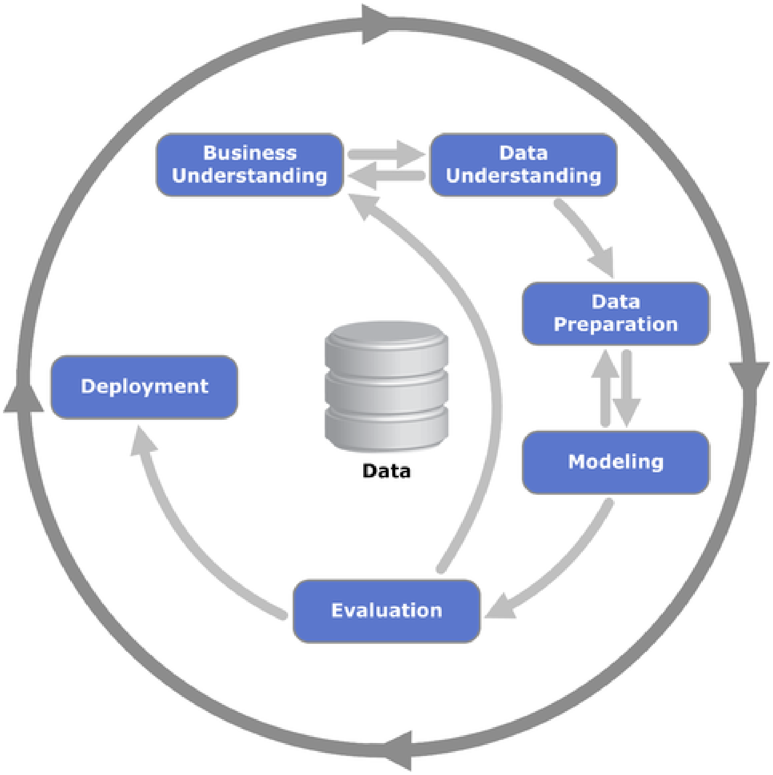


Figure 1.1: CRISP-DM Process Diagram

1.5 Thesis Structure

The thesis is structured as follows: Chapter 1 introduces the project through motivation, objectives and research design. Chapter 2 reviews the literature on trading strategies and reinforcement learning. Chapter 3 describes the project methodology in detail, including data processing, model construction and evaluation strategies. Chapter 4 discusses the experimental and evaluation results. Finally, Chapter 5 provides a summary, reflections on project management, and learning outcomes.

Chapter 2

Literature Review

This chapter outlines the progression of financial trading strategies, starting with Section 2.1 on traditional fundamental and technical analyses. Section 2.2 explores the rise of algorithmic trading and machine learning's role in enhancing trading systems. Section 2.3 introduces reinforcement learning's innovative approach to trading through market interaction. Section 2.4 reviews DRL methods, including critic-only strategies, actor-only strategies and actor-critic strategies, highlighting the potential of these methods to transform the practice of financial trading.

2.1 Fundamental Analysis and Technical Analysis

In financial trading, conventional methods rely heavily on fundamental and technical analysis, each providing unique insights into market evaluation. These traditional strategies can yield profits but necessitate significant time, knowledge, and experience from the participants [10][9]. Fang et al. points out that traders employ technical and fundamental analyses or a combination of both to evaluate financial markets, with a significant portion of research favoring technical analysis due to its reliance on backtestable, quantifiable data [16]. It is also noted that systematic trading and econometric methods are gaining attention in cryptocurrency trading, offering strategies based on historical data patterns, and are sometimes integrated with machine learning technologies for enhanced prediction and trading efficiency [16]. This evolution has highlighted the importance of adaptability and innovation in the changing financial market environment and has led to a re-evaluation of traditional approaches in favour of more sophisticated and technologically advanced strategies.

Fundamental analysis offers insight into the intrinsic value of a security by scrutinising a wide range of economic, financial, and qualitative and quantitative factors. It involves a comprehensive study of the general economy, industry conditions, and a company's financial condition and governance, focusing on metrics such as earnings, expenses, assets and liabilities [5][51]. This approach is characterised by a nuanced approach to assessing a company's true value, and is distinguished by its in-depth study and consideration of complex and unstructured data [35]. However, it may ignore short-term market volatility and emphasise the entity behind the stock rather than direct market trends.

Technical analysis, distinct from fundamental analysis, focuses on detecting patterns in market data to forecast future market trends by examining details like stock prices and trading volumes. This approach utilises various charting tools and indicators to analyse price actions and trading signals. Key technical indicators include Moving Average Convergence Divergence (MACD) [6], Relative Strength Index (RSI) [49], Commodity Channel Index (CCI) [32], Moving Average (MA) and Exponential Moving Average (EMA) [22]. The theory posits that past trading data can effectively predict future price movements [17]. Although this method predominantly concentrates on short-term trends and signals, it might overlook rapid economic changes or unforeseen news that can significantly affect market prices.

2.2 Algorithmic Trading

Algorithmic trading involves the deployment of computer systems and programs that automatically carry out trades in financial markets based on specific pre-set criteria. These automated systems surpass human traders in their ability to analyse market data and execute trades both rapidly and voluminously. This form of trading, which leverages complex algorithms, often results in more efficient and effective market transactions by optimising decision-making regarding timing, price, and volume [3]. With advancements in machine learning and artificial intelligence, algorithmic trading has become increasingly sophisticated and adaptable. Machine learning algorithms are particularly adept at analysing extensive financial data, learning from market trends, and making predictive decisions. These algorithms are capable of adjusting to new information and fluctuating market conditions, which enhances their suitability for dynamic trading strategies [11].

Rather than employing machine learning for predicting asset prices or identifying patterns in price volatility, this project will adopt a RL approach. This method allows for direct interaction with the market environment without predefined rules, enabling the trained agent to make trading decisions autonomously in the market.

2.3 Reinforcement Learning

Reinforcement Learning is a fundamental machine learning approach where an agent learns to make decisions through interactions with an environment to attain specific goals. Unlike other machine learning methods like supervised and unsupervised learning, the primary mechanism of RL involves learning from the outcomes of actions instead of direct instruction, setting it apart from other forms [27]. In RL, agents aren't trained using pre-labeled data but instead through experiences they gather themselves. Moreover, the main objective of their training is to maximize their potential rewards rather than discovering patterns within the data [45].

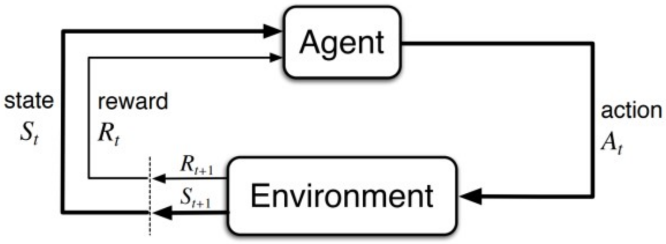


Figure 2.1: Agent and environment interaction

In the reinforcement learning setup, the agent operates independently to decide on its actions, and the environment encompasses elements beyond the agent's control. At each timestep t, the agent chooses an action A_t based on its current state S_t. Subsequently, the environment responds at the next timestep R_{t+1} by providing a reward R_{t+1} for the chosen action. Actions that conform to the objectives set by the system's designer are rewarded positively, whereas actions that deviate from these goals receive negative rewards. The dynamics of the interactions between the agent and the environment are depicted in Figure 2.1 [31]. In RL, the influence exerted by each agent's interaction with the environment is predominantly determined by the reward function. Consequently, the design of a cogent and effective reward function is paramount throughout the entire training process. An optimally designed reward function must balance informativeness, facilitating the model's convergence during training, with an appropriate level of sparsity, which enhances the reward's interpretability [14].

2.4 Deep Reinforcement Learning

This section explores academic work on DRL in the field of financial trading. The review focuses on the core challenges encountered when applying reinforcement learning to financial trading.

2.4.1 Critic-only and Actor-only Deep Reinforcement Learning

It is vital to grasp the fundamental limitations of the critic method, particularly the Deep Q-Network (DQN) and its variants, which are extensively recognized in this field of research [52]. The main limitation of DQN is its design for discrete action spaces, necessitating adaptations for use in continuous environments. Although DQN is effective in discrete state scenarios, its application to continuously varying stock prices presents considerable challenges. This problem is exacerbated when managing multiple stocks and assets, leading to a dramatic expansion in the dimensions of the state and action spaces [29]. Moreover, the design of the reward function is crucial, especially since critic-only methods are highly sensitive to the rewards they receive from the environment [50].

To address these challenges, recent studies have proposed various solutions. For example, the extension of DQN through algorithms like Deep Deterministic Policy Gradient (DDPG) and Double DQN (DDQN) enables handling continuous action spaces by integrating a policy-based approach with the critic method [28][46]. These algorithms extend DQN's applicability to continuous action spaces, offering a more sophisticated means of interacting with complex environments by refining the policy that dictates optimal actions in given states. Furthermore, the intricacies of constructing effective reward functions have spurred the development of novel strategies aimed at enhancing the efficiency and performance of learning models. Approaches such as reward shaping, which involves tailoring reward functions to incorporate domain-specific knowledge, alongside the utilization of deterministic policy gradient algorithms, have demonstrated significant potential in augmenting the adaptability and robustness of models primarily reliant on critic methods [36][44].

Actor-only methods present a considerable advantage by learning a policy that directly accommodates a continuous action space, effectively learning a mapping for actions in specific states, which is not restricted to being discrete [39]. However, as highlighted in the research by Zhang et al., a significant challenge with this approach is the extended training duration required to learn optimal policies [52]. This prolonged training period is especially notable in trading environments, where a vast number of samples are necessary for effective training. In the absence of sufficient samples, there is a risk that suboptimal actions might be misinterpreted as beneficial if they result in high total rewards. This risk arises because, unlike the critic-only approach, there isn't a quantifiable value assigned to each state and action that would otherwise guide expectations of outcomes. Nonetheless, the concern regarding extended training times remains a topic of debate, as some studies have demonstrated the potential for faster learning convergence [19].

2.4.2 Actor-Critic Deep Reinforcement Learning

The actor-critic model in RL uniquely trains two models simultaneously: the actor, which determines the agent's actions in given states, and the critic, which evaluates the quality of these actions. This dual-model strategy has become one of the most effective approaches in the field, with state-of-the-art actor-critic DRL algorithms such as Proximal Policy Optimization (PPO) in its original actor-critic format and Advantage Actor-Critic (A2C) effectively addressing complex environments [42]. Despite these advances, as of 2019, actor-critic methods were still relatively underexplored in DRL for trading, with only a few additional studies emerging since then [52]. Algorithms like PPO are particularly adept at managing challenges common in applying RL to complex settings. A prominent issue is the dependence of training data on the evolving policy, which leads to changing data distributions in observations and rewards as learning progresses, often resulting in instability. Additionally, the high sensitivity of RL to initial conditions and hyperparameters is a notable challenge. Drastic policy changes, possibly triggered by a high learning rate, can direct the agent into less advantageous areas of the search space. This can lead to data collection based on suboptimal policies, potentially impeding the agent's capacity to recover or improve [39].

Recent advancements in actor-critic methods have been notable in the field of robotic manipulation. DRL systems have shown significant success in tasks requiring delicate handling and manipulation, indicative of the algorithm's versatility and effectiveness in real-world applications. This success is attributed to the dual-model strategy, where the actor's policy-based actions are refined through continuous feedback from the critic's value-based assessments, optimizing both decision-making and performance in dynamic and uncertain environments [21]. Moreover, the incorporation of assorted actor-critic deep reinforcement learning with hindsight experience replay (AACHER) exemplifies the innovation in addressing challenges of continuous control and sample efficiency. AACHER, building upon the strengths of the Deep Deterministic Policy Gradient (DDPG) method, demonstrates the adaptability and resilience of actor-critic algorithms in learning from a diverse set of experiences, thus enhancing their performance in complex scenarios [43]. Model Predictive Actor-Critic (MoPAC) represents a notable leap in the application of model-based reinforcement learning. MoPAC leverages the theoretical guarantees of actor-critic methods to accelerate skill acquisition in robots, showing how learning dynamics models in tandem with policies can achieve near-optimal performance. This methodology underscores the actor-critic model's capacity for monotonic improvement when applied to learned dynamics, showcasing its potential in enhancing learning efficiency and outcome predictability in robotics and other high-dimensional, dynamic systems [34].

Chapter 3

Methodologies

This chapter outlines the methodology behind the development of DRL trading agents for the cryptocurrency market. The project is structured into three main layers: data processing, reinforcement learning environment, and agent development and training. This layered framework facilitates a comprehensive approach from initial data processing to the complex decision-making process required for successful trading strategies. The approach emphasises iterative enhancement, incorporating feedback loops to continuously improve agent performance and adaptability.

