الجمهورية الشعبية الديمقراطية الجزائرية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
المدرسة العليا للإعلام الآلي ٠80 ماي 5491٠ بسيدي بلعباس
École Supérieure en Informatique
-08 Mai 1945- Sidi Bel Abbès

THESIS

To obtain the diploma of **Engineer**
Field: **Computer Science**
Specialty: **Ingénierie des Systèmes Informatiques (ISI)**

Theme

Trading bot based on Reinforcement Learning

Presented by: BENAHMED Djawed

Submission Date: **June, 2022**
In front of the jury composed of:

Mr. CHAIB Souleymane	President
Mr. KHALDI Belkacem	Supervisor
Ms. DIF Nassima	Examiner

Academic Year : 2021/2022

Acknowledgments

God, the Almighty, is to be praised and thanked first and foremost for showering me with blessings during my research to enable me to accomplish it successfully.

Words cannot express how grateful I am to my parents for their love, prayers, care, and sacrifices in educating and preparing me for my future; the level of support we had and continue to receive is indescribable.

I would like to express my deep and sincere gratitude to my research supervisor and teacher, **Mr. KHALDI Belkacem** for providing valuable guidance and advice throughout this research.

I cannot thank **Mr. ALAOUI MDAGHRI Abdellah** of Swissdigilab enough for providing me the opportunity to work on this project, from which I have acquired a huge amount of fresh new information. I also would like to thank him for his guidance and continuous support.

A special thanks to the entire working team of my school, including the School director **Dr. Benslimane Sidi Mohammed** and **Dr. Amar Bensaber Djamel**, as well as all teachers, without whom I could not have accomplished my goals. Finally, a big thank you to our friends and colleagues for their assistance along this trip.

BENAHMED Djawed.

Abstract

Artificial intelligence has been increasingly popular in financial sectors such as stock and currency trading in recent years. Reinforcement learning is one of AI widely used branches in financial markets problems.

In this thesis, we'll try to describe the problem in detail and talk about the project's goals and requirements. Furthermore, we will explain the procedures and steps we took and the techniques that were selected to implement this project.

Keywords

Crypto Market, Currency trading, Automated Trading, Reinforcement Learning, Deep Learning, Advantage Actor Critic (A2C), Proximal Policy Optimization (PPO).

Résumé

L'intelligence artificielle est de plus en plus populaire dans les secteurs financiers tels que le commerce des actions et des devises ces dernières années. L'apprentissage par renforcement est l'une des branches de l'IA largement utilisées dans les problèmes de marché financier.

Dans cette thèse, nous essaierons de décrire le problème en détail et de parler des objectifs et des exigences du projet. De plus, nous expliquerons les procédures et les étapes que nous avons suivies et les techniques qui ont été sélectionnées pour mettre en œuvre ce projet.

Keywords

Marché de la cryptographie, trading de devises, trading automatisé, apprentissage par renforcement, apprentissage en profondeur, critique d'acteur d'avantage (A2C), optimisation de la politique proximale (PPO).

الملخص

أصبح الذكاء الاصطناعي شائعًا بشكل متزايد في القطاعات المالية مثل تداول الأسهم والعملات في السنوات الأخيرة. التعلم المعزز هو أحد فروع الذكاء الاصطناعي المستخدمة على نطاق واسع في مشاكل السوق المالية.

سنحاول في هذه الرسالة وصف المشكلة بالتفصيل والتحدث عن أهداف المشروع ومتطلباته. علاوة على ذلك ، سنشرح الإجراءات والخطوات التي اتخذناها والتقنيات التي تم اختيارها لتنفيذ هذا المشروع.

Keywords

سوق العملات المشفرة ، تداول العملات ، التداول الآلي ، التعلم المعزز ، التعلم العميق ، ناقد الممثل المتميز (A2C) ، تحسين السياسة القريبة .(PPO)

Contents

List of Figures

List of Tables

List of Acronymes

A2C Advantage Actor Critic. 6, 9, 10, 63–69

ADX Average Directional Index. 20

AI Artificial intelligence. 6, 12, 13, 25, 26, 29, 39

BH Buy and Hold. 57, 60–62, 64, 65, 67

CCI Commodity Channel Index. 20

DDPG Deep Deterministic Policy Gradient. 37

DNN Deep Neural Network. 19

DPG Deterministic Policy Gradient. 36

DQN Deep Q-Network. 36

DRL Deep Reinforcement Learning. 13, 35

MACD Moving Average Convergence Divergence. 8, 18–20

OBV On-Balance Volume. 20

OHLCV Open, Hight, Low, Close, Volume. 19

PG Policy Gradient. 36

PPO Proximal Policy Optimization. 6, 9, 10, 37, 59–62, 65, 67–69

RL Reinforcement Learning. 6, 13, 28, 39, 70

RSI Relative Strength Index. 19

General Introduction

Artificial Intelligence, like many other technical achievements, emerged from the pages of fairy tales and science fiction stories. People fantasized about machines that might fix issues, speak and make decisions. Fast forward to the 21 century, Artificial intelligence has become part of our daily lives, from phone voice assistants to smart homes and self-driving cars, Artificial intelligence (AI) solutions are all over the place.

The financial industry is one of the first adopters of artificial intelligence, from banking fraud detection to automated investment and trading. Anthony Antenucci, vice president of global business development at Intelenet Global Services, recently said "Machine learning is evolving at an even quicker pace, and financial institutions are one of the first adaptors,"

Thanks to artificial intelligence, trading nowadays is brought to a whole new level – more professional and advanced strategies are applied easily and comfortably even by beginners. "Artificial intelligence is to trading, what fire was to the cavemen." That's how one industry player described the impact of disruptive technology on a staid industry. In other (less creative) words, AI is a game-changer for the financial markets [52].

0.1 Swissdigilab

Swissdigilab is a private company based in Switzerland that specializes in the design, development, and maintain applications, frameworks, and other software components for businesses and consumers.

Swissdigilab proposed the project of an AI-based cryptocurrency trading platform, which is separated into two sub-projects, one of which is "Trading bot using reinforcement learning".

0.2 Problem and solution

Financial markets, such as stocks, fiat money, cryptocurrencies, and even physical assets such as gold and silver, are notoriously complicated and chaotic, with highly intricate systems that often make prediction very difficult for a human trader. A quote by William Gallaher "When you think the market can't possibly go any higher (or lower), it almost invariably will, and whenever you think the market "must" go in one direction, nine times out of ten it will go in the opposite direction. Be forever skeptical of thinking that you know what the market is going to do". The complexity of the financial markets created a need for a sophisticated intelligent solution.

Stiff trading strategies designed by experts in the field are one of the intelligent solutions, but these strategies often fail to achieve profitable returns in all market conditions due to their incapability of adapting to new trends and changes in the markets [42]. Hence, Machine learning methods such as RL/DRL are used to address these problems.

While humans remain an important element of the trading equation, AI is becoming increasingly important, and human interventions becoming less and less needed, to the point where the whole trading process can be automated. Which is the goal of this project, exploring different approaches and methods to automate the trading process using AI Reinforcement Learning.

0.3 Objectives

The project's major objective is to experiment with trading strategies based on Reinforcement Learning in the cryptocurrency market by using numerous currency pairings to test different reinforcement learning algorithms and methodologies identified in the literature, to investigate the effects of applying AI and Reinforcement Learning to a market with such volatility.

The algorithm should be efficient in terms of execution time, taking into account height slippage and the volatility of the cryptocurrency market, which might result in different prices from the time an order is placed and the time it is executed.

The testing conditions should be as realistic as possible, taking into consideration the transaction costs, the slippage factor, and any other small aspects that might have a long-term impact on the trading strategy.

The test results obtained by the models should be compared and analyzed to determine the advantages and disadvantages of each approach and the final objective is to provide a visualization tool to easily test and evaluate the performance of each model.

0.4 Limitations

- The Cryptocurrency market is highly influenced by the media and news, for example, simple twitter posts from influential people such as Elon Musk can boost or crash some currencies in a matter of days and these events can't be predicted from only price movements and patterns in data since they don't happen often and they are the exception not rules.
- The Crypto currency market goes through phases due to the circumstances in the world, for instance, the prices between 2017 and 2018 vary from the prices between 2019 and 2021 because of the onset of the covid pandemic in late 2019, so the period of training and testing chose is crucial.
- Deep Reinforcement Learning is known to be one of the most demanding algorithms in terms of computing power and training time. Training a single model might take days for even a small environment since some algorithms train two neural networks at the same time (policy and value function networks) and collect data on the fly from the environment. This means that while the model is being trained, a separate process is gathering data from the environment, which adds processing effort. Rl algorithms are also sensitive to weight initialization which makes them unstable and multiple runs are required since some runs never converge due to high or low values in model weights.
- Most exchanges, if not all, do not offer a sandbox account for testing bot integration; therefore, it is impossible to build a bot that is functionally integrated with an exchange without testing on a real account with real money.

0.5 Plan of the thesis

- **Chapter 1** This chapter will examine the world of trading, introducing fundamental ideas such as the distinction between trading and investing, types of trading, and describing various financial markets.
- **Chapter 2** In this chapter, we will give the information necessary for the reader to comprehend several fundamental AI and machine learning subjects.
- **Chapter 3** In this chapter, we'll discuss the design of experiments and the methods involved in conducting training and testing.
- **Chapter 4** In this chapter, we will give the test results (statistics and graphics) as well as the conclusions taken from them.
- **Chapter 5** This chapter will describe the backtesting platform, including its architecture and underlying technology.
- **Chapter 6** This chapter focuses on the project management technique, collaborative work, and technologies utilized to facilitate a smoother execution of this project.

CHAPTER 1

Trading

Trading is the process of purchasing and selling different assets or financial instruments (such as stocks, currencies, commodities, companies, etc) that are listed on the market. The idea is to maximize the capital's return on investment by using recurring buy and sell orders to take advantage of market volatility. Profit is generated when one buys at a lower price than it sells afterward. Trading is a short-term and volatile activity as compared to long-term transactions such as mutual fund or bond deals. In a nutshell, the fundamental rule of trading is to purchase when the price is low and sell when the price is high to make profits.

1.1 Trading VS Investing

There is often a misunderstanding between the concepts of investing and trading, and this must be clarified. When it comes to investing, an investor is someone who stays on to their position or security for a longer amount of time and is a long-term participant, while a trader is someone who is impacted by the ups and downs of the market's securities. Investing is a long-term method in which the objective is to accumulate wealth gradually over time via the use of investment schemes such as mutual funds, and the purchase and sale of a portfolio of stocks, bonds, and other securities. When compared to trading, investment is kept for years or even decades, and it comes with a variety of benefits such as interest, dividends, stock splits, and many more [37]. The difference between trading and investing is summarized in the table 1.1.