3.1 Project Structure

The project architecture designed for the thesis is divided into three layers focusing on the creation of a Deep Reinforcement Learning trading agent. The overall structure is shown in the figure 3.1. The data layer is responsible for acquiring, cleaning and enhancing transactional data through feature engineering. The environment layer consists of a reinforcement learning setup that allows for iterative strategy testing and improvement based on the state-action-reward principle. Finally, the agent layer details the implementation of the DRL agent, including training, validation, and testing phases for final application to paper or real-time cryptocurrency transactions. Iterative improvement is a recurring theme throughout the lifecycle of a project, as reflected in the feedback loops for adjusting parameters and environment settings, showing the agility of the development process.

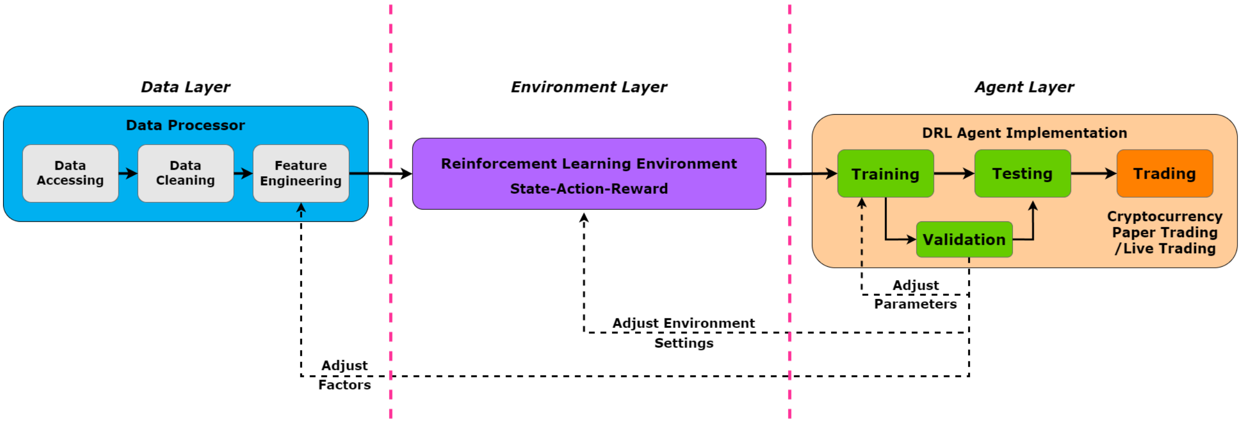


Figure 3.1: Project Structure

3.2 Data Processor

The training and testing data for the deep reinforcement learning model comes from the cryptocurrency (Bitcoin) price history on Yahoo Finance [18]. This resource has a comprehensive collection of historical price data for Bitcoin, which includes daily opening, high, low, closing, and trading volume. Data cleaning is a key step in the data preparation process and includes removing any outliers or erroneous entries, filling in missing values through interpolation or forward/backward padding, and organising the data in time-series order.

3.2.1 Technical Indicators

In order to capture subtle changes in the cryptocurrency market, some technical indicators will be calculated based on the OHLCV (Open, High, Low, Close, Volume) data, and used for training. The following technical indicators are added:

- Moving Average (MA) of 5, 10, 30 and 90 periods
- Relative Strength Index (RSI) of 14 periods
- DIF, DEA, MACD, with $fastperiod = 12$, $slowperiod = 26$, $signalperiod = 9$
- KDJ, with $period = 9$, $kperiod = 3$, $dperiod = 3$

A detailed description of technical indicators and related strategies is in Section 3.4.

3.2.2 Data Splitting

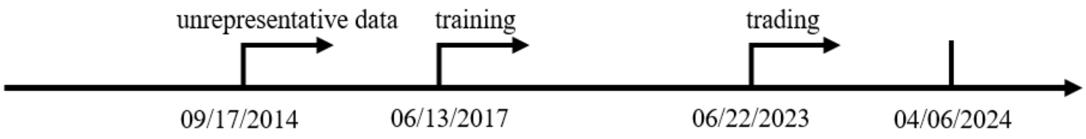


Figure 3.2: Data splitting

As shown in the Figure 3.2, the dataset is divided into three parts: unrepresentative data, training and trading. The agent will use the data from June 13^{th}, 2017 to June 22^{nd}, 2023 for training, and from June 22^{nd}, 2023 to April 6^{th}, 2024 for trading, to validate the performance. The data from 17^{th} September 2014 to 13^{th} June 2017 was dropped because this part of the data is not representative. As can be seen from the overall Bitcoin trend in Figure 3.3, the Bitcoin market did not have enough attention prior to 2017, and the market volatility and volume were so low that it did not help our agent's learning, so data from that time period was not included in the training.

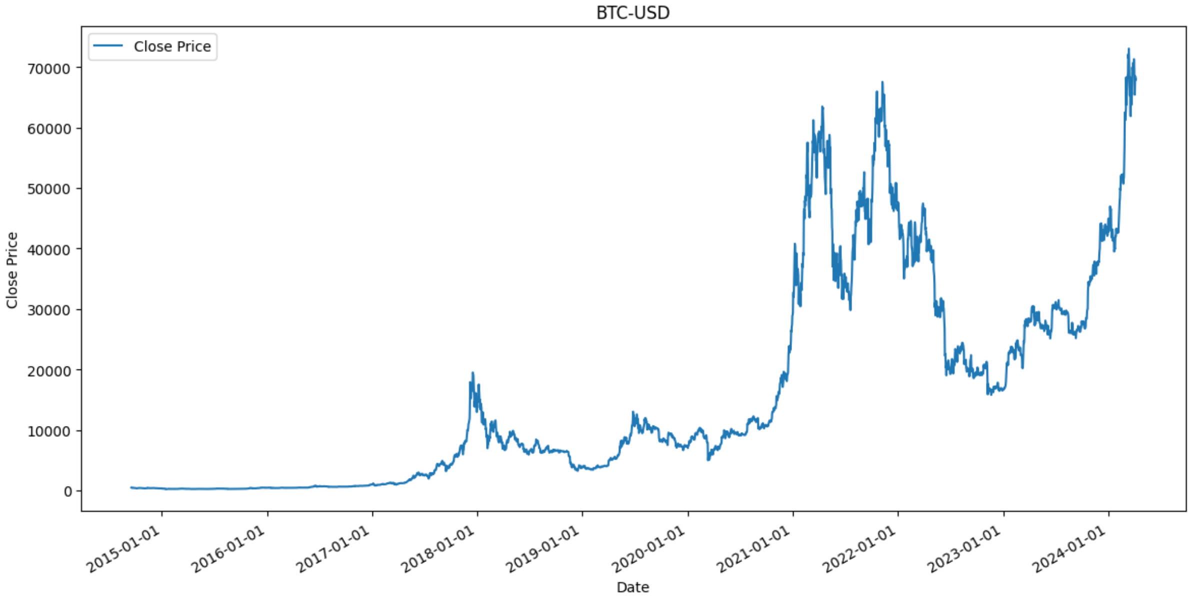


Figure 3.3: BTC Trends

3.3 Modeling

The environment and the agents are meticulously designed and rigorously validated by myself. The RL algorithms used in this project are implemented by the Stable Baselines3 (SB3) library [40]. This section contains the modelling of the entire trading environment, including the selection of the neural network, the design of the observation space and the design of the reward function. The structure of the model is shown in Figure 3.4 and contains all the interactions of the agent with the environment as well as the structure of the agent.

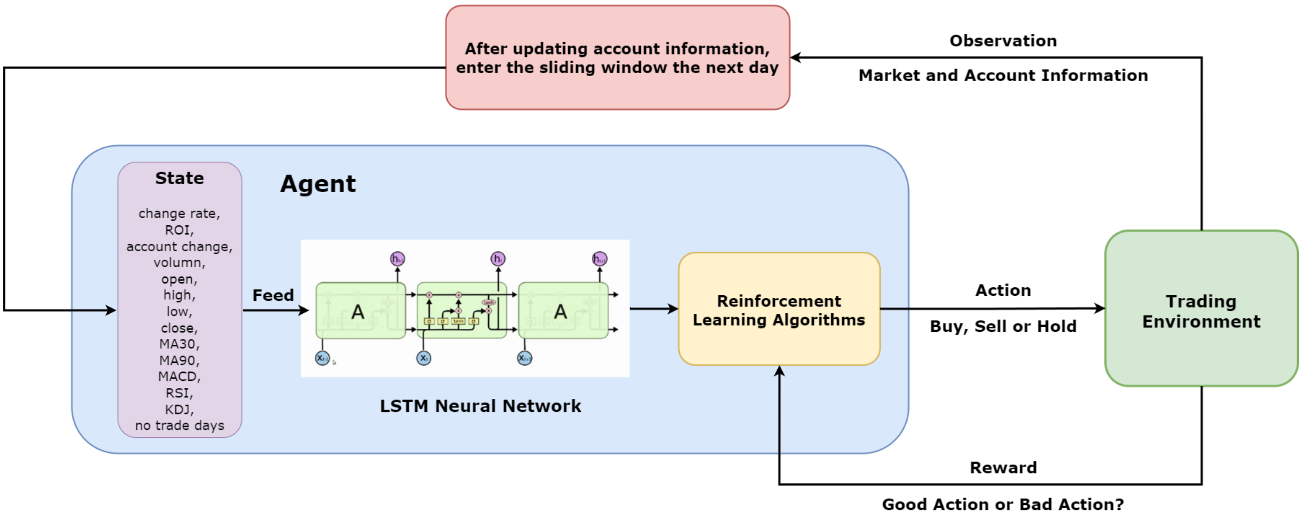

After updating account information, enter the sliding window the next day
Observation
Market and Account Information
Agent
State
change rate, ROI, account change, volumn, open, high, low, close, MA30, MA90, MACD, RSI, KDJ, no trade days
Feed
A
A
LSTM Neural Network
Reinforcement Learning Algorithms
Action
Buy, Sell or Hold
Trading Environment
Reward
Good Action or Bad Action?

Figure 3.4: Model Structure

3.3.1 Neural Network

In financial trading, data is time-sequential and it is crucial to understand the order in which events occur. Long Short-Term Memory (LSTM) networks are particularly well suited to this task as they are able to process sequences and remember information over long periods of time. This makes them a reliable choice for analysing financial time series data, where each point is a step in the sequence with potential implications for the future. The model includes a custom LSTM extractor designed to process the data series, which starts with a specially tailored LSTM layer that processes 16 features at each time step, mapping them into a high-dimensional space containing 256 features.

3.3.2 Observation Space

The environment space observed by agents at each step. A sliding window approach is used to focus on recent market history. At each time point t, the observation environment contains a sliding window size of data, in this project the sliding window is set to 10 days. The data for time step t is the last day in the entire batch, and the number of data entries in the entire batch is the number of days in the sliding window. The data for each day contains:

- Rate of change of each day's price from the previous day's price
- ROI: total profit on the account divided by the initial money hold by the account and scaled down by a factor of 10
- The change in the account's net worth (market value of cryptocurrency holdings + cash) from today to the previous day, divided by the initial money owned and scaled down by a factor of 10.
- The day's trading volume scaled down by a factor of 10e11.
- The day's crypto price scaled down by 10e5
- The day's open price scaled down by 10e5
- The day's highest price scaled down by 10e5
- The day's lowest price scaled down by 10e5
- Value of 30-day period moving average at t, scaled down by 10e5
- Value of 90-day period moving average at t, scaled down by 10e5
- The day's MACD value scaled down by 10e4
- The day's RSI value scaled down by 100
- Number of consecutive days without generating trading signals
- K value in the KDJ indicator for the day, scaled down by 100
- D value in the KDJ indicator for the day, scaled down by 100

- J value in the KDJ indicator for the day, scaled down by 100

The purpose of scaling the observations is to maintain all the data within the same similar scale, to balance the impact of each indicator on the agent, and to avoid a disproportionate impact of a particular data.

The required input data format for LSTM is:

tensor(sequent_length, batch_size, input_size)

which is

tensor(1, window_size, 16)

in this project, when window. Here is an example to visualise the data for each state for easier understanding:

Assuming that one wants to obtain the observation space on day 70, that is, $t = 70$, and the window size is 10. The observations for this state are:

(tensor(0, 0, Data at day 61: ROI, price, MA, KDJ...)
tensor(0, 1, Data at day 62: ROI, price, MA, KDJ...)
tensor(0, 2, Data at day 63: ROI, price, MA, KDJ...)
tensor(0, 3, Data at day 64: ROI, price, MA, KDJ...)
tensor(0, 4, Data at day 65: ROI, price, MA, KDJ...)
tensor(0, 5, Data at day 66: ROI, price, MA, KDJ...)
tensor(0, 6, Data at day 67: ROI, price, MA, KDJ...)
tensor(0, 7, Data at day 68: ROI, price, MA, KDJ...)
tensor(0, 8, Data at day 69: ROI, price, MA, KDJ...)
tensor(0, 9, Data at day 70: ROI, price, MA, KDJ...))

3.3.3 Reward Design

When designing a reward function for a trading system, both market trends and account dynamics need to be taken into account so that each step of the process can affect the observation space. The basis of our reward mechanism is the percentage change in market trend between two consecutive points in time (today and tomorrow, i.e. t and t+1), which visually reflects the potential profit or loss of a held or traded position. Many of the rises and falls in the stock market are actually meaningless rises and falls (small rises or small downward pullbacks), which if taken into account would cause the model to struggle to extract some valid information. Therefore, agent adjusts the reward based on the absolute size of the previously calculated percentage. For relatively small fluctuations (less than 1.5% absolute change), agent judges that the market is essentially stable or insignificant for our trading strategy and therefore scales back the reward to 20% of the original value to discourage overreaction to smaller fluctuations. For movements between 1.5% and 3%, agent will judge these movements to be significant but not dramatic and adjust the reward to 70% to balance risk and opportunity. For larger movements (more than 5% in absolute terms), the reward is scaled back to 10% of the original value. This design allows the agent to remain cautious in the market and avoid being overly sensitive to market dynamics.

In addition, the rewards feature takes into account changes in account net worth to promote healthy trading habits and long-term profitability. Changes in portfolio net worth are added to the reward, minus the trading service fee, and scaled down proportionally to ensure that it does not overshadow the main trend-based reward, but still incentivises overall portfolio growth. Similarly, a small bonus based on the ratio of total profit to initial capital rewards the long-term profitability of the agent.

Penalties have also been introduced to discourage the behaviour of not making any trades. If there are no trades for more than 10 days, but less than 30 days, and this is not due to a fall in the market, this inactivity is penalised. If there are no trades for more than 30 days and the market is not on a downward trend, the penalties are more severe. This is designed to encourage a more dynamic trading strategy and the penalty increases with the length of inactivity.

Here is the pseudocode for the reward function:

Algorithm 1 Calculate Reward

```
Calculate the reward based on trend change: $reward = \frac{trend[t+1]-trend[t]}{trend[t]}$
if $|reward| \leq 0.015$ then
    $reward = reward \times 0.2$
else if $|reward| \leq 0.03$ then
    $reward = reward \times 0.7$
else if $|reward| \geq 0.05$ then
    if $reward < 0$ then
        $reward = (reward + 0.05) \times 0.1 - 0.05$
    else
        $reward = (reward - 0.05) \times 0.1 + 0.05$
    end if
end if
$reward = reward + (value\_change \times 0.00001) + (\frac{total\_profit}{init\_money} \times 0.01)$
if no_trade_days $> 10$ AND no_trade_days $\leq 30$ AND NOT fall then
    $reward = reward - 1$
else if no_trade_days $> 30$ AND NOT fall then
    $reward = reward - 5$
end if
```

In essence, such rewards are designed to strike a balance between encouraging profitable trading and managing risk while incentivising active participation in the market.

3.3.4 Overall Modelling

The stock trading process can be modelled as a Markov Decision Process (MDP):

- State s: The environment space observed by agents at each step, have been detailed introduced in Section 3.3.2.
- Action a: a set of actions that the agent can take at each step. The available actions are buying, selling, and holding, which will result in increasing, decreasing and no change of the holdings (h) respectively.

The specific actions are explained as follows, time t will increase to $t+1$ at each step:

- Buying: k coins can be bought and it leads to $h_{t+1} = h_t + k$. The minimum value of k is set to 0.01, which means that each lot represents 0.01 bitcoins. Each purchase is set as a full buy with a buy commission of 0.1%, set according to Binance's commission [8].
- Selling: $k(k \in [0.01, h])$ coins can be sold from the current account. k must be an positive number, $h_{t+1} = h_t - k$. Each sale is a full position sale with the same commission as a buy.
- Holding: $k = 0$ and no change in h_t, $h_{t+1} = h_t$

- Reward $r(s, a, s^{'})$: the reward is detailed introduced in the Section 3.3.3, reward will be given when action a is taken at state s, leading to a new state $s^{'}$. The total account value is the market value of cryptocurrency holdings plus cash.

- Policy $\pi(s)$: the trading strategy (choose which action in the action space) at state s. It reflects the probability of each action a at state s.

- Value function $Q_\pi(s, a)$: the rewards expected to be achieved at state s by action a following policy π.

Before training start, t is started from day 0 and the initial balance in the account is set to be \$10,000. Both h and $Q_\pi(s, a)$ are 0 at beginning, and $\pi(s)$ is uniformly distributed among all actions for any state. Then, the agent will learn from the history data using different DRL algorithms.

3.4 Evaluations

The performance of the trained agent will be tested using the trading dataset as shown in the Figure 3.2. It will be compared with multiple benchmark trading strategies and the Bitcoin trends, Return on Investment (ROI) is the main evaluation criteria. The benchmark strategies are introduced below:

3.4.1 MA Strategy

Introduction and Calculation

A Moving Average (MA) is calculated to identify the direction of a stock's trend or to determine its support and resistance levels. The moving average crossover strategy is a basic analytical tool that involves the use of moving averages over different time periods to identify potential upward or downward trends [23]. For this strategy, we consider a 5-period moving average (MA5) and a 10-period moving average (MA10).

$$MA5_t = \frac{1}{5} \sum_{i=t-4}^{t} \text{CLOSE}_i \tag{3.1}$$

$$MA10_t = \frac{1}{10} \sum_{i=t-9}^{t} \text{CLOSE}_i \tag{3.2}$$

Trading Signal

The strategy generates a bullish signal when the shorter moving average crosses the longer moving average, suggesting upside momentum and a potential buying opportunity.

$$\text{if } MA5_{t-1} < MA10_{t-1} \text{ and } MA5_t > MA10_t \tag{3.3}$$

Conversely, a bearish signal is generated when the shorter moving average falls below the longer moving average, indicating downward momentum and a potential selling opportunity.

$$\text{if } MA5_{t-1} > MA10_{t-1} \text{ and } MA5_t \leq MA10_t \tag{3.4}$$

3.4.2 MACD Strategy

Introduction and Calculation

Moving Average Convergence Divergence (MACD) is a widely used technical analysis indicator for measuring market momentum and potential price reversals. It was developed by Gerald Appel and is based on the difference between two moving averages of different lengths [24].

The core of MACD is the calculation of two Exponential Moving Averages (EMAs) with different weights (12 days and 26 days respectively in this project), from which signal and MACD lines are derived.

Short-term Exponential Moving Average (EMA):

$$S_t = \left(\frac{2 \times \text{CLOSE}_t + 11 \times S_{t-1}}{13}\right) \tag{3.5}$$

Long-term Exponential Moving Average (EMA):

$$L_t = \left(\frac{2 \times \text{CLOSE}_t + 25 \times L_{t-1}}{27}\right) \tag{3.6}$$

with initial conditions set as $S_0 = L_0 = CLOSE_o$

The difference between the short-term and long-term EMAs is the DIF:

$$\text{DIF}_t = S_t - L_t \tag{3.7}$$

The DEA line, a further smoothed version of the DIF using the factor 9, is computed as:

$$\text{DEA}_t = \left(\frac{2 \times \text{DIF}_t + 8 \times \text{DEA}_{t-1}}{10}\right) \tag{3.8}$$

with the initial condition $DEA_0 = DIF_0$

The MACD line itself is twice the difference between the DIF and DEA:

$$\mathrm{MACD}_t = (\mathrm{DIF}_t - \mathrm{DEA}_t) \times 2 \tag{3.9}$$

Trading Signals

A short-term trend breaking out of a long-term trend is a buy point, and a short-term trend being broken by a long-term trend is a sell point.

Buy when:

$$\mathrm{DIF}_{t-1} < \mathrm{DEA}_{t-1} \quad \text{and} \quad \mathrm{DIF}_t > \mathrm{DEA}_t \tag{3.10}$$

Sell when:

$$\mathrm{DIF}_{t-1} > \mathrm{DEA}_{t-1} \quad \text{and} \quad \mathrm{DIF}_t < \mathrm{DEA}_t \tag{3.11}$$

Also, The difference between the short-term trend and the long-term trend breaks through the critical point (0) upwards as a buy point, and the difference between the short-term trend and the long-term trend breaks through the critical point (0) downwards as a sell point.

Buy when:

$$\mathrm{MACD}_{t-1} < 0 \quad \text{and} \quad \mathrm{MACD}_t > 0 \tag{3.12}$$

Sell when:

$$\mathrm{MACD}_{t-1} > 0 \quad \text{and} \quad \mathrm{MACD}_t < 0 \tag{3.13}$$

3.4.3 KDJ Strategy

Introduction and Calculation

The KDJ indicator is a momentum indicator originally conceived by George Lane [48]. Its primary function is to predict the momentum of stock prices by comparing the closing price to its price range over a given period.

At the core of the KDJ indicator is the Raw Stochastic Value (RSV), which is computed as:

$$RSV_t = \frac{CLOSE_t - \min(LOW_t)}{\max(HIGH_t) - \min(LOW_t)} \times 100 \tag{3.14}$$

where:

- $CLOSE_t$ represents the closing price at time t,
- LOW_t denotes the lowest price observed over the period t,
- $HIGH_t$ signifies the highest price observed over the same period.

Typically, the period t encompasses the past 9 sessions, this period is also used in this project.

The K line is a smoothed version of the RSV and is derived using the following iterative formula:

$$K_t = \frac{RSV_t + (n-1) \times K_{t-1}}{n} \tag{3.15}$$

where n is the smoothing factor, the value of n determines how much weight is given to the past values of k. In this project, this smoothing factor is set to 3. For the initial value (K_0), $K_0 = RSV_0$.

The D line is a moving average of the K line, indicating the trend of the market. Its calculation is as follows:

$$D_t = \frac{K_t + (n-1) \times D_{t-1}}{n} \tag{3.16}$$

where n is also the smoothing factor, this smoothing factor is set to 3 for D line. The initial value of D_0 is set equal to K_0.

The J line serves as a predictor of future volatility and is determined by:

$$J_t = 3 \times K_t - 2 \times D_t \tag{3.17}$$

Interpretation and Trading Strategy

Overbought and Oversold Conditions:

- If D is above 80 and J is above 100, the market could be considered overbought, suggesting a potential sell signal.
- If both D is below 20 and J is below 0, the market could be considered oversold, suggesting a potential buy signal.

Crosses:

- The '*Golden Cross*' occurs when the J line crosses above the D line, especially when both are below 50, which can be considered a buying opportunity.
- Conversely, the '*Death Cross*' happens when the J line crosses below the D line while both are above 50, indicating a selling opportunity.

The trading strategy combines these two scenarios to generate buy and sell signals. Experiments have shown that reacting to market fluctuations through a combination of the three KDJ indicators has a high degree of accuracy [15], so this project uses a strategy based on this technical indicator as a baseline strategy.

3.4.4 Mixed Strategy: MA, MACD, RSI

The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and variability of price movements. Developed by J. Welles Wilder Jr. and fluctuating between 0 and 100, the RSI is often used to complement other technical indicators to confirm the direction of a security's price trend, momentum and potential turning points [25]. RSI is calculated using the following formula, where average gains and losses are based on a specified period, the project use the period of 14 days:

$$RSI = 100 - \frac{100}{1 + RS} \tag{3.18}$$

$$RS = \frac{\text{Average Gain over 14 days}}{\text{Average Loss over 14 days}} \tag{3.19}$$

The strategy integrates the Moving Average Convergence Divergence (MACD) indicator, the Relative Strength Index (RSI) and Moving Averages (MA) to generate buy and sell signals for the trading system. By combining these indicators, the strategy aims to capture momentum changes while filtering out potential false signals.

Buy Signal Conditions:

- The MACD value is greater than the DEA (signal line), and
- The RSI is below 70, indicates that the asset is not yet overbought, and
- The closing price is above the short-term moving average.

Sell Signal Conditions:

- The MACD value is less than the DEA, and
- The closing price is below the long-term moving average.