Investing	Trading
Long period (Years, decades, or even longer periods)	Short period (Depending on the type of trading, it might be for a week, a day, or hours)
Art of creating wealth (accumulate wealth gradually over time)	Skill of timing the market (buying when the price is lowest and selling when the price is the highest)
Comparatively lower risk and lower returns in the short run but might deliver higher returns in the long run	Involves a high risk-reward ratio (since the price might swing either way, high or low, in a short time)
Profit from stability and long-term predictions of the markets	Profits from volatile trends in the market

Table 1.1: Difference between Trading and Investing

1.2 Types of trading

There are different types of trading that we are going to mention below [1]

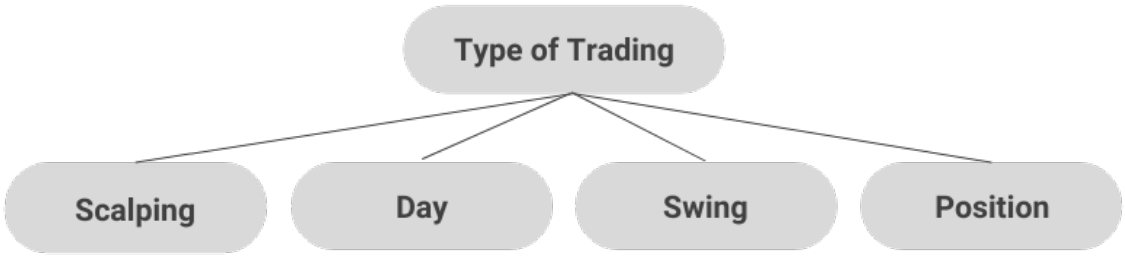


Figure 1.1: Types of trading

1.2.1 Scalping

Scalping is the most short-term form of trading. Scalp traders only hold positions open for seconds or minutes at most. These short-lived trades target small intraday price movements. The purpose is to make lots of quick trades with smaller profit gains, but let profits accumulate throughout the day due to the sheer number of trades being executed in each trading session [10].

[1] https://www.capitalindex.com/bs/eng/pages/trading-guides/different-types-of-trading-strategies

1.2.2 Day trading

Day traders take positions and exit them on the same day, eliminating the risk of big overnight movements. All trades will be closed with a profit or a loss at the end of the day. Because trades are typically held for minutes or hours, there is enough time to evaluate the markets and monitor positions periodically throughout the day [10].

Day traders pay particularly close attention to fundamental and technical analysis, using technical indicators such as Moving Average Convergence Divergence (MACD), the Relative Strength Index, and the Stochastic Oscillator, to help identify trends and market conditions.

1.2.3 Swing trading

Swing traders usually hold positions for several days, sometimes even weeks. Traders do not need to sit constantly monitoring the charts and their trades throughout the day because positions are held over some time [10].

1.2.4 Position trading

Position traders are interested in long-term price movement, hoping to benefit as much as possible from large price shifts. As a result, trades typically last for weeks, months, or even years. Position traders typically analyze and evaluate markets using weekly and monthly price charts, using technical indicators and fundamental analysis to find suitable entry and exit levels [10].

Scalping	Day	Swing	Position
Make multiple trades per hour	Make multiple trades per day	Make several trades per week	Make fewer trades per months
Positions last from seconds to minutes	Positions last from hours to days	Positions last from days to weeks	Positions last from weeks to months
Full-time job	Full-time job	Part-time job	Part-time job
Uses short-term buy and sell signals	Uses short-term buy and sell signals	Utilizes trends and momentum indicators	Utilizes some investing strategies
Smaller relatively stable gains	Multiple, smaller gains or losses	Fewer, but more substantial gains or losses	Fewer, long-term gains

Table 1.2: Comparison between diffrent type of trading

1.3 Technical analysis and indicators in trading

The trading data is generally represented with a sequence (at regular intervals of time) of open, close, height, low, and volume of the entity traded (stocks, currencies, bonds, etc) and it is called OHLCV representation [55]. Even Deep Neural Network (DNN) find it challenging to grasp this data in this manner due to the high degree of noise and non-stationary nature of the data [42]. Technical indicators were created to describe market behavior and make the patterns easier to interpret. Technical indicators are used by traders to gain insight into the supply and demand of securities and market psychology. Together, these indicators form the basis of technical analysis. Metrics, such as trading volume, provide clues as to whether a price move will continue [50]. The most popular indicators are:

- **Moving Average Convergence Divergence (MACD)** MACD is a trend-following momentum indicator that shows the relationship between two moving averages of a security's price [5].
- **Relative Strength Index (RSI)** The relative strength index (RSI) is a momentum indicator used in technical analysis that measures the

magnitude of recent price changes to evaluate overbought or oversold conditions in the price of a stock or other asset [54]

- **Average Directional Index (ADX)** ADX is used to quantify trend strength. ADX calculations are based on a moving average of price range expansion over a given period of time [23]
- **Commodity Channel Index (CCI)** The CCI was originally developed to spot long-term trend changes but has been adapted by traders for use on all markets or timeframes [32].
- **On-Balance Volume (OBV)** OBV is a technical trading momentum indicator that uses volume flow to predict changes in stock price [53]

There are plenty more indicators, and the encyclopedia [16] book fully describes each one.

Figure 1.2: MACD indicator in binance platform

1.4 Financial markets

Financial markets are a kind of marketplace in which assets such as bonds, shares, and foreign currency are bought and sold (traded). They go under many names, like "Wall Street" and "Capital Market" but all refer to the same thing [18]. financial markets are used by:

- **Businesses** to raise capital for expansion and growth.

- **Investors & Traders** to invest or benefit from market instability to expand the capital.

Several financial markets could be used for trading; we will explore a few of them:

1.4.1 Crypto currency market

Cryptocurrency refers to a kind of digital or virtual money that is protected by cryptography. This kind of protection makes it impossible to counterfeit or double-spend the cash.

Bitcoin is by far the most common and lucrative kind of virtual money. It was developed by an unknown individual who went by the name Satoshi Nakamoto, and it was presented to the public in the form of a white paper in the year 2008 [36]. Since that time, hundreds of other cryptocurrencies have appeared, such as Ethereum, Litecoin, Solana, and many more.

One of the most distinguishing characteristics of cryptocurrencies is the fact that they are not issued by any central authority. Because of this, cryptocurrencies are considered to be resistant to the influence or meddling of governments or any other parties.

What makes cryptocurrencies possible is the revolutionary blockchain technology which was also introduced in Satoshi Nakamoto's white paper [36]. A blockchain may be thought of as an electronic database distributed, and shared among nodes of computers in a network, in crypto-currencies blockchain securely stores information about transactions.

A cryptocurrency exchange, also known as digital currency exchange (DCE), is an online marketplace that gives consumers the option to acquire, sell, and trade cryptocurrencies or digital currencies for other assets, such as traditional fiat money or other digital currencies.

Trading cryptocurrency with fiat currency such as USD can be expensive (height transaction cost) which led to the creation of stable coins. Stablecoins are digital currencies that are designed to keep their value constant regardless of market conditions. Tether (USDT) and Binance Dollar (BUSD) are two examples of cryptocurrencies that have a fixed value of one dollar. Essentially forming a pair of BTC-USDT (Bitcoin and Tether) or BTC-BSDT (Bitcoin and Binance USD) is effectively the same as making a pair of BTC-USD (Bitcoin USD)

1.4.2 Stocks market

Stocks (also called shares or a company's equity), are a financial instrument that shows that you own a piece of a company or business and that you have a percentage of its assets [24]. Stock ownership entails that the shareholder owns a piece of the business in proportion to the number of shares owned. For example, if someone owns 100,000 shares of a company with one million shares, that person or group would own 10% of the company [24].

Why Do Companies Issue Shares? Starting a company (or growing an existing one) needs a large amount of capital to cover the costs of getting the business up and running (expenses such as raw materials, hiring employees, equipment, etc). To get the necessary cash, the business asks for funds from investors, who in turn receive some shares based on the amount invested.

Figure 1.3: Shares holder illustration [11].

The stock market, also known as the stock exchange or bourse in Europe, is the most well-known marketplace. It is a regulated market for the acquisition and selling of securities such as shares and stocks. In simplified terms, the stock market is where shares of publicly traded companies are bought and sold. The stock market is also a measure of the general state of the economy and it allows the discovery of the price of corporate shares [51].

Stock trading is the practice of purchasing and selling shares of a publicly-traded business in an attempt to profit from the variation in their prices. This short-term perspective distinguishes stock traders from more conventional stock market investors, who often invest for the long term [19].

1.4.3 Bonds market

The bond market (sometimes referred to as the debt market or credit market) is a financial market in which participants may either issue new debt (referred to as the primary market) or acquire and sell existing debt instruments (referred to as the secondary market)[2].

The primary market is commonly referred to as the "new issues" market since transactions take place solely between bond issuers and bond purchasers. In short, the primary market results in the production of entirely new debt securities that did not exist before. Securities that have previously been sold in the primary market are then acquired and sold at a later period in the secondary market. Numerous of online brokers provide this kind of bond for trading[3].

1.4.4 Foreign Exchange (Forex)

The forex market enables parties, including banks and individuals, to purchase, sell, and exchange currencies for hedging and speculating reasons. The forex market is the world's biggest and most active financial market, with billions of dollars changing hands daily. The forex market does not have a central location; rather, it is an electronic network of banks, institutions, brokers, and traders [13]. Forex brokers operate as market makers, they are online platforms allowing individuals to execute exchanges easily with simple clicks, most famous forex brokers[4].

Foreign exchange can be as simple as changing (trading) one currency for another at a local bank. For example, one can swap the U.S. dollar for the euro. When trading on the forex market, currencies are listed in pairs, for example, USD/CAD means changing the United States dollar against the Canadian dollar, and USD/JPY means changing the United States dollar against the Japanese yen [12].