3.5 Continuous Adjustment

The final part of the methodology is continuous tuning. Based on parameter observations during agent training (e.g., loss, reward, value_loss) and comparison of the effects with the benchmark strategies, the environment settings, reward function design, and hyperparameters of the model are continuously adjusted to achieve better model results. This iterative refinement process is crucial to enhance the performance of the model. It involves a careful analysis of the results, leading to targeted adjustments to all aspects of the model, from data preprocessing to algorithmic tuning within the DRL agent. By adopting this approach, trading strategies are continually honed, ensuring that the agent is able to evolve in response to dynamic market conditions and achieve optimal decision-making efficiency.

Chapter 4

Results and Evaluations

4.1 Dataset Overview

The project is using a daily candlestick dataset of **Bitcoin** to **US Dollar** (BTC/USD). The dataset presented includes daily trading information between September 17^{th}, 2014, to April 6^{th}, 2024, and it was given a data pre-processing to clean the data and add technical indicators. The data includes 17 columns of financial metrics for each date: ’open’, ’high’, ’low’, ’close’, ’adj close’, ’volume’, ’Short_MA’, ’Long_MA’, ’MA_30’, ’MA_90’, ’DIF’, ’DEA’, ’MACD’, ’RSI’, ’K’, ’D’, ’J’. These metrics are critical for analysing market behavior and guiding the decision-making processes of the DRL trading agent. Below is a breakdown of these financial metrics:

- **Open**: Records the opening price of Bitcoin for the trading day, providing a baseline from which daily price movements are assessed.
- **High**: Indicates the highest price of Bitcoin reached during the trading day, reflecting the peak market valuation on that specific day and indicative of bullish trends when significantly exceeding the opening price.
- **Low**: Denotes the lowest price of Bitcoin within the day, signalling bearish market sentiments when there is a substantial deviation from the opening price.
- **Close**: Represents the closing price of Bitcoin, crucial as it reflects the final market consensus on price for the day and often serves as a reference for the next day’s opening.
- **Volume**: The total volume of Bitcoin traded during the day, where high trading volumes can indicate high interest or significant price movements in the market.

- **Short_MA**: A moving average with the period of 5 days, providing a quick look at Bitcoin's recent price trend over a shorter period, aiding in identifying immediate market momentum.

- **Long_MA**: A moving average with the period of 10 days, offering insights into the longer-term price trend of Bitcoin, instrumental in understanding the recent market direction.

- **MA_30**: The 30-day moving average provides a monthly trend of Bitcoin's price, smoothing out daily volatility to highlight broader market movements.

- **MA_90**: This 90-day moving average offers a quarterly perspective on the price trend, further smoothing market fluctuations to reveal sustained trends.

- **DIF**: Represents the difference between two exponential moving averages (EMAs) of Bitcoin's closing price, used in calculating the MACD indicator.

- **DEA**: The Exponential Moving Average of the DIF, acting as a signal line for the MACD indicator with crossovers indicating potential buy or sell signals.

- **MACD**: A trend-following momentum indicator that shows the relationship between the DIF and DEA, highlighting potential bullish or bearish phases.

- **RSI**: The Relative Strength Index assesses overbought or oversold conditions in the price of Bitcoin through the magnitude of recent price changes.

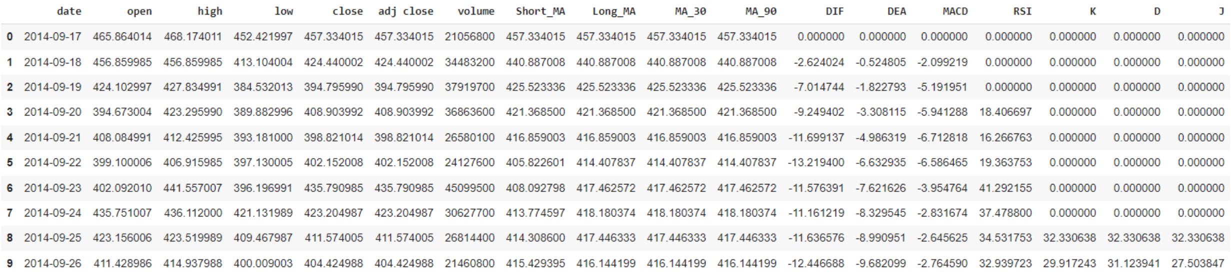
- **K**: A measure of current price relative to the high and low range over a given period, indicating momentum.

- **D**: The moving average of the 'K' value, smoothing the data to provide a clearer signal.

- **J**: Derived from the 'K' and 'D' values, 'J' can indicate potential price reversals due to overbought or oversold conditions.

Specific parameter settings and calculations for each indicator are detailed in Section 3.2.1.

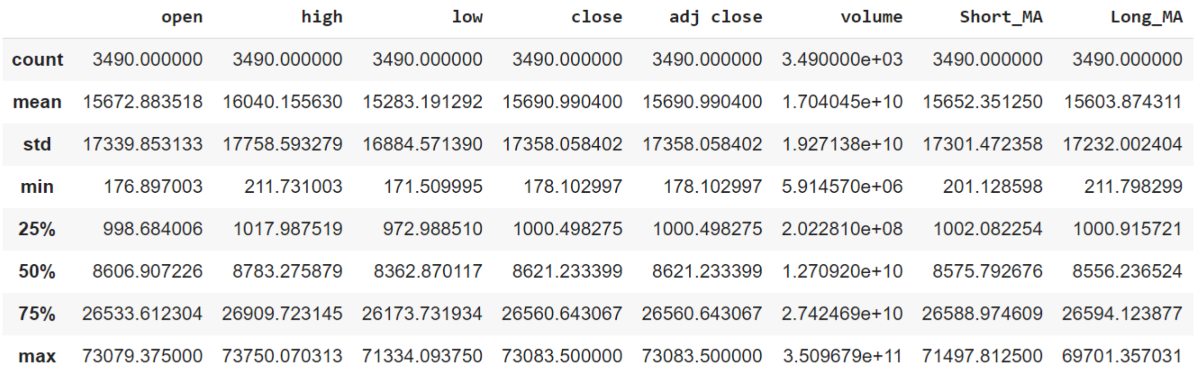
As shown in the Figure 4.1, this figure provides a snapshot of the data set. It is easy to find that the data are arranged in chronological order, and indicators that cannot be calculated are assigned a value of zero.



	date	open	high	low	close	adj close	volume	Short_MA	Long_MA	MA_30	MA_90	DIF	DEA	MACD	RSI	K	D	J
0	2014-09-17	465.864014	468.174011	452.421997	457.334015	457.334015	21056800	457.334015	457.334015	457.334015	457.334015	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1	2014-09-18	456.859985	456.859985	413.104004	424.440002	424.440002	34483200	440.887008	440.887008	440.887008	440.887008	-2.624024	-0.524805	-2.099219	0.000000	0.000000	0.000000	0.000000
2	2014-09-19	424.102997	427.834991	384.532013	394.795990	394.795990	37919700	425.523336	425.523336	425.523336	425.523336	-7.014744	-1.822793	-5.191951	0.000000	0.000000	0.000000	0.000000
3	2014-09-20	394.673004	423.295990	389.882996	408.903992	408.903992	36863600	421.368500	421.368500	421.368500	421.368500	-9.249402	-3.308115	-5.941288	18.406697	0.000000	0.000000	0.000000
4	2014-09-21	408.084991	412.425995	393.181000	398.821014	398.821014	26580100	416.859003	416.859003	416.859003	416.859003	-11.699137	-4.986319	-6.712818	16.266763	0.000000	0.000000	0.000000
5	2014-09-22	399.100006	406.915985	397.130005	402.152008	402.152008	24127600	405.822601	414.407837	414.407837	414.407837	-13.219400	-6.632935	-6.586465	19.363753	0.000000	0.000000	0.000000
6	2014-09-23	402.092010	441.557007	396.196991	435.790985	435.790985	45099500	408.092798	417.462572	417.462572	417.462572	-11.576391	-7.621626	-3.954764	41.292155	0.000000	0.000000	0.000000
7	2014-09-24	435.751007	436.112000	421.131989	423.204987	423.204987	30627700	413.774597	418.180374	418.180374	418.180374	-11.161219	-8.329545	-2.831674	37.478800	0.000000	0.000000	0.000000
8	2014-09-25	423.156006	423.519989	409.467987	411.574005	411.574005	26814400	414.308600	417.446333	417.446333	417.446333	-11.636576	-8.990951	-2.645625	34.531753	32.330638	32.330638	32.330638
9	2014-09-26	411.428986	414.937988	400.009003	404.424988	404.424988	21460800	415.429395	416.144199	416.144199	416.144199	-12.446688	-9.682099	-2.764590	32.939723	29.917243	31.123941	27.503847

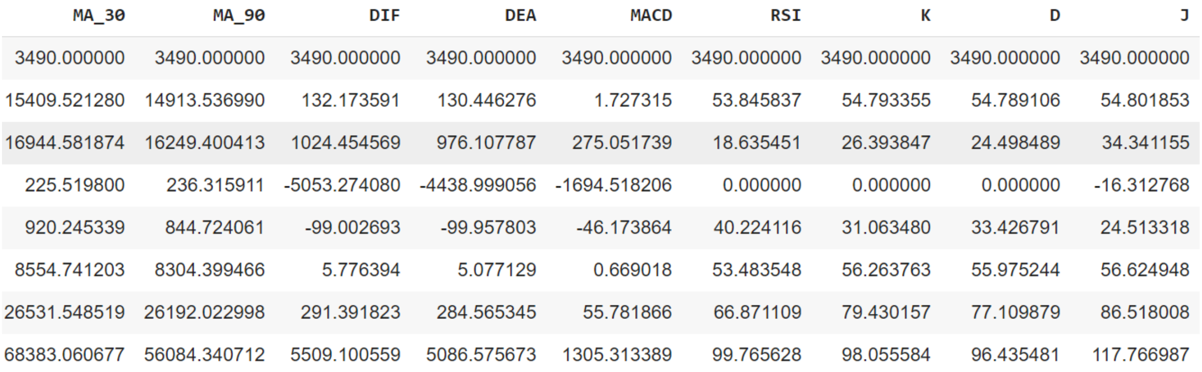
Figure 4.1: Data sample

Descriptive Statistics for the data set can be seen in Figure 4.2 and 4.3.



	open	high	low	close	adj close	volume	Short_MA	Long_MA
count	3490.000000	3490.000000	3490.000000	3490.000000	3490.000000	3.490000e+03	3490.000000	3490.000000
mean	15672.883518	16040.155630	15283.191292	15690.990400	15690.990400	1.704045e+10	15652.351250	15603.874311
std	17339.853133	17758.593279	16884.571390	17358.058402	17358.058402	1.927138e+10	17301.472358	17232.002404
min	176.897003	211.731003	171.509995	178.102997	178.102997	5.914570e+06	201.128598	211.798299
25%	998.684006	1017.987519	972.988510	1000.498275	1000.498275	2.022810e+08	1002.082254	1000.915721
50%	8606.907226	8783.275879	8362.870117	8621.233399	8621.233399	1.270920e+10	8575.792676	8556.236524
75%	26533.612304	26909.723145	26173.731934	26560.643067	26560.643067	2.742469e+10	26588.974609	26594.123877
max	73079.375000	73750.070313	71334.093750	73083.500000	73083.500000	3.509679e+11	71497.812500	69701.357031

Figure 4.2: Descriptive statistics of the BTC/USD trading pair dataset I



MA_30	MA_90	DIF	DEA	MACD	RSI	K	D	J
3490.000000	3490.000000	3490.000000	3490.000000	3490.000000	3490.000000	3490.000000	3490.000000	3490.000000
15409.521280	14913.536990	132.173591	130.446276	1.727315	53.845837	54.793355	54.789106	54.801853
16944.581874	16249.400413	1024.454569	976.107787	275.051739	18.635451	26.393847	24.498489	34.341155
225.519800	236.315911	-5053.274080	-4438.999056	-1694.518206	0.000000	0.000000	0.000000	-16.312768
920.245339	844.724061	-99.002693	-99.957803	-46.173864	40.224116	31.063480	33.426791	24.513318
8554.741203	8304.399466	5.776394	5.077129	0.669018	53.483548	56.263763	55.975244	56.624948
26531.548519	26192.022998	291.391823	284.565345	55.781866	66.871109	79.430157	77.109879	86.518008
68383.060677	56084.340712	5509.100559	5086.575673	1305.313389	99.765628	98.055584	96.435481	117.766987

Figure 4.3: Descriptive statistics of the BTC/USD trading pair dataset II

4.2 Experimental Settings

The project is written in Python and runs in Jupyter Notebook, with dependencies on external libraries such as gym, stable_baseline3 and torch. the code is divided into three main sections:

data collection and processing, model training and backtesting. the code is designed to be used in a variety of ways, including the following: data collection and processing, model training and backtesting. The agents are trained using two different DRL algorithms: PPO and DQN. These algorithms are implemented using stable_baseline3 library, which is a library provides the reliable implementation of reinforment learning algorithms. Both the training and testing environments have an initial capital of $100,000, and the overall returns are compared to different strategies after the trades are completed. The whole process has been carefully explained in the methodology chapter.