1.4.5 Physical Assets

Investment in physical assets means the acquisition of tangible assets, such as metals, jewels, real estate, livestock, and anything else that has a physical

[2] https://en.wikipedia.org/wiki/Bond_market

[3] https://www.schwab.com/
https://www.tdameritrade.com/
https://us.etrade.com/

[4] https://www.ig.com/
https://www.home.saxo/
https://www.cmcmarkets.com/
https://www.forex.com/

form and a recognized value. A physical asset is essentially anything tangible that you own that can be sold or exchanged other than cash. A trader's aim in this market is that the price at which they may sell an item will be higher than the price at which they purchased it initially [25].

CHAPTER 2

Artificial intelligence

Artificial intelligence (AI), the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings[1]. The term is frequently applied to the project of developing systems endowed with the intellectual processes characteristic of humans, such as the ability to reason, discover meaning, generalize, or learn from experience [17]

There are many forms and types of AI, In this thesis, we will mainly focus on one category, Machine learning which in turn can be divided into other categories

Machine learning according to Arthur Samuel (American pioneer in the field of computer gaming and artificial intelligence) is "the field of study that gives computers the ability to learn without being explicitly programmed" another definition by Tom Mitchell (research scientist and data science pioneer) "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"

The majority of machine learning algorithms are exposed to a dataset. There are several ways in which a dataset may be described. In all cases, a dataset is a collection of examples, which are themselves collections of features [21].

The learning process starts with observations or data, such as examples, direct experience, or instruction, so that we can seek patterns in data and

[1] https://www.britannica.com/technology/artificial-intelligence

make better decisions in the future based on the examples we provide [45].

2.1 Types of machine learning

Machine learning can be divided into four main categories as shown in the Figure 2.1

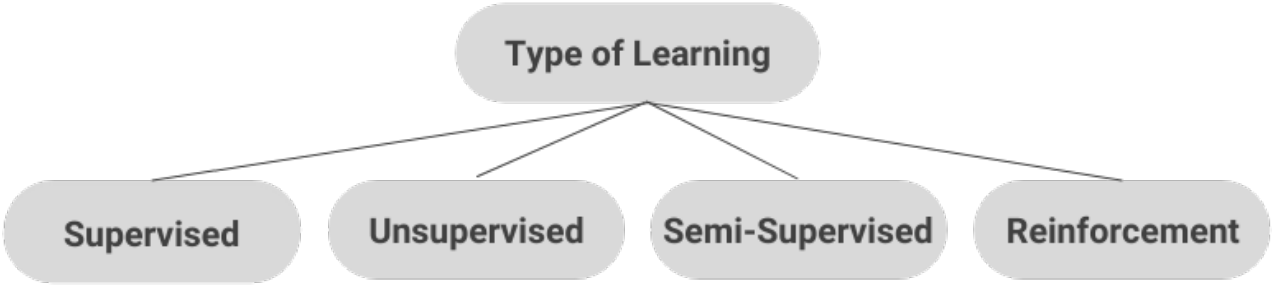


Figure 2.1: types of machine learning

2.1.1 Supervised learning

Supervised learning is an approach to creating artificial intelligence (AI), where a computer algorithm is trained on input data that has been labeled for a particular output. The model is trained until it can detect the underlying patterns and relationships between the input data and the output labels, enabling it to yield accurate labeling results when presented with never-before-seen data [40].

When learning to categorize handwritten digits, for example, a supervised learning system takes thousands of images of handwritten digits along with labels containing the correct number each image represents, the algorithm will then learn the relationship between the images and the numbers associated with them, and use that knowledge to identify entirely new images (without labels) that it has never seen before, as shown in Figure 2.2.

2.1.2 Unsupervised learning

Unsupervised learning refers to the use of artificial intelligence (AI) algorithms to identify patterns in data sets containing data points that are neither classified nor labeled. The algorithms are thus allowed to classify, label, and/or group the data points contained within the data sets without

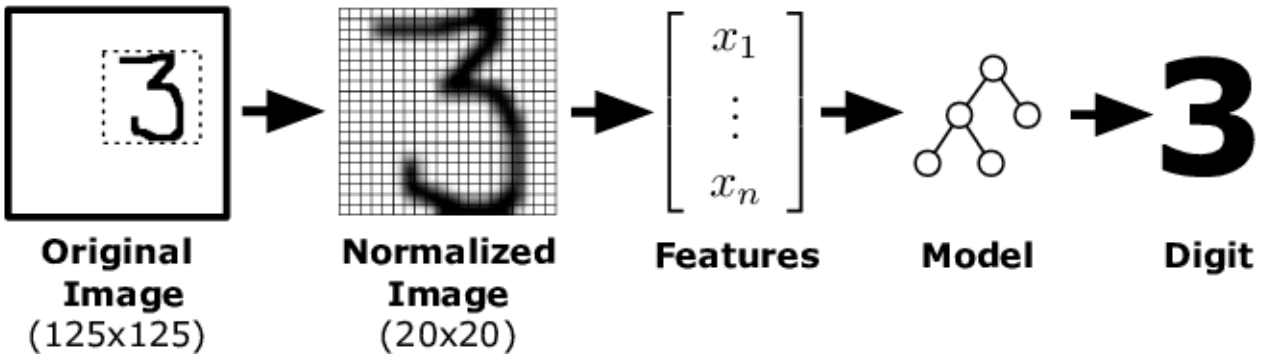


Figure 2.2: Handwritten digits classification model [39]

having any external guidance in performing that task. In other words, unsupervised learning allows the system to identify patterns within data sets on its own[2] [41].

How do you find the underlying structure of a dataset? How do you summarize it and group it most usefully? These are the goals of unsupervised learning. Examples of unsupervised learning applications are Customer segmentation or understanding, of different customer groups, around to build an effective marketing strategy as shown in Figure 2.3.

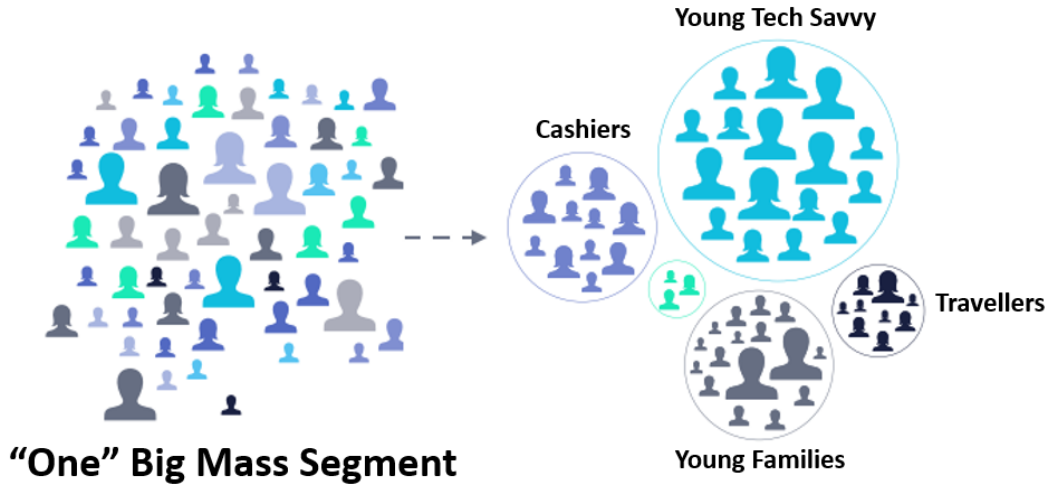


Figure 2.3: Unsupervised learning Segmentation [48]

2.1.3 Semi-Supervised learning

Semi-supervised learning is a learning problem that involves a small number of labeled examples and a large number of unlabeled examples. Semi-supervised learning is a type of machine learning that sits between supervised and unsupervised learning

[2]https://www.techtarget.com/searchenterpriseai/definition/unsupervised-learning

2.1.4 Reinforcement learning

"Reinforcement learning is learning what to do—how to map situations to actions so as to maximize a numerical reward signal. The learner is not told which actions to take but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics—trial-and-error search and delayed reward—are the two most important distinguishing features of reinforcement learning" [49].

In a typical RL a learner and a decision-maker called "agent" interact with the environment. As a result of the agent's actions, the environment provides rewards and transit to a new state. So, in reinforcement learning, we do not instruct an agent on how to do a task, but rather give it with rewards, which may be either positive or negative, depending on its actions [1]. And the agent will learn to choose actions that maximize the current and future rewards. This method of modeling the environment is called Markov-Decision Process (MDP) formalization

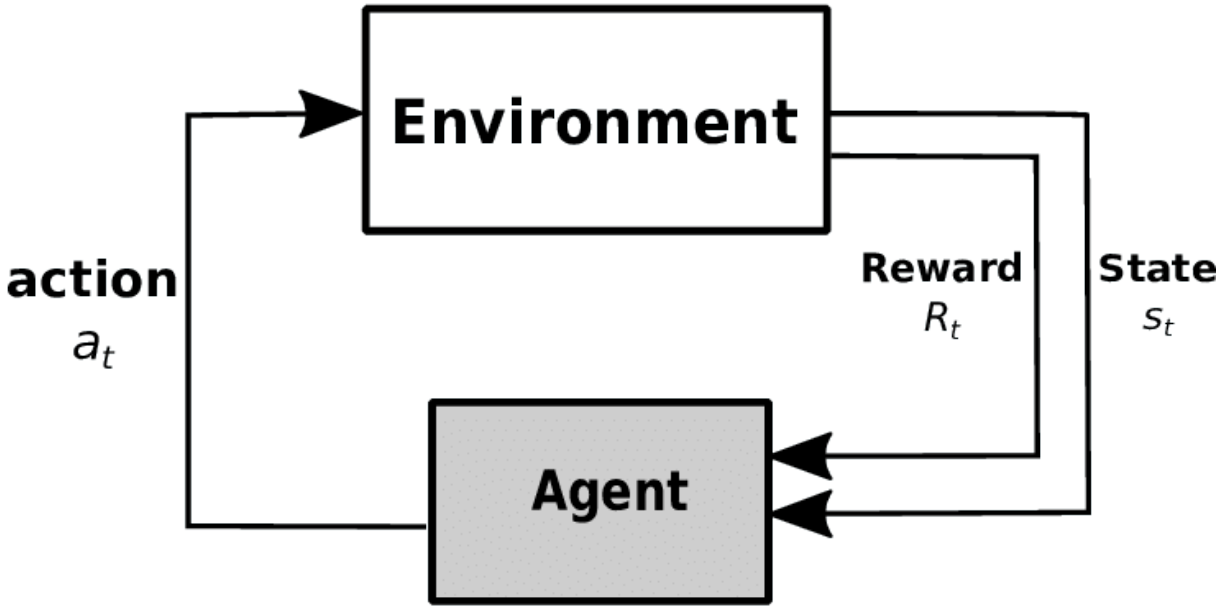


Figure 2.4: Reinforcement Learning, Agent and Environment [3]

- **Policy function** policy (π) describes the decision-making process of the agent. In general, a policy assigns probabilities to every action in every state, for example $\pi(s1|a1) = 0.3$ probability of taking action "a1" when state is "s1" is 30%. The policy thus represents a probability distribution for every state over all possible actions [20]. The policy function is commonly represented with the symbol (π)

- **Value function** The Value Function represents the value for the agent to be in a certain state. In simplified words, the value function represents how good to be in a specific state/ how good to take action in this specific state. There are two main important value functions:
  - **State-Value function** The state value function tells us the value for being in some state when following some policy [20]. The state value function is commonly represented with the symbol (V)
  - **Action-Value function** The action value function tells us the value of taking an action in some state when following a certain policy. The action value function is commonly represented with the symbol (Q)

2.2 Deep learning

The simple machine learning algorithms perform excellently on a wide range of significant problems, despite their simplicity. However, they were not successful in resolving the fundamental challenges of artificial intelligence, such as speech recognition and object detection, deep learning was developed in part as a result of the failures of standard algorithms to generalize effectively on AI challenges of this kind [21].