The hyperparameters used in the PPO algorithms:

- “n_steps”: 256,
- “ent_coef”: 0.00015,
- “learning_rate”: 0.0001,
- “batch_size”: 16,
- “n_epochs”: 20,
- “gae_lambda”: 0.8,
- “clip_range”: 0.4,
- “vf_coef”: 0.30244638,
- “max_grad_norm”: 0.7,
- “gamma”=0.9,
- “gae_lambda”=0.8,
- “vf_coef”=0.30244638,
- “max_grad_norm”=0.7,
- “target_kl”=None,
- “seed”=1,

The value of the PPO hyperparameters are inspired from a previous research [31] and continously adjusted according to the training results.

The hyperparameters used in the DQN algorithms:

- “learning_rate” = 0.0001,
- “buffer_size” = 4000,
- “learning_starts” = 1000,
- “batch_size” = 16,
- “tau” = 1.0 / 10,
- “gamma” = 0.98,
- “train_freq” = 1,
- “gradient_steps” = 1,
- “exploration_initial_eps” = 1,
- “exploration_final_eps” = 0.12,
- “exploration_fraction” = 0.1,
- “target_update_interval” = 10,
- “max_grad_norm” = 10,
- “seed” = 1

4.3 BackTest: Agents Validations

The validation of the agent’s training can be done by simulating trades using the data in the training set. For ease of verification, the agent’s trading data is only compared to Bitcoin trends to initially verify that the agent learns patterns from historical data.

The validation trading runs from 12^{th} February 2022 to 1^{st} December 2022, and here are the results from two different models using different algorithms:

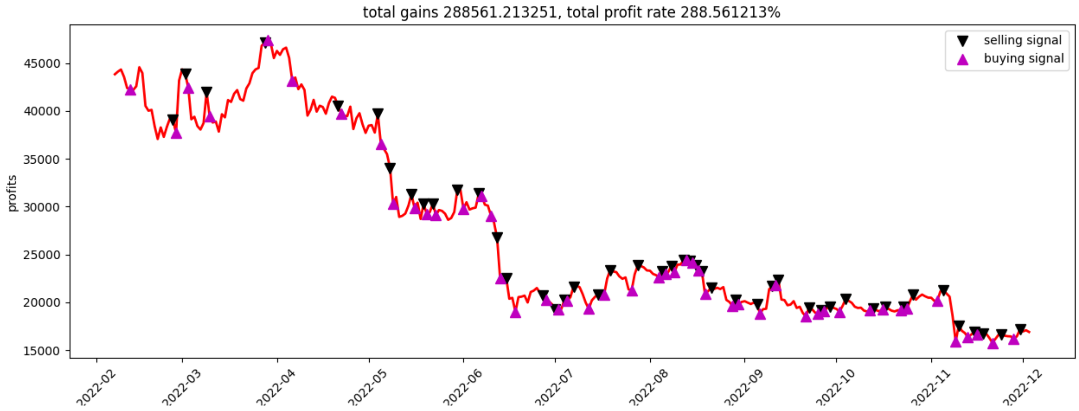


Figure 4.4: PPO BackTest Validation Trading Points

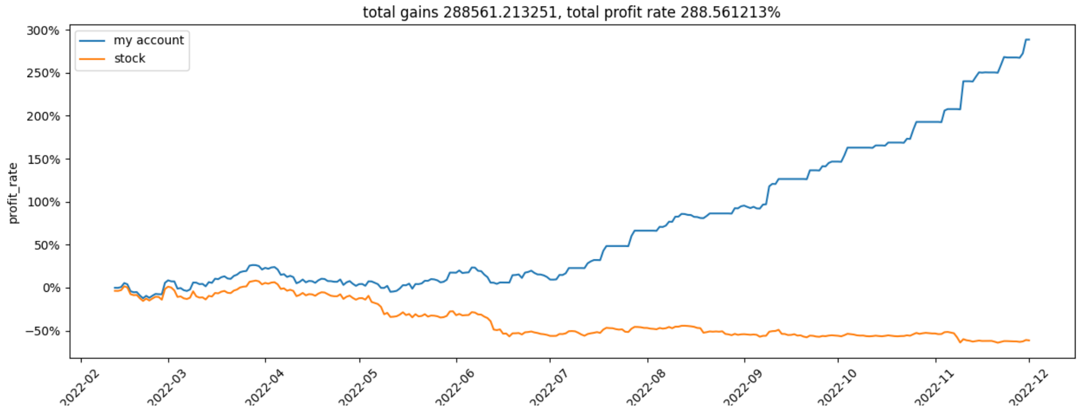


Figure 4.5: PPO BackTest Validation Trading Profit Rate

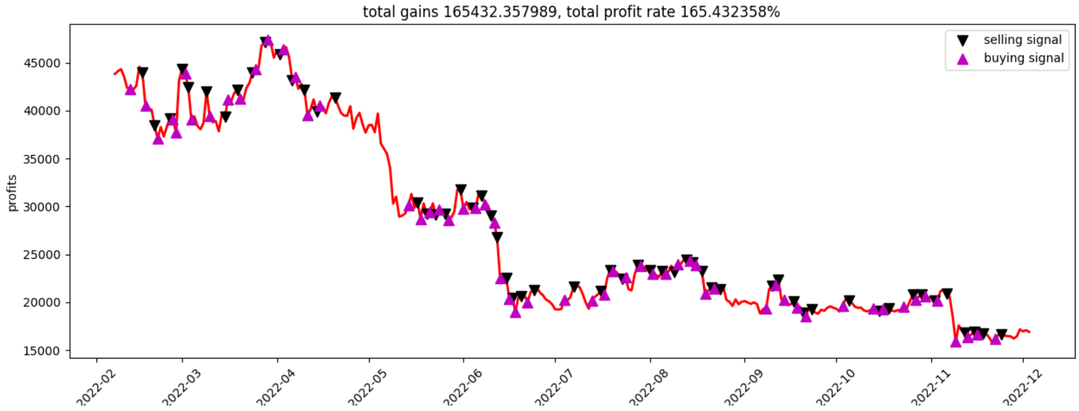


Figure 4.6: DQN BackTest Validation Trading Points

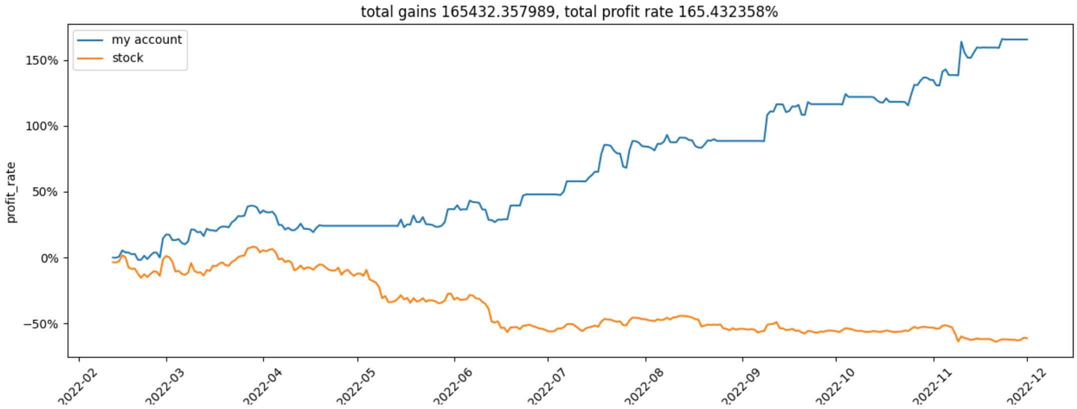


Figure 4.7: DQN BackTest Validation Trading Profit Rate

Figure 4.4 and 4.5 show the trading points and profit rate of PPO agent, while Figure 4.6 and 4.7 show for the DQN agent. Both agents have shown strong profitability, achieving exponential profits in a year when the bitcoin market was down close to 60%. The intuitive numerical results of the trading can be found in Table 4.1.

Table 4.1: Validation Trading Results

	Start Value	**End Value**	**Rate of Change**
BTC-USD Trends	42244.46875	17088.660156	-59.548171%
PPO Agent	100000.0000	388561.2132513294	288.561213%
DQN Agent	100000.0000	265432.3579891072	165.432358%

This proves that the agent's design is sound, and the next part will be live trading on a test set that the agent has never seen before.

4.4 BackTest: Real Trading

The test set runs from 22^{nd} June 2023 to 6^{th} April 2024, with actual trading taking place between 27^{th} June 2023 and 4^{th} April 2024 due to sliding window settings. Here is the results:

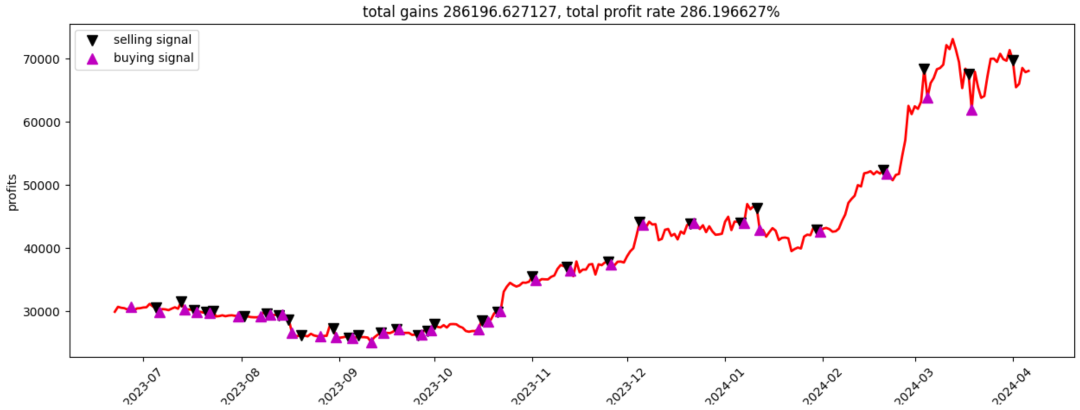


Figure 4.8: PPO BackTest Trading Points

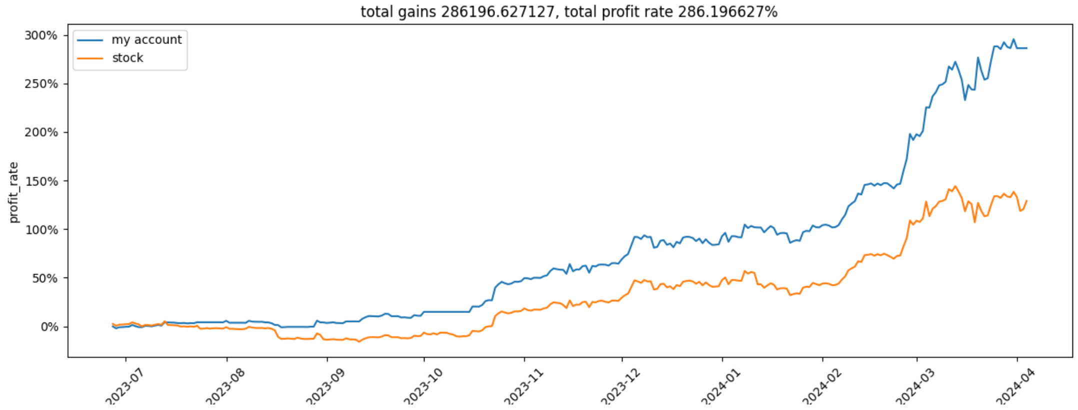


Figure 4.9: PPO BackTest Trading Profit Rate

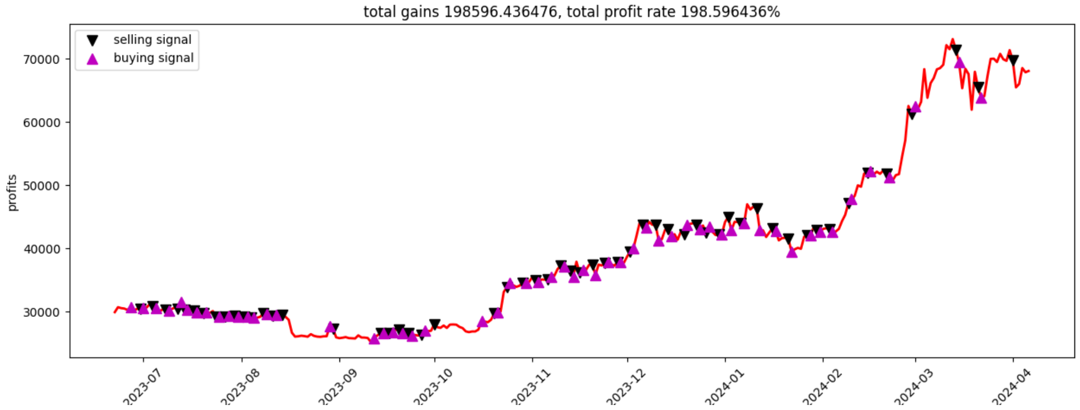


Figure 4.10: DQN BackTest Trading Points

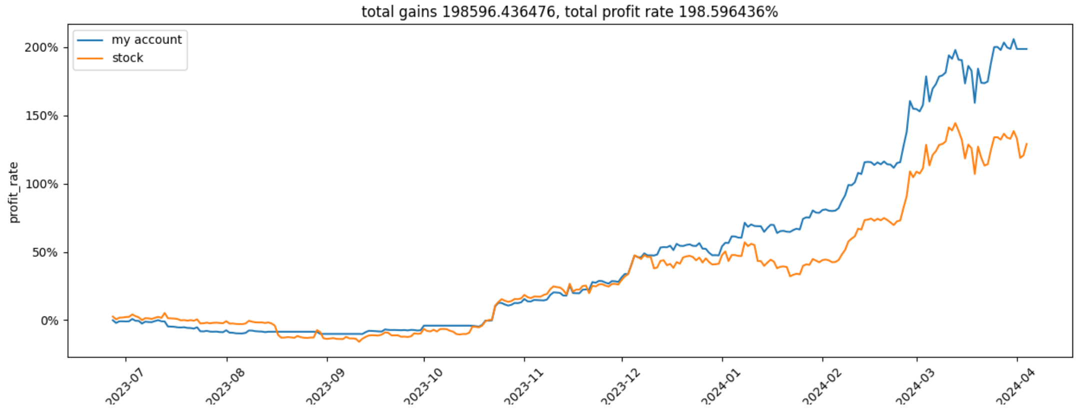


Figure 4.11: DQN BackTest Trading Profit Rate

The results of the test were impressive, despite the fact that due to the 121% growth rate in the Bitcoin market, the PPO agent generated a profit margin of more than double the growth rate, and the DQN agent demonstrated a profit margin well ahead of the market. The intuitive numerical results of the trading can be found in Table 4.2.

Table 4.2: Trading Results

	Start Value	**End Value**	**Rate of Change**
BTC-USD Trends	30688.164063	67837.640625	121.054738%
PPO Agent	100000.00000	386196.62712662574	286.196627%
DQN Agent	100000.00000	298596.4364755854	198.596436%

Next, the agents will be compared to benchmark trading strategies, it will first show the buy and sell points for the 4 benchmark strategies.

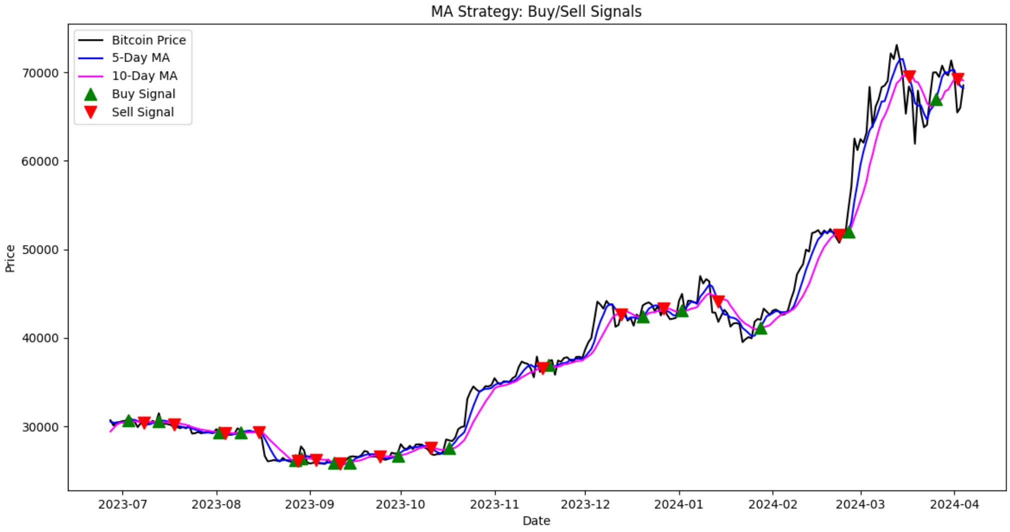


Figure 4.12: MA Strategy Buy and Sell Points

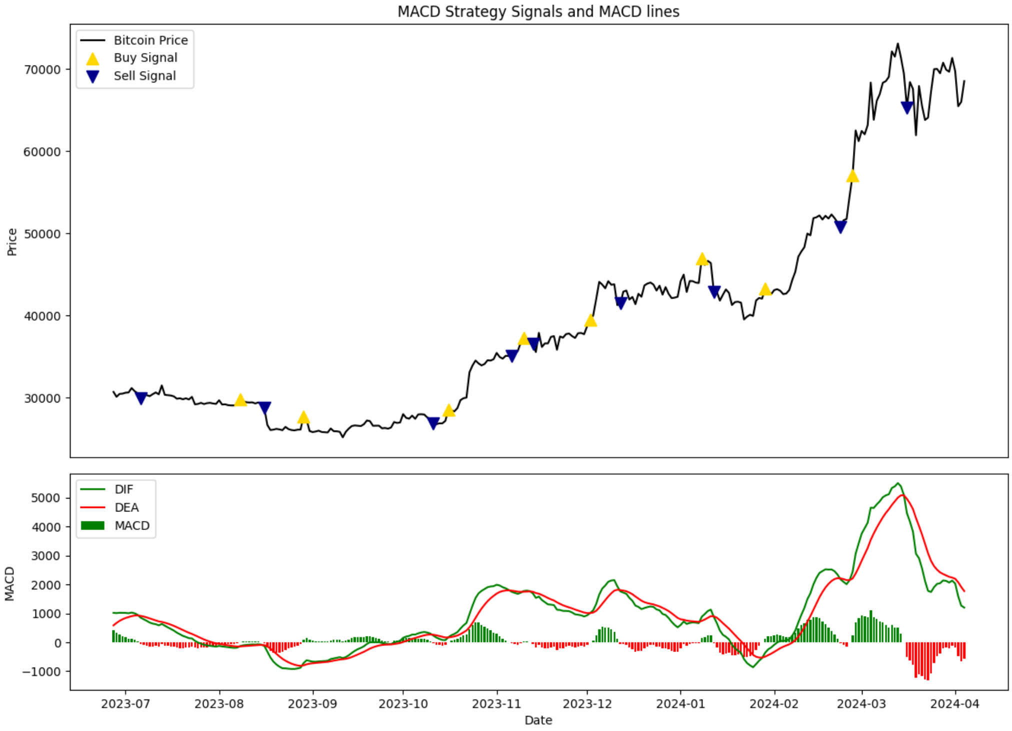


Figure 4.13: MACD Strategy Buy and Sell Points

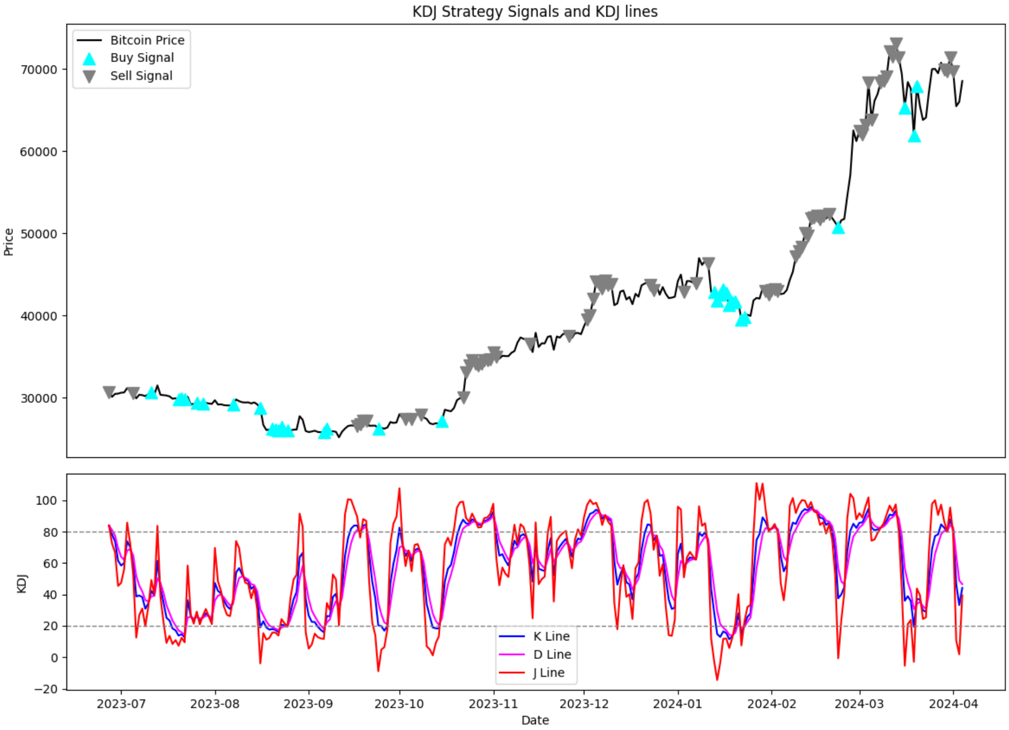


Figure 4.14: KDJ Strategy Buy and Sell Points

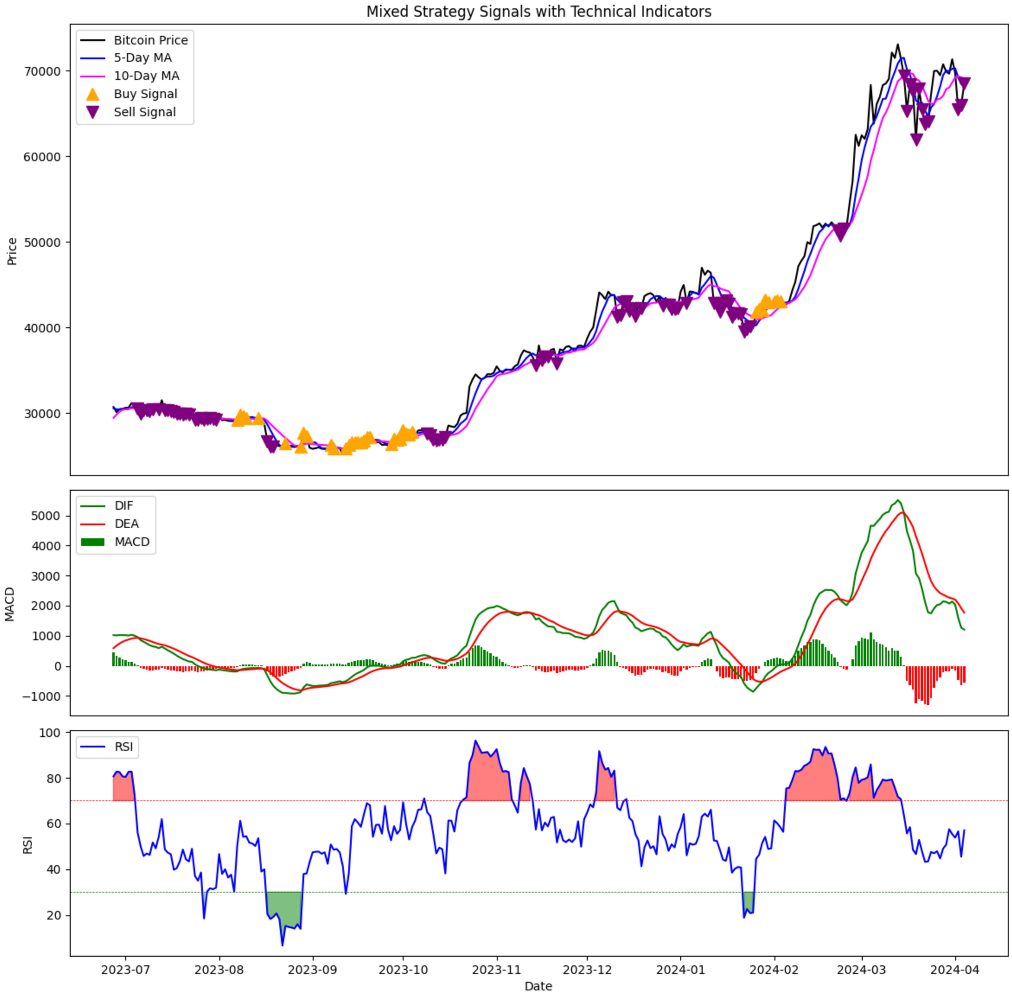


Figure 4.15: Mixed Strategy Buy and Sell Points

From the figure of these four buy and sell signals, it can be learnt that the selected technical indicator strategy is able to roughly determine the trend, buy at lows and sell at highs. Next, the actual results of these strategies in simulated trading and the comparison with AI agents are shown.