Understanding deep learning begins with an understanding of artificial neural networks since deep learning is just a sophisticated collection of tools for learning in neural networks.

Artificial Neural networks are simply an interconnected collection of simple processing pieces, units, or nodes whose operation is inspired by the animal neuron in some way. The interunit connection weights, store the network's processing capability, obtained by learning from a dataset [22]. The artificial neural networks are inspired by brain neurons. There are around 10 billion neurons in the human brain like the one shown in this figure 2.5.

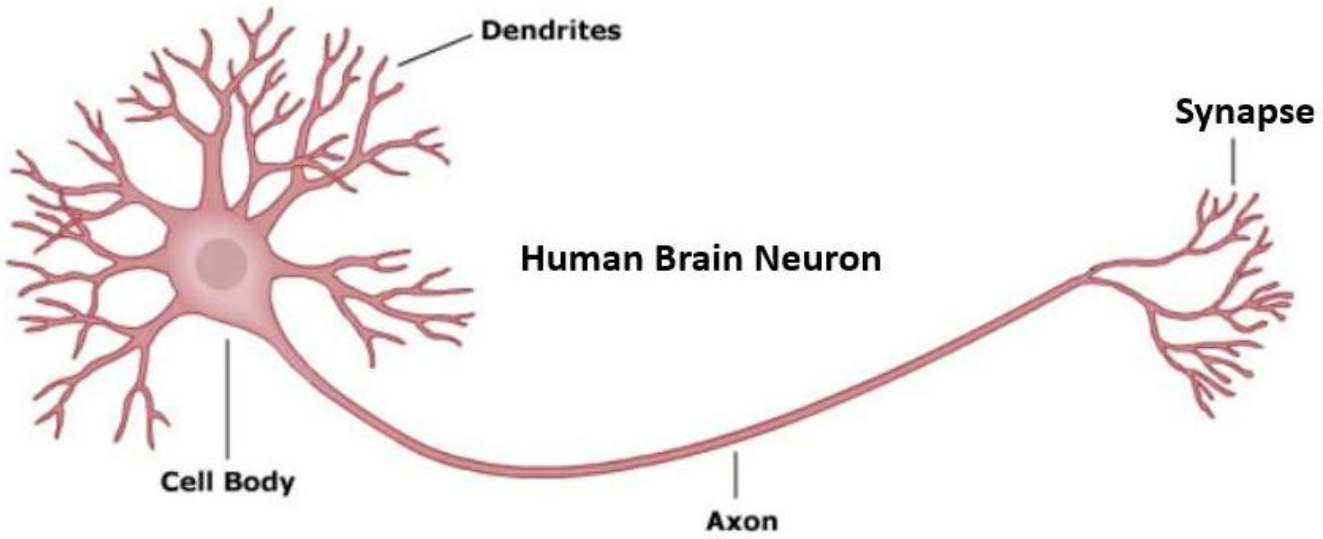


Figure 2.5: Illustration of a neuron [8].

A single artificial neuron (or perceptron) can be imagined as a single Logistic Regression model. This Figure 2.6 shows the artificial equivalents of biological neurons.

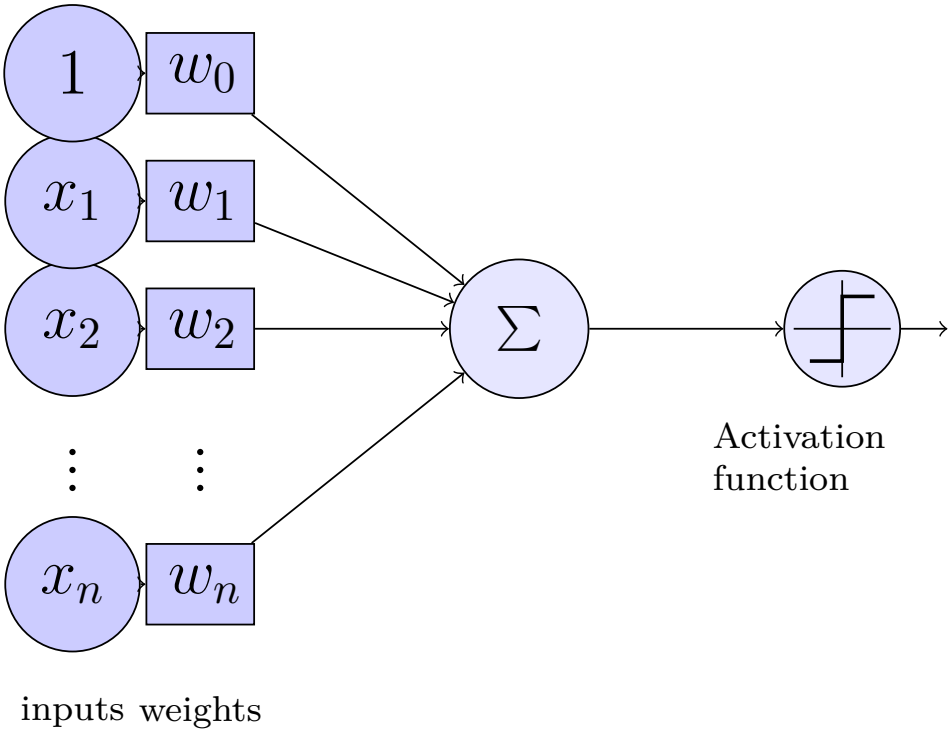


Figure 2.6: Artificial Neuron

A neural network is created by layering together neurons in several different configurations. There are three levels of layers: the input layer, which accepts the input of the model, single or multiple hidden layers, each of which

accepts inputs from the previously hidden layer or the input layer, and an output layer, which accepts the input from the last hidden layer and as his name suggests will output the result (prediction, action, etc.) of the model as shown in this figure 2.7. The strength of neural networks lies in their capacity to approximate and map inputs to outputs by learning from examples in a complex nonlinear manner.

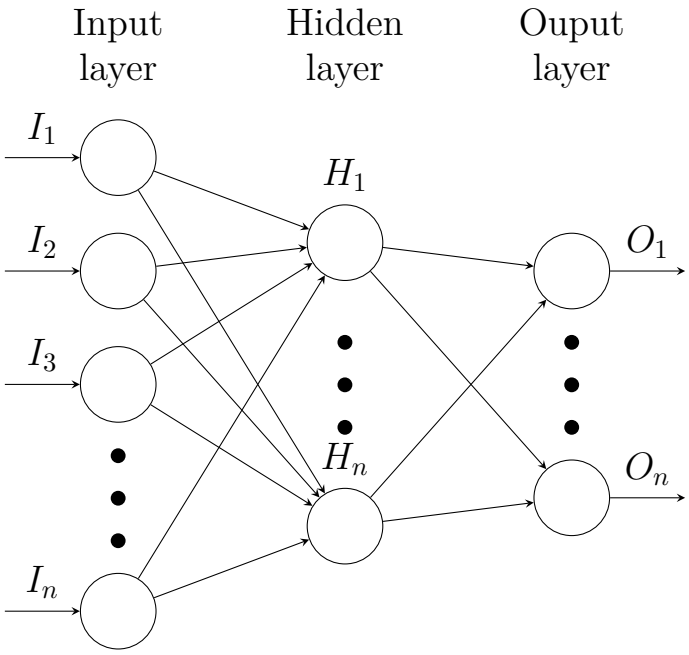


Figure 2.7: Artificial Neural Network ANN (Feedforward)

There are many different kinds of neural networks, and we will go through three important types.

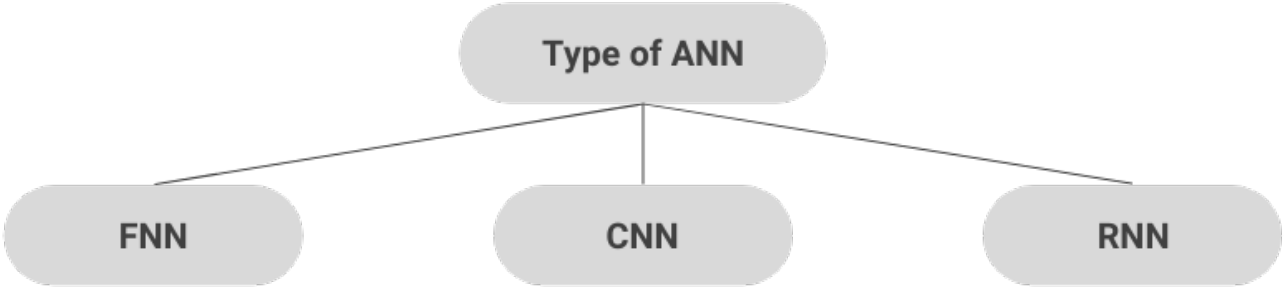


Figure 2.8: Types of Artificial Neural Network

2.2.1 Feedforward Neural Network (FNN)

Feedforward neural networks, also often called multilayer perceptrons (MLP) is the standard and simplest models of deep learning. Consists of multi per-

ceptrons (Artificial neurons) stacked in multiple layers, with each layer's output serving as the input for the following layer. It is referred to as Feedforward since information is only processed in a single direction without cycles or recursion as shown in the figure 2.9. These networks are often referred to as "Universal Function Approximators" because they have the capability of learning weights that approximate any input to an output with a high degree of precision.

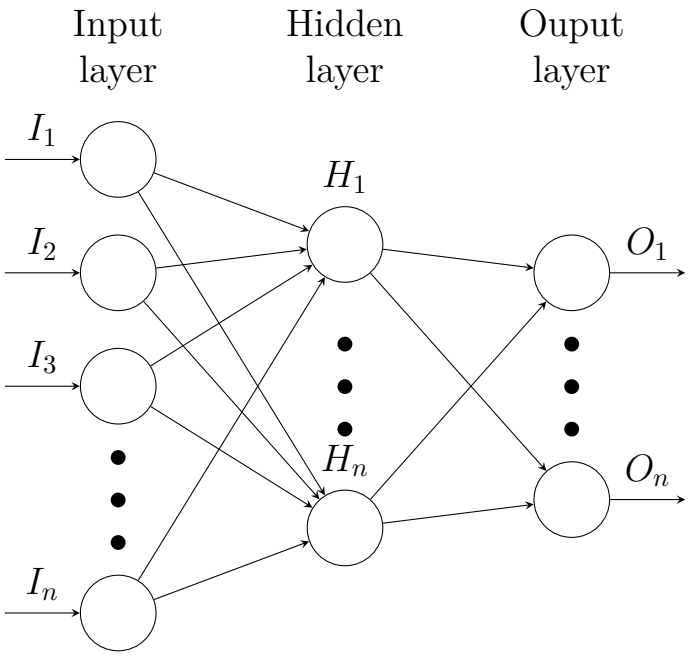


Figure 2.9: Feedforward Neural Network

2.2.2 Convolutional Neural Network (CNN)

Recent years have seen a significant increase in interest in convolutional neural networks within the deep learning community, due to their efficiency and performance in processing two-dimensional data with grid-like topologies, such as images and videos. When constructing CNNs, the general matrix multiplication operation used in standard Feedforward neural networks is replaced with a convolution operation, which helps in the detection of features in 2d and 3d matrices while simultaneously reducing the number of weights used and, as a result, the complexity of the network. Furthermore, the raw images can be imported directly to the network without the need for feature extraction to be performed beforehand. With the fast growth of computer hardware such as graphics processing units (GPUs) and tensor processing units (TPUs), training CNNs became easier and more efficient, opening the door to deeper networks with hundreds of layers. Currently, CNNs have been effectively used for a variety of tasks such as handwriting recognition, face detection, speech recognition, recommender systems, image classification, and natural language

processing (NLP) [30]. As shown in figure 2.10, CNNs are composed of three distinct types of layers. Convolution layers (L=1, L=3), subsampling or pooling layers (L=2, L=4), (Typically, each convolutional layer is followed by a pooling layer) and finally fully connected or Dense layers, which are the same as in Feedforward networks.

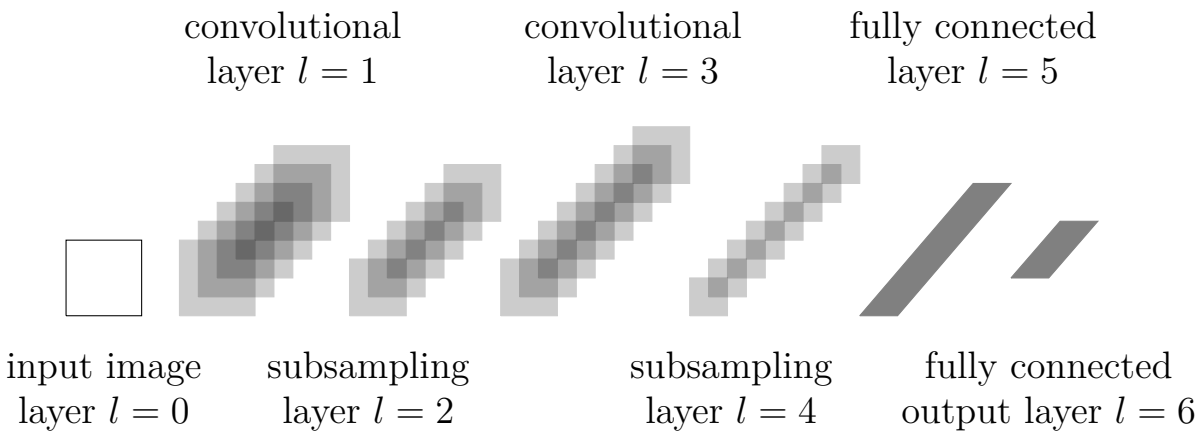


Figure 2.10: High-level illustration of a simple convolutional neural network indicating convolutional layers, pooling layers and fully-connected layers.

2.2.3 Recurrent Neural Networks (RNN)

All of the networks that have been presented so far deal with multidimensional data that has independent properties (every instance of the dataset is independent of other instances). However, some data kinds, such as text and time-series, as well as biological data, are, on the other hand, sequential and dependent on one another [2]. Examples of such dependencies:

- Predicting future stock prices based on historical data is an example of a time series. Without knowledge of past time steps, it is impossible to determine whether prices have increased or decreased at a given point in time. By treating every time step independently, we would lose this valuable information.
- By processing text with a bag of words technique the ordering of words in the document will be ignored this approach works fine in certain areas such as sentiment analysis, However in certain areas where an order of words is important such as voice command this approach will be inadequate [2].

To deal with sequential data Recurrent Neural Network (RNN) architecture is introduced. They can do this because they have loops that let data be stored inside the network (hence the name recurrent). This way, RNNs

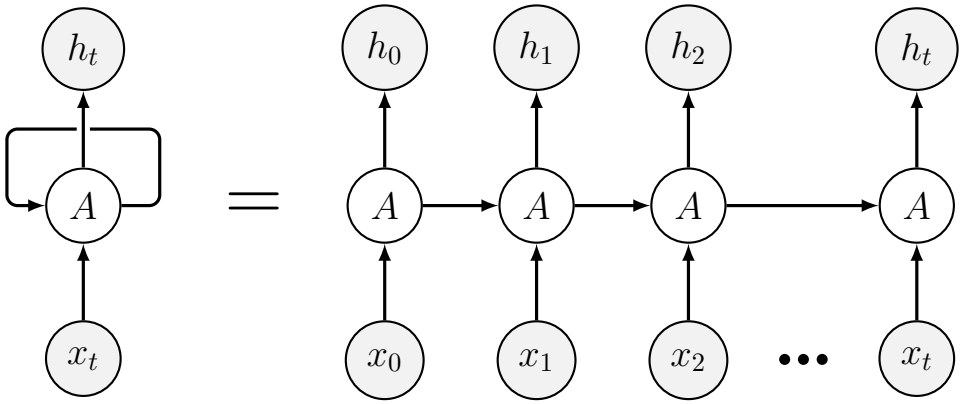


Figure 2.11: High-level illustration of a simple Recurrent Neural Network (RNN)

can pick up on sequential information in the data. It is possible to think of a recurrent neural network as a collection of numerous copies of the same network, each of which produces a signal to its successor as shown in the figure 2.11.

When working with RNN networks, there are a few complicating factors. One of them is the vanishing and the exploding gradient problem [2]. The vanishing happens during the backpropagation phase when the algorithm moves backward from output to input layer, gradients get smaller and smaller slowly approaching zero, leaving the weights of the first and lower layers almost unchanged which will cause the optimizing function (gradient descent, adam, etc) to never converge. In contrast, exploding gradients happen when gradients get larger and larger during the backpropagation step which eventually will cause the optimizer to diverge [9]. There is also the issue of RNN networks losing track of outdated data that may have been fed into the system. It is theoretically possible for RNNs to handle "long-term dependencies," but in practice, RNNs seem incapable of doing so [38], this problem was explored in-depth in this paper [7]. To overcome these issues, variants of RNNs were introduced, LSTMs and GRUs.

- **LSTM** Long Short Term Memory networks – often referred to as "LSTMs" are a type of Recurrent neural network that is capable of learning long-term relationships between variables. LSTM networks were invented by Hochreiter and Schmidhuber [26]. And have since been refined and popularized by many other researchers. LSTMs are specifically intended to prevent the issue of long-term dependency in the first place. The ability to retain information over extended periods is their default mode of operation [38].

- **GRU** Gated Recurrent Neural Network was introduced in 2014 by Kyunghyun Cho, Bart van Merrienboer and Caglar Gulcehre [15]. GRU

is a simpler version of LSMT with the same goal of addressing the issues of long-term dependency and vanishing/exploding gradient.

2.3 Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) is a relatively new branch of machine learning that combines two powerful techniques (Deep learning + Reinforcement learning). Reinforcement learning took birth in early 1980 as dynamic programming descendent Sutton and Barto explain the history and principles of reinforcement learning and the birth of DRL in their book [49]. In brief, reinforcement learning was used where the data were limited and few and the requirement were complex, however as a result of the emergence and advent of deep networks Reinforcement learning has gained more power to tackle more complex problems. More details about Reinforcement learning in this reviews [28, 4]

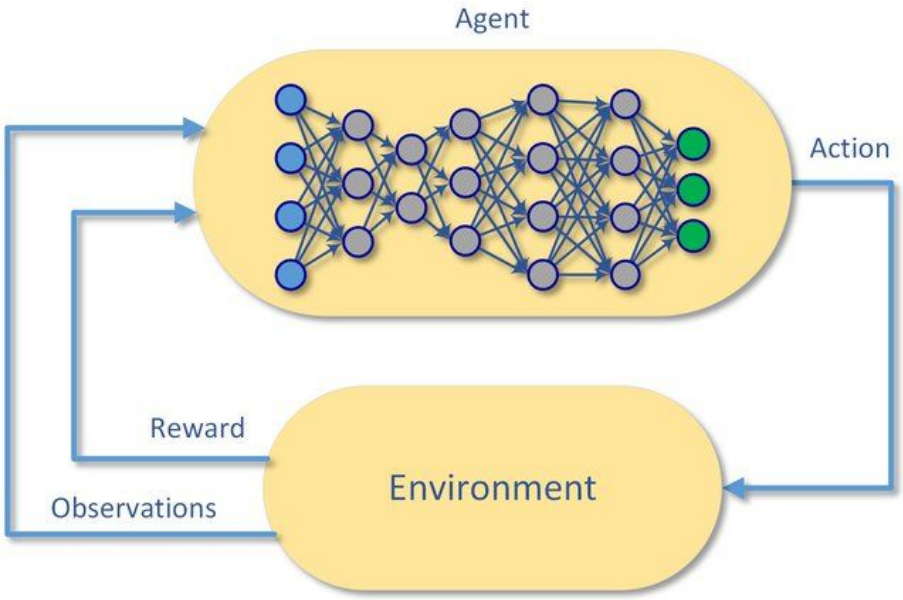


Figure 2.12: Deep reinforcement learning illustration [27].