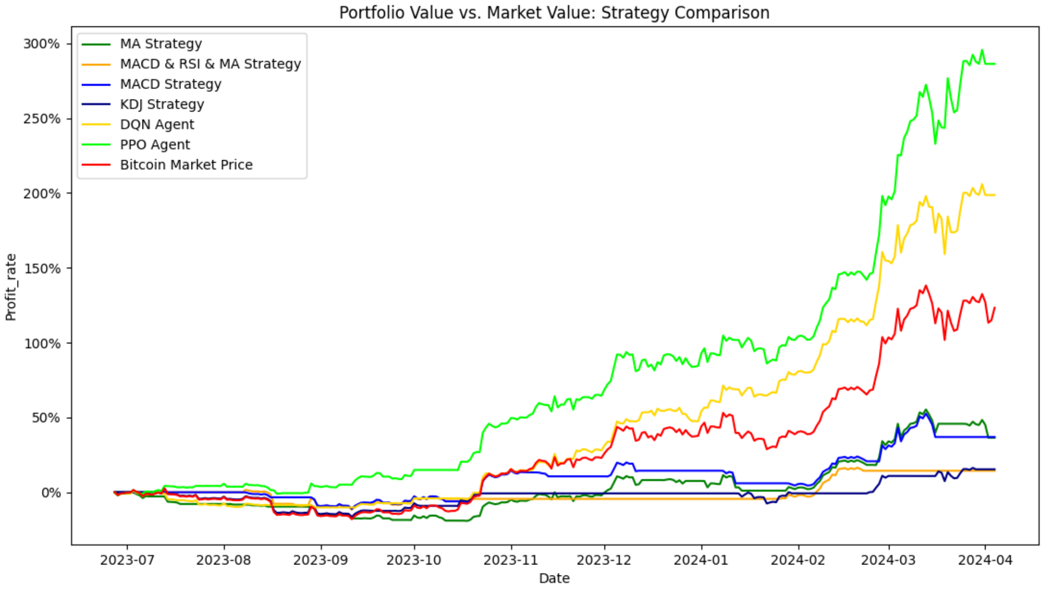


Figure 4.16: Portfolio Value vs Market Value

Table 4.3: Trading Results Comparison

	Start Value	**End Value**	**Rate of Change**
BTC-USD Trends	30688.164063	67837.640625	121.054738%
PPO Agent	100000.00000	386196.62712662574	286.196627%
DQN Agent	100000.00000	298596.4364755854	198.596436%
MACD Strategy	100000.00000	136899.8601342281	36.899860%
MA Strategy	100000.00000	136412.30683387117	36.412307%
KDJ Strategy	100000.00000	136899.8601342281	15.344789%
Mixed Strategy	100000.00000	114368.40738970505	14.368407%

It is easy to conclude from the figure 4.16 that the AI agent's return on investment is much higher than other benchmark strategies and Bitcoin trends. The visual numerical results of trading returns are shown in the Table 4.3. Such results are encouraging, with the PPO agent outperforming the DQN, but both possessing excellent performance in the test set, which demonstrates the viability of deep reinforcement learning in the realm of bitcoin trading. At the same time, it can also be found that while technical indicators can capture trends to a certain extent, a simple strategy that relies only on technical indicators will not lead to good performance, and technical indicators are more of an adjunct to human traders than they can be used directly for trading.

Chapter 5

Summary and Reflections

5.1 Project Management

The original planned and actual timeline for the project is shown below in Figure 5.1, Figure 5.2, Figure 5.3 and Figure 5.4. At the beginning of the project, the most important things were to acquire enough data, learn the basics related to trading, and conduct an exhaustive literature review. Acquiring data was much easier than planned, as accurate data was easily accessible due to the free and hot cryptocurrency market. With some basic knowledge, the learning of various technical indicators and trading rules progressed quickly. But the literature review took longer than expected because I had no previous exposure to reinforcement learning and deep learning, not to mention no hands-on practice on my own, and it took longer to understand these. Also python was an unfamiliar language to me, and it took me a while to learn how to configure and use it, as well as to learn about the common reinforcement learning environment gym and the reinforcement learning library pytorch. All of this led to a delay in the timeline.

Unlike planned, I did not implement the benchmarking strategy at the beginning of the project because machine learning was more difficult than I expected, so I instead focused on model design and training first. At the very beginning of designing the model, I struggled a lot with designing the trading environment due to my lack of relevant experience. The initial preconception was to directly use the existing FinRL [30] library, which is a packaged library capable of direct deep reinforcement learning training for stocks. As it is an external library, it is very inconvenient to make changes to the internal environment, and I have trained agents using this library several times, but the results are generally poor. I spent a lot of time modifying the environment of this library as well as debugging the agents, but since this library was originally designed to perform trading on US stocks, many of the details were different and created a lot of problems. And since it was a packaged project, I had a hard time being free to design a custom environment for Bitcoin.

So in the spring semester, I started designing and implementing my own trading environment to customise the way agents learn based on the Bitcoin market, which was followed by my unique design of the observation space and reward function described in the methodology section. The environment architecture was inspired by a Github project [47]. Early on in this phase I used a custom DQN algorithm, and a simple two-layer fully connected neural network, which continued to perform poorly. Moreover, adding more metrics to the observation space caused the Q-table in the DQN algorithm to become too large, making training slow. So I started to shift to focus mainly on the PPO algorithm, which is a better solution for the scenario of this project. I changed the original backend custom DQN algorithm to a stable PPO and DQN algorithm from the stable_baseline3 library for training. After the redesign of the reward function and the tuning of the hyperparameters, the performance of the model was optimised, but it still fell short of my expectations.

While doing this, I implemented a number of different baseline strategies for comparison, which took about a week. To further improve the performance of the agent, I switched the network used to extract features before reinforcement learning to LSTM, due to the usefulness of LSTM for time-series feature extraction. As I was unfamiliar with the data input format and implementation of the LSTM model, I spent a week tweaking these, modifying the format of the data returned by the get_state() function in the environment, as well as further modifying the types of features in the observation space. The results of this training were excellent and showed that my constant modifications to the environment design were fruitful. The results of the project were not yet at a level that would allow a conference paper to be published, so this stage was deleted. Due to time constraints, the project outcomes did not allow time for conference paper writing, so this phase of conference paper writing was removed. The real-time trading system involves more complex design, including real-time data acquisition and analysis, model porting, etc., which takes more time, so it will be done after this semester.

Task List:

A. Initial Proposal and Ethics Checklist Compilation

B. Exhaustive Literature Review

C. Data Acquisition and Pre-processing

D. Model Design and Validation

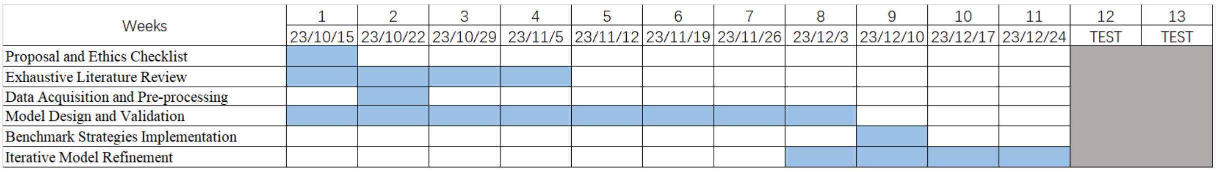
E. Benchmark Strategies Implementation

F. Iterative Model Refinement

G. In-depth Performance Evaluation

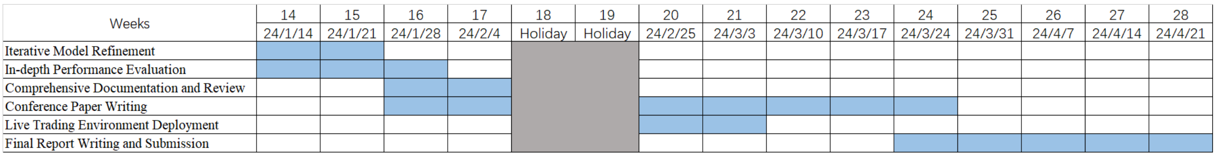
H. Comprehensive Documentation and Review

I. Final Report Writing



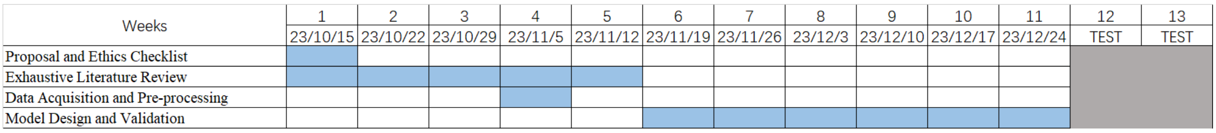
Weeks	1	2	3	4	5	6	7	8	9	10	11	12	13
	23/10/15	23/10/22	23/10/29	23/11/5	23/11/12	23/11/19	23/11/26	23/12/3	23/12/10	23/12/17	23/12/24	TEST	TEST
Proposal and Ethics Checklist	🟦											▨	▨
Exhaustive Literature Review	🟦	🟦	🟦	🟦								▨	▨
Data Acquisition and Pre-processing		🟦										▨	▨
Model Design and Validation	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦				▨	▨
Benchmark Strategies Implementation									🟦			▨	▨
Iterative Model Refinement								🟦	🟦	🟦	🟦	▨	▨

Figure 5.1: Original First semester plan



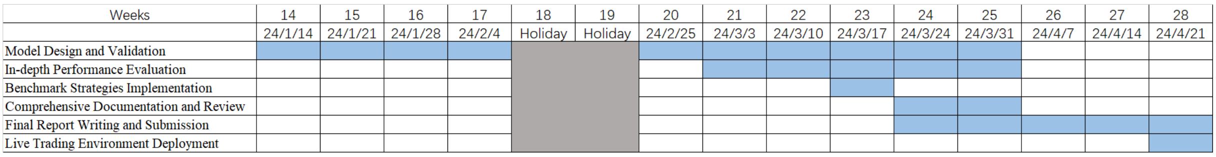
Weeks	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
	24/1/14	24/1/21	24/1/28	24/2/4	Holiday	Holiday	24/2/25	24/3/3	24/3/10	24/3/17	24/3/24	24/3/31	24/4/7	24/4/14	24/4/21
Iterative Model Refinement	🟦	🟦			▨	▨									
In-depth Performance Evaluation	🟦	🟦	🟦		▨	▨									
Comprehensive Documentation and Review			🟦	🟦	▨	▨									
Conference Paper Writing			🟦	🟦	▨	▨	🟦	🟦	🟦	🟦	🟦				
Live Trading Environment Deployment					▨	▨	🟦	🟦							
Final Report Writing and Submission					▨	▨					🟦	🟦	🟦	🟦	🟦

Figure 5.2: Original Second semester plan



Weeks	1	2	3	4	5	6	7	8	9	10	11	12	13
	23/10/15	23/10/22	23/10/29	23/11/5	23/11/12	23/11/19	23/11/26	23/12/3	23/12/10	23/12/17	23/12/24	TEST	TEST
Proposal and Ethics Checklist	🟦											▨	▨
Exhaustive Literature Review	🟦	🟦	🟦	🟦	🟦							▨	▨
Data Acquisition and Pre-processing				🟦								▨	▨
Model Design and Validation						🟦	🟦	🟦	🟦	🟦	🟦	▨	▨

Figure 5.3: Actual First semester plan



Weeks	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
	24/1/14	24/1/21	24/1/28	24/2/4	Holiday	Holiday	24/2/25	24/3/3	24/3/10	24/3/17	24/3/24	24/3/31	24/4/7	24/4/14	24/4/21
Model Design and Validation	🟦	🟦	🟦	🟦	▨	▨	🟦	🟦	🟦	🟦	🟦	🟦			
In-depth Performance Evaluation					▨	▨		🟦	🟦	🟦	🟦	🟦			
Benchmark Strategies Implementation					▨	▨				🟦					
Comprehensive Documentation and Review					▨	▨					🟦	🟦			
Final Report Writing and Submission					▨	▨					🟦	🟦	🟦	🟦	🟦
Live Trading Environment Deployment					▨	▨									🟦

Figure 5.4: Actual Second semester plan

5.2 Reflections

Through reflecting on this complex and demanding project, I have gained valuable insights into both the technical and managerial aspects of undertaking a major piece of research. Here, I will delve into my reflections and categorise them into four key areas: time management, risk management, learning outcomes and the benefits of participating in a final year project.

5.2.1 Time Management

My experience highlights the fact that effective time management is critical to the successful completion of a project. Initially, I underestimated the time required for certain tasks, particularly the literature review and the learning curve associated with new technologies and methods. This misjudgement led to delays in the early stages of the project that could have been mitigated with a more realistic timeline and perhaps a buffer for unforeseen challenges. The lesson here is clear: future projects would benefit from more conservative time estimates and the incorporation of flexible time periods to accommodate unexpected delays or learning requirements. Additionally, project plans need to be updated in real time, and this experience taught me the value of an iterative planning and review process, with plans updated regularly to reflect current understanding and progress.

In addition, the project emphasised the importance of setting realistic goals and expectations for yourself. Balancing ambition with practicality in the project timeline was crucial. In future projects, I aim to apply these lessons learnt by adopting more rigorous time management strategies to better anticipate and minimise delays.

5.2.2 Risk Management

The project also served as a practical lesson in risk management. The early challenges of data capture went better than expected, but the time saved was consumed elsewhere. My struggles with the FinRL library and subsequent decision to design my own trading environment emphasised the risks associated with relying on external libraries without a thorough review. But I made the right choice in not continuing to experiment with external libraries, wasting time but controlling the risk and avoiding a situation where I might not be able to produce satisfactory results until the end of the semester. Looking ahead, I learnt the importance of early and comprehensive risk identification, including technology dependencies, knowledge gaps and project scope. Implementing a risk management strategy that includes mitigation plans and contingency plans will be critical for future projects.

5.2.3 Learning Outcomes

The technical and conceptual learning outcomes of the project were far-reaching. Immersing myself in the complexities of reinforcement learning, deep learning, and financial trading was overwhelming at first, but ultimately rewarding. This project facilitated significant growth in my understanding of machine learning principles and their application to real-world problems.

From a technical perspective, this project has been a steep learning curve that has greatly enriched my knowledge. Delving into the world of reinforcement learning, deep learning, and financial market trading was initially intimidating, but it proved to be extremely rewarding. Learning to code in Python and familiarising myself with libraries such as gym and PyTorch not only expanded my skill set, but opened up new avenues for exploring machine learning and artificial intelligence. The hands-on experience of designing and refining trading environments and models has provided me with practical insights rarely found in textbooks.

In addition to gaining technical skills, the project helped develop my critical thinking and problem solving skills. Designing custom trading environments and iterating on model designs requires deep analytical thinking and creativity. The process involved not only the application of theoretical knowledge, but also innovation and experimentation to overcome practical challenges. The learning journey also emphasises the importance of resilience and perseverance. Facing setbacks, such as poor initial model performance or environment design challenges, tested my resolve but ultimately strengthened my determination and resilience.

5.2.4 Benefits of the Final Year Project

Being involved in the final year project has been a very rewarding endeavour. Not only did it allow me to apply my academic knowledge to real-world situations, but it also helped me to develop key soft skills such as project management, problem solving and self-directed learning. In addition, the programme enhanced my ability to synthesise information from a variety of sources, design experiments and adapt to feedback, all of which are invaluable skills in both academic and professional environments.The FYP also allowed me to explore some career possibilities in this area of quantitative finance, helping me to develop an interest in this area.

The finalised agent harvested unexpected results and I will try to explore more possibilities by applying it practically to trading after the project is over. In conclusion, although the project presented many challenges, the lessons learnt and skills gained far outweighed the difficulties faced. The experience has been profoundly transformative, not only in terms of technical knowledge and skills, but also in terms of personal growth and understanding of the research process.

5.3 Conclusion and Future Work

This thesis presents a methodology for the design of a deep reinforcement learning environment that trains agents capable of making a profit in the Bitcoin market. More trading pairs will be introduced (e.g. ETH-USD) and more algorithms will be tried to test performance. The trading agents will eventually be put into a real-time trading environment for execution.

References

[1] Cryptocurrency market size, share & trends analysis report by component, by hardware, by software, by process (mining, transaction), by type, by end-use, by region, and segment forecasts, 2023 - 2030, 2023. Report ID: GVR-4-68039-979-9.

[2] Cross-industry standard process for data mining - Wikipedia. https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining, 2024. Accessed: 2024-04-06.

[3] Irene Aldridge. *High-Frequency Trading: A Practical Guide to Algorithmic Strategies and Trading Systems*. Wiley, 2 edition, 2013. Chapter 3: ”The Basics of High-Frequency Trading”, pp. 45-67.

[4] Bo An, Shuo Sun, and Rundong Wang. Deep reinforcement learning for quantitative trading: Challenges and opportunities. *IEEE Intelligent Systems*, 37(2):23–26, 2022.

[5] Thirunavukarasu Anbalagan and S. Uma Maheswari. Classification and prediction of stock market index based on fuzzy metagraph. *Procedia Computer Science*, 47:214–221, 2015. Graph Algorithms, High Performance Implementations and Its Applications (ICGHIA 2014).

[6] Jamil Baz, Nicolas Granger, Campbell R Harvey, Nicolas Le Roux, and Sandy Rattray. Dissecting investment strategies in the cross section and time series, 2015. Available at SSRN 2695101.

[7] Binance. Bitcoin price. https://www.binance.com/en/price/bitcoin, 2024. Accessed: 2024-04-05.

[8] Binance. Fees & transactions overview. https://www.binance.com/en/fee/trading, 2024. Accessed: 2024-04-09.

[9] Binance Academy. A guide to cryptocurrency fundamental analysis. https://acad

emy.binance.com/en/articles/a-guide-to-cryptocurrency-fundamental-analysis, 2023. Accessed: 2023-11-28.

[10] Binance Academy. What is fundamental analysis (fa)? https://academy.binance.com/en/articles/what-is-fundamental-analysis-fa, 2023. Accessed: 2023-11-28.

[11] Dragos Bozdog. *Algorithmic Trading with Machine Learning*. Springer, 2021. Chapter 4: ”Machine Learning for Algorithmic Trading”, pp. 103-126.

[12] CEIC Data. Market capitalization - china. https://www.ceicdata.com/en/indicator/china/market-capitalization, 2024. Accessed: 2024-04-05.

[13] Rajashree Dash and Pradipta Kishore Dash. A hybrid stock trading framework integrating technical analysis with machine learning techniques. *The Journal of Finance and Data Science*, 2(1):42–57, 2016.

[14] Rati Devidze, Goran Radanovic, Parameswaran Kamalaruban, and Adish Singla. Explicable reward design for reinforcement learning agents. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 20118–20131. Curran Associates, Inc., 2021.

[15] Baoyu Ding, Ling Li, Yunliang Zhu, Hui Liu, Junfeng Bao, and Zezhu Yang. Research on comprehensive analysis method of stock kdj index based on k-means clustering. 01 2019.

[16] Fan Fang, Carmine Ventre, Michail Basios, Leslie Kanthan, David Martinez-Rego, Fan Wu, and Lingbo Li. Cryptocurrency trading: a comprehensive survey. *Financial Innovation*, 8:13, 2022.

[17] Rodolfo Toríbio Farias Nazário, Jéssica Lima e Silva, Vinicius Amorim Sobreiro, and Herbert Kimura. A literature review of technical analysis on stock markets. *The Quarterly Review of Economics and Finance*, 66:115–126, 2017.

[18] Yahoo Finance. Btc-usd historical data. https://finance.yahoo.com/quote/BTC-USD/history, 2024. Accessed: 2024-04-08.

[19] Thomas G. Fischer. Reinforcement learning in financial markets - a survey. FAU Discussion Papers in Economics 12/2018, Friedrich-Alexander University Erlangen-Nuremberg, Institute for Economics, 2018.

[20] Forbes. Cryptocurrency prices. `https://www.forbes.com/digital-assets/crypto-prices/`, 2024. Accessed: 2024-04-05.

[21] Dong Han, Beni Mulyana, Vladimir Stankovic, and Samuel Cheng. A survey on deep reinforcement learning algorithms for robotic manipulation. *Sensors*, 23(7), 2023.

[22] J Stuart Hunter. The exponentially weighted moving average. *Journal of quality technology*, 18(4):203–210, 1986.

[23] Investopedia. Moving average (ma): Purpose, uses, formula, and examples. `https://www.investopedia.com/terms/m/movingaverage.asp`, 2023. Accessed: 2024-04-04.

[24] Investopedia. Moving average convergence divergence - macd. `https://www.investopedia.com/terms/m/macd.asp`, 2024. Accessed: 2024-04-03.

[25] Investopedia. Relative strength index (rsi) indicator explained with formula. `https://www.investopedia.com/terms/r/rsi.asp`, 2024. Accessed: 2024-04-04.

[26] Liu Jing and Yuncheol Kang. Automated cryptocurrency trading approach using ensemble deep reinforcement learning: Learn to understand candlesticks. *Expert Systems with Applications*, 237:121373, 2024.

[27] Yuxi Li. Deep reinforcement learning, 2018.

[28] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019.

[29] Xiao-Yang Liu, Zhuoran Xiong, Shan Zhong, Hongyang Yang, and Anwar Walid. Practical deep reinforcement learning approach for stock trading, 2022.

[30] Xiao-Yang Liu, Hongyang Yang, Jiechao Gao, and Christina Dan Wang. FinRL: Deep reinforcement learning framework to automate trading in quantitative finance. *ACM International Conference on AI in Finance (ICAIF)*, 2021.

[31] Dimitri Mahayana, Elbert Shan, and Muhammad Fadhl'Abbas. Deep reinforcement learning to automate cryptocurrency trading. In *2022 12th International Conference on System Engineering and Technology (ICSET)*, pages 36–41, 2022.

[32] Mansoor Maitah, Petr Prochazka, Michal Cermak, and Karel Srédl. Commodity channel index: Evaluation of trading rule of agricultural commodities. *International Journal of Economics and Financial Issues*, 6(1):176–178, 2016.

[33] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[34] Andrew S. Morgan, Daljeet Nandha, Georgia Chalvatzaki, Carlo D'Eramo, Aaron M. Dollar, and Jan Peters. Model predictive actor-critic: Accelerating robot skill acquisition with deep reinforcement learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2021.

[35] Raúl Navas, Ana Paula Matias Gama, and Sónia R. Bentes. Can fundamental analysis provide relevant information for understanding the underlying value of a company? In Vito Bobek, editor, *Trade and Global Market*, chapter 10. IntechOpen, Rijeka, 2018.

[36] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, page 278–287, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[37] Aisha Peng, Sau Loong Ang, and Chia Yean Lim. Automated cryptocurrency trading bot implementing drl. *Pertanika Journal of Tropical Agricultural Science*, 30(4), 2022.

[38] Matteo Prata, Giuseppe Masi, Leonardo Berti, Viviana Arrigoni, Andrea Coletta, Irene Cannistraci, Svitlana Vyetrenko, Paola Velardi, and Novella Bartolini. Lob-based deep learning models for stock price trend prediction: A benchmark study. Department of Computer Science, Sapienza University of Rome, Italy and J.P. Morgan AI Research, New York, USA., 2023.

[39] Tidor-Vlad Pricope. Deep reinforcement learning in quantitative algorithmic trading: A review, 2021.

[40] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[41] Christoph Schröer, Felix Kruse, and Jorge Marx Gómez. A systematic literature review on applying crisp-dm process model. *Procedia Computer Science*, 181:526–534, 01 2021.

[42] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[43] Adarsh Sehgal, Muskan Sehgal, and Hung Manh La. Aacher: Assorted actor-critic deep reinforcement learning with hindsight experience replay, 2022.

[44] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. *31st International Conference on Machine Learning, ICML 2014*, 1, 06 2014.

[45] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[46] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015.

[47] wbbhcb. Stock Market Simulation and Analysis Tool. `https://github.com/wbbhcb/stock_market`, 2023. Accessed: 2024-04-10.

[48] Wikipedia. Stochastic oscillator, 2024. Accessed: [2024-04-03].

[49] J Welles Wilder. *New concepts in technical trading systems*. Trend Research, 1978.

[50] Xing Wu, Haolei Chen, Jianjia Wang, Luigi Troiano, Vincenzo Loia, and Hamido Fujita. Adaptive stock trading strategies with deep reinforcement learning methods. *Information Sciences*, 538:142–158, 2020.

[51] Zhimin (Jimmy) Yu. Cross-section of returns, predictors credibility, and method issues. *Journal of Risk and Financial Management*, 16(1), 2023.

[52] Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep reinforcement learning for trading, 2019.