In the literature, there are three main types of deep reinforcement learning

2.3.1 Critic-only

These methods are based on estimating the value function (State value-function/ Action value-function) of being in a given state. The learning (estimating the value function) early on was done by methods called dynamic programming but since the emergence of deep learning, the learning process was much more effective, fast, and efficient by using deep learning algorithms

[4]. To take an action in the critic-only, we just use the action-value function. We can choose to be greedy and constantly choose the action that provides the highest possible future reward, or we can add an exploration factor so the agent will continue learning from new experiences [42]. One of the Critic-only disadvantages is it can't be applied on a continuous space of actions, works only on a discreet limited set of actions the agent can perform.

- **Deep Q-Network (DQN)** were originally proposed by DeepMind back in 2013 [35] and it was the first algorithm that brought the capabilities of deep learning to reinforcement learning. The only difference between Q-learning and DQN is that Q-learning employs temporal difference to estimate the Q function while Deep Q-learning uses a neural network to approximate it.

- **Deep SARSA** algorithm is very similar to Q-learning, the difference is in the way the value function is updated, update is done using the value of the future state and the action of the current policy [31].

2.3.2 Actor-only

If we consider the policy to be a probability distribution over the whole action space, Policy Gradient techniques may be used to optimize that distribution such that the agent picks the action with the highest probability, out of all the potential actions, that yields the highest reward [47]. A major advantage of actor-only methods over critic-only methods is that they allow the policy to generate actions in the complete continuous action space.

- **Deterministic Policy Gradient (DPG)** This is the first algorithm that deals with continuous action space, it was first introduced in this paper [47].

- **Augmented Random Search (ARS)** The ARS is an improved version of BRS (Basic Random Search), it was first introduced in 2018 in this paper [33]. The authors state that they have created an algorithm that would be at least 15 times more effective than the fastest competing model-free approaches.

2.3.3 Actor-critic

In the Actor-Critic method, the policy is referred to as the actor that proposes a set of possible actions given a state, and the estimated value function is referred to as the critic, which evaluates actions taken by the actor

based on the given policy. Actor-critic methods combine the advantages of actor-only and critic-only methods. While the parameterized actor brings the advantage of computing continuous actions the critic's estimate of the expected return allows for the actor to update with gradients that have lower variance [4]

- **Proximal Policy Optimization (PPO)** Introduced by the OpenAI team in this paper [44] and it has become the primary reinforcement learning algorithm at OpenAI due to its simplicity of use and high performance. The objective was to create an algorithm with data efficiency and consistent performance that perform comparable to or better than state-of-the-art techniques while being considerably easier to develop and tune.

- **Asynchronous Advantage Actor-Critic (A2C or A3C)** This algorithm was first mentioned in 2016 in a research paper appropriately named Asynchronous Methods for Deep Learning [34] by Google's DeepMind team which is the Artificial Intelligence branch of Google. And it was the first algorithm that uses two neural networks (policy and value function network). A3C is an asynchronous version of A2C.

- **Deep Deterministic Policy Gradient (DDPG)** is a model-free off-policy algorithm for learning continuous actions It combines ideas from DPG (Deterministic Policy Gradient) and DQN (Deep Q-Network), it uses DPG for policy network and DQN for value function network [29].

CHAPTER 3

Experiment Design

In this chapter, we will explain the design experiment. First of all, we will start by describing each technology and library used in this project, and then we will go into explaining the experimentation design and describing the training and testing process.

3.1 Tools and Technologies

In this section, we will list all technologies and tools used in this project

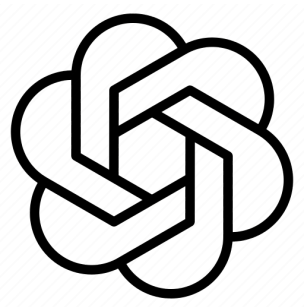
Python language

"**Python** is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library"[1].

[1]https://en.wikipedia.org/wiki/Python_(programming_language)

We can distinguish two main parts in this project, AI reinforcement learning, and the backtesting platform, we will describe technologies used to implement each part

3.1.1 AI Reinforcement Learning Technologies



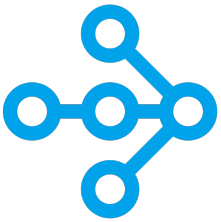
OpenAi Gym

"**OpenAi Gym** provides a standard API for reinforcement learning environments, this API is compatible with nearly all reinforcement learning training frameworks. OpenAi gym allows us to separate environment design and implementation from reinforcement learning training. OpenAi gym also contains a list of predefined ready-to-use reinforcement learning environments."[2].

PyTorch

"**PyTorch** is an open-source machine learning framework based on the Torch library. It is used for applications like computer vision and natural language processing and was mostly made by Meta AI. It is free software that can be used by anyone. It is licensed under the Modified BSD license. PyTorch has both a Python interface and a C++ interface. The Python interface is more polished and is the main focus of development."[3].

.



Ray

"**Ray** ray is an open-source project that was made at the RISE Lab at UC Berkeley. As a general-purpose and universal distributed compute framework, you can run any compute-intensive Python job, from distributed training or hyperparameter tuning to deep reinforcement learning and production model serving"[4].

[2]https://www.gymlibrary.ml/
[3]https://en.wikipedia.org/wiki/PyTorch
[4]https://ray.io

Rlib

"**Rlib** is the standard Python framework for reinforcement learning. It was built on Ray. It is made to be easy to change and get into production quickly. It has more than 25 of the newest algorithms, all of which are built to run at scale and in multi-agent mode"[5].

Tune

"**Tune** is a Python library that lets you run experiments and change the values of hyperparameters at any scale. We can tune in any machine learning framework, such as PyTorch, XGBoost, TensorFlow, or Keras, and choose from the latest algorithms, such as Population Based Training (PBT), BayesOptSearch, or HyperBand/ASHA. Tune works with many hyperparameter optimization tools, such as Optuna, Hyperopt, Ax, and Nevergrad, to name a few"[6].

.

Numpy

"**Numpy** numpy is an open-source library for the Python programming language. It adds support for large, multi-dimensional arrays and matrices, as well as a large collection of high-level mathematical functions that can be used on these arrays"[7].

[5]https://www.ray.io/rllib
[6]https://www.ray.io/ray-tune
[7]https://numpy.org

.

Pandas

"**Pandas** pandas is a data manipulation and analysis toolkit created in the Python computer language. It contains data structures and methods for manipulating numerical tables and time series in particular. It is free software distributed under the BSD license's three clauses"[8].

.

Tensorboard

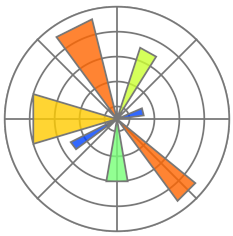
"**Tensorboard** is a tool that helps with the machine learning workflow by giving you the measurements and visuals you need. It lets you track experiment metrics like loss and accuracy, see the model graph, project embeddings to a lower-dimensional space, and do a lot more"[9].

.

TA

"**TA** is a library for technical analysis that can be used to build features from financial time series datasets (Open, Close, High, Low, Volume). It was made with Pandas and Numpy"[10].

.



Matplotlib

"**Matplotlib** produces figures that are good enough to be published in several hardcopy formats and interactive environments. Matplotlib can be used in Python scripts, the Python and IPython shells, web application servers, and different graphical user interface toolkits"[11].

[8] https://pandas.pydata.org
[9] https://www.tensorflow.org/tensorboard
[10] https://github.com/bukosabino/ta
[11] https://matplotlib.org

3.1.2 Back-testing platform technologies

Backend

FastApi

"**FastApi** is a web framework for building APIs with Python that is modern, fast, and based on standard Python-type hints. The most important things about fastapi are High level of performance, Fast to code: Speed up the development of features Easy: made to be easy to use and understand"[12].

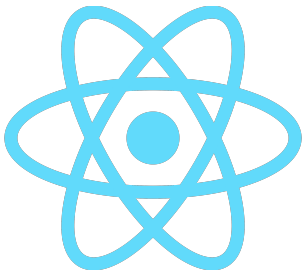
.

Websockets

"**Websockets** is a Python library for making WebSocket servers and clients that focuses on correctness, simplicity, reliability, and speed. It is built on top of asyncio, Python's standard asynchronous I/O framework, and offers a coroutine-based API that is very elegant"[13].

Frontend

.



React JS

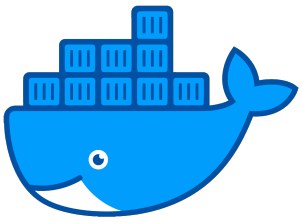
"**React** also called React.js or ReactJS, is a free and open-source JavaScript library for building user interfaces from UI components. It is maintained by Meta (which used to be Facebook) and a group of developers and companies"[14].

[12]https://fastapi.tiangolo.com
[13]https://websockets.readthedocs.io
[14]https://reactjs.org

MUI

”**Material ui** is a React UI library with interactive components that are ready for prototyping and production. MUI offers a full set of UI tools to help ship new features faster, and all of the components are based on the Google Material Design principles”[15].

Deployment



Docker

”**Docker** is a set of platform as a service (PaaS) products that deliver software in packages called containers by using OS-level virtualization. There are both free and paid tiers for the service. Docker Engine is the software that hosts the containers. It began in 2013 and is being worked on by Docker, Inc”[16].

Nginx

”**Nginx** is a free, open-source web server that can also work as a load balancer, mail proxy, HTTP cache, and reverse proxy. Igor Sysoev made the software, which came out to the public in 2004”[17].

[15] https://mui.com
[16] https://www.docker.com
[17] https://www.nginx.com

Uvicorn

"**Uvicorn** is a web server that is compatible with the ASGI protocol (async server gateway interface). It is the binding element that manages the web connections coming from the browser or API client, which subsequently makes it possible for FastAPI to fulfill the request itself. Uvicorn is responsible for listening on a socket, receiving the connection, doing some processing, and then handing over the request to FastAPI following the ASGI protocol"[18].

3.2 OHLCV Dataset

OHLCV is a market data aggregation and acronym that stands for Open, High, Low, Close, and Volume. The OHLCV data consists of five types of data that are most common in financial analysis: the Open and Close, which reflect the initial and latest price levels reached over a specific time, respectively. The terms High and Low refer to the highest and lowest prices that were obtained during that period. The entire quantity of assets transacted within that period is referred to as volume.

- **Open** the opening price at the beginning of the period.
- **High**: The highest price, asset reached during the period.
- **Low**: The lowest price, asset reached during the period.
- **Close**: the closing price at the end of the period.
- **Volume**: The amount of the asset bought and sold during that period.

These five data points are recorded during a certain period, the most common periods are 1 minute, 15 minutes, 1 hour, 4 hours, 1 day, and 1 week.

The Figure 3.1 and Figure 3.3, illustrate how the ohlcv data is represented in a candle stick graph.

[18] https://www.uvicorn.org

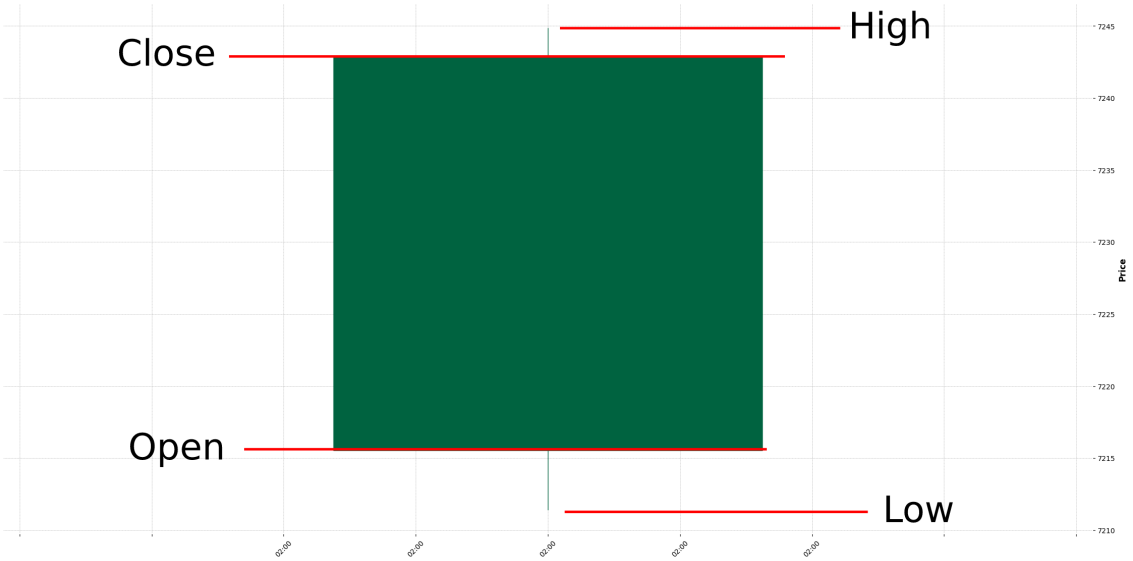


Figure 3.1: OHLCV data candle stick

Figure 3.2: OHLCV graph of Bitcoin prices for 1 day (2020-01-01) of 1 hour time period

We created a python script to fetch the dataset from the Binance platform to train our model on this data.

```
/opt/dev/trading/pfe/DRL-trading-bot | develop !1 ?8
./fetch_data.py binance --key secrets.json --period 1h --symbol "BTCUSDT" --start-date "2019-01-01" --end-date "2021-01-01" --output dataset/
Fetching data ...
Fetched 17500 entries of BTCUSDT
Period of 1h from 2019-01-01 to 2021-01-01
```

Figure 3.3: Fetch data script for fetching crypto currencies ohlcv data from binance platform

- –**key**: Secret key (API key) for the binance exchange.
- –**period**: Period of data (EX: '1m' '1h' '1d' '1m').
- –**symbol** Crypto Currency pair symbol (EX: BTCUSDT).
- –**start-date**: Data staring date.
- –**end-date**: Data ending date.
- –**output**: Output folder to save dataset to.

Before training, the downloaded dataset is cleaned up and new indicators are added to it. Moreover about this process is in the next section.

3.3 Reinforcement learning environment

As stated in previous sections, reinforcement learning can be splitted into two main parts, the environment, and the agent, the agent interacts with the environment by executing actions in our case actions are buy and sell orders and the environment in return provide two things:

- Observations or State, in our case observations, are prices and indicators.
- Reward (could be also a penalty) value, that represents how good that action was.

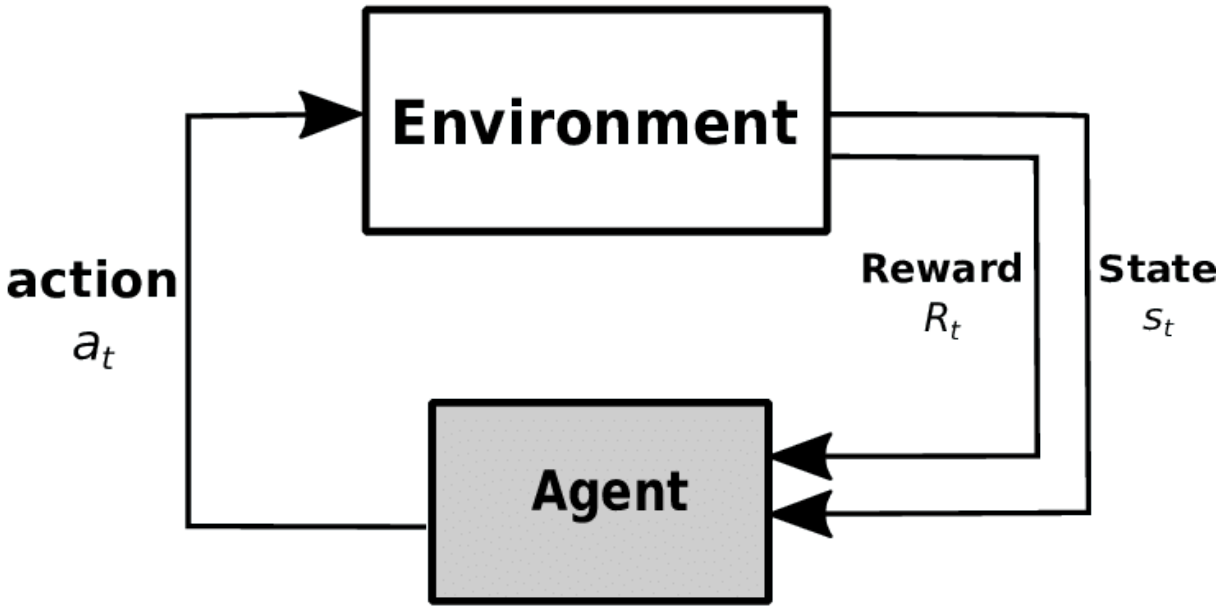


Figure 3.4: Reinforcement Learning, Agent and Environment [3]

The main goal was to create a modular environment that can be changed and tweaked easily if we ever wanted to deploy the trained model on a live market. In this section, we will discuss the implementation of the environment, and describe the observation space and different reward functions tested.

The implemented environment follows the OpenAi gym specifications, by following these specifications we assure compatibility with most of the RL framework since OpenAi gym is by far the most supported one. The environment has 4 main components:

- **Data Provider** which is responsible for providing data and observations for the environment.
- **Reward** which is responsible for calculating the value of the reward.
- **Trading Strategy** which is responsible for executing the actions provided by the agent.
- **Rendering component** which is responsible for rendering and providing a visual representation of the trading environment.

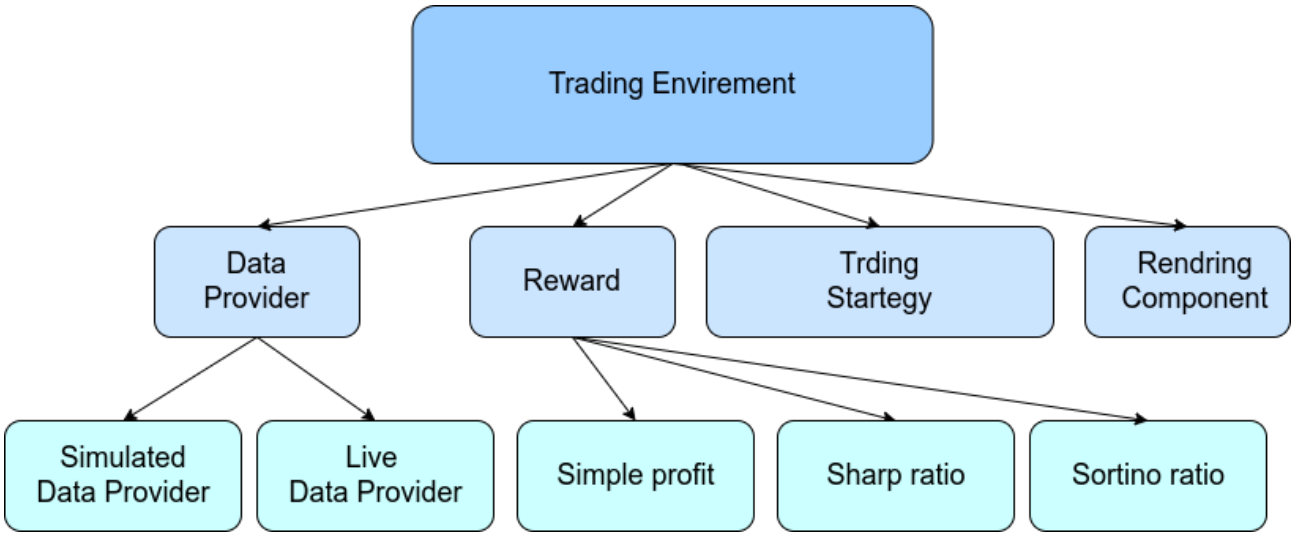


Figure 3.5: Trading environment graph.

3.3.1 Data Provider

This component is responsible for processing the raw OHLCV data and providing observations to the environment. By having this component separated from the environment we can create multiple versions and types of observations without changing the environment itself.

The data processing phase consists of normalizing the OHLCV data and adding technical indicators. The list of added indicators is listed in Table 3.1.

There are two implemented types of data provider, Simulated Data provider and Live Data Provider

- **Simulated Data Provider:** Create observations from historical OHLCV prices (this type of provider is used during the training phase and evaluation phase)
- **Live Data Provider:** Create observation from current market prices, this means that this provider will fetch live prices from the Binance exchange. This provider is used when deploying the model on a live market

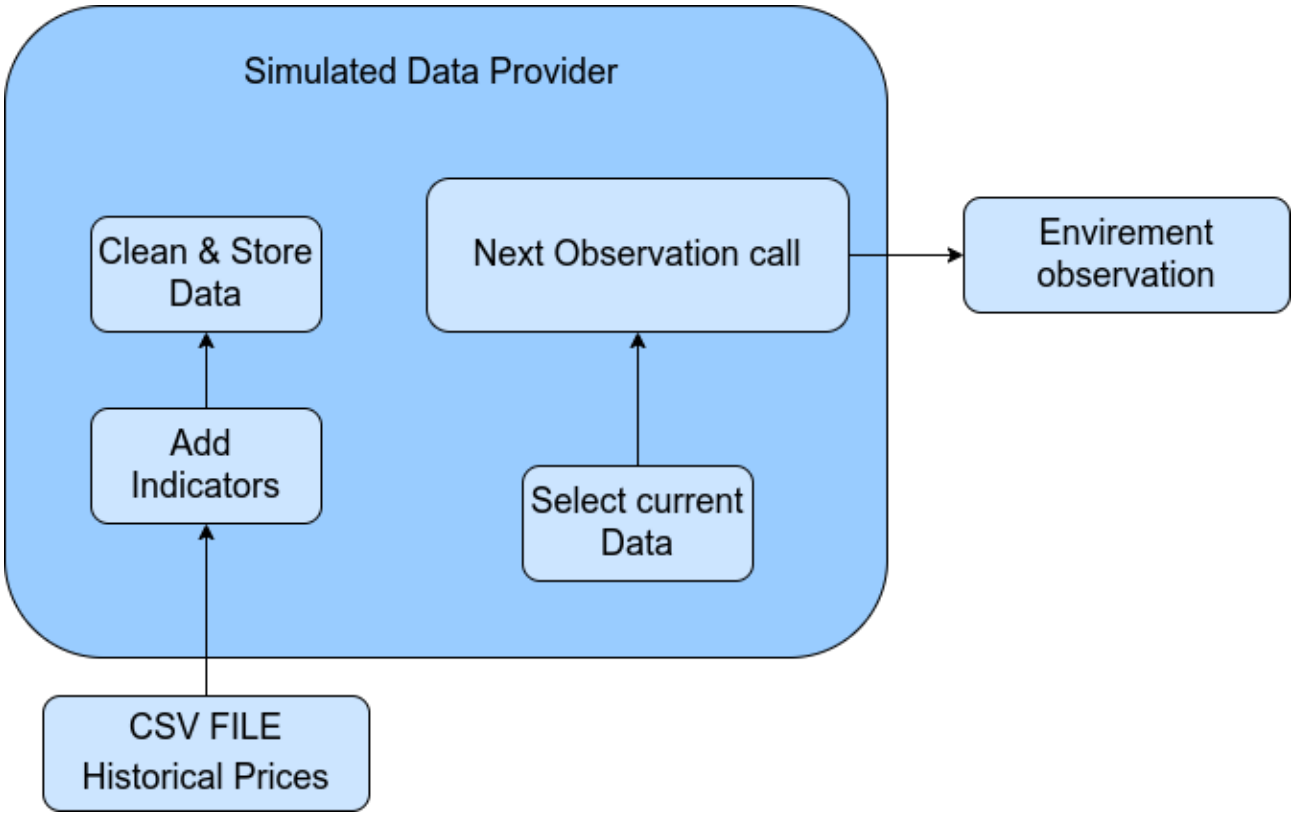


Figure 3.6: Simulated Data provider.

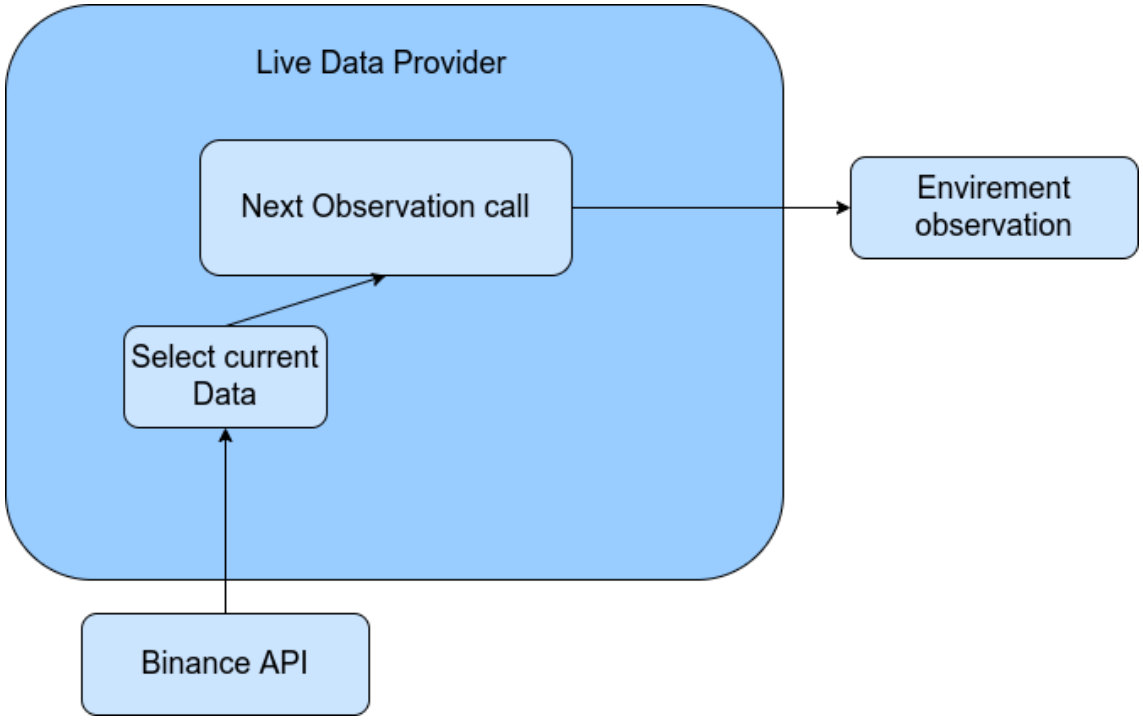


Figure 3.7: Live Data provider.

Momentum indicators	**Volume indicators**	**Trend indicators**	**Volatility indicators**
Relative Strength Index (RSI), True Strength Index (TSI), Ultimate Oscillator(UO), Awesome Oscillator (AO)	Money Flow Index (MFI), Accumulation Distribution Index (ADI), On-balance volume (OBV), Chaikin Money Flow (CMF), Force Index (FI), Ease of movement (EoM, EMV),	Moving Average Convergence Divergence (MACD), Vortex Indicator (VI), Trix (TRIX), Mass Index (MI), Commodity Channel Index (CCI), Detrended Price Oscillator (DPO), KST Oscillator (KST), Aroon Indicator,	Bollinger Bands, KeltnerChannel,

Table 3.1: List of technical indicators added to the raw OHLCV data

3.3.2 Reward

The reward function is responsible for informing the agent of what is right and what is wrong via the use of rewards and punishments (reward functions represent the feedback to the agent). Moreover, the primary objective of the reinforcement learning agent is to maximize the reward. This implies that the agent's behavior is entirely dictated by the reward function. As a result, it is vital to have a good reward function in place for a good performance. From risk-based measurements to profitability or cumulative return, there are many different types of rewards functions we can choose from, see this [43] article for a comprehensive review about reinforcement learning reward functions in trading. In this project we tested and experimented with three different reward functions, we will describe each one of them.

3.3.2.1 Simple profit

This is the simplest reward function, it calculates the ratio between current net worth and the previous net worth. The formula for the simple profit reward function is described in Equation 3.1.

$$Reward = \frac{current\_networth}{previous\_networth} - 1 \tag{3.1}$$

3.3.2.2 Sharp ratio

The Sharpe ratio [46] is a risk-based measurement also called the reward-to-variability ratio and is the most common portfolio management metric that indicates how well an equity investment is performing compared to a risk-free investment, taking into consideration the additional risk level involved with holding the equity investment. For an investment or a trade to have a high Sharpe ratio, it needs both high returns and low volatility (i.e. risk). The formula for sharp ratio is described in Equation 3.2.

$$Reward = \frac{mean(networth\_returns)}{\sigma(networth\_returns)} \tag{3.2}$$

3.3.2.3 Sortino ratio

The Sortino ratio is a variation of the Sharpe ratio that measures the performance of the investment relative to the downward deviation. Unlike Sharpe, the Sortino ratio does not consider the total volatility of the investment. For an investment to have a high Sortino ratio, it needs both high re-

turns and low volatility in return downside only. The formula for the Sortino ratio is described in Equation 3.3.

$$Reward = \frac{mean(networth\_returns)}{\sigma(networth\_downside\_returns)} \tag{3.3}$$

3.3.3 Trading Strategy

This component is responsible for executing actions coming from the agent. The main idea is to create one action for holding and multiple actions for buy and sell for example 10 action sell and 10 actions buy, to simplify things action sell 1 means sell 10% of the asset and action sell 9 means sell 90% of the asset and same things for buy actions. By doing this the agent will have more control over the quantity sold and bought. This component also simulates slippage by using random numbers.

3.3.4 Rendering component

This component is responsible for rendering visualization for the environment to get an idea about agent behavior in the market. The visualization is implemented using the matplotlib library, it's interactive and real-time (shows agent live-action). The rendered UI is shown in Figure 3.8.

Figure 3.8: Environment rendering UI

The user interface displays actions taken by the agent (buy/sell/hold) alongside asset prices, as well as the net worth of the agents and the net worth of other benchmarks (benchmarks are explained in the next section). On the left, it shows the current asset and balance held.

3.4 Training and Testing strategies

In this section, we will explain the strategy used for training and testing reinforcement learning.

Dataset used in this project is two years of hourly OHLCV prices, from 2019-01-01 to 2021-01-01, the first year (from 2019-01-01 to 2019-12-30) was used for training and the second year (from 2020-01-01 to 2020-12-30) was used for backtesting the agent.

3.4.1 Training strategy

Reinforcement learning is one of the most demanding algorithms in terms of processing power. It requires more CPU and ram than traditional deep learning since the dataset is infrenced (generated) in real-time by interacting with the environment. To speed up the training and inference process, some techniques were employed.

- Using a vectorized environment means creating multiple environment instances in one vectorized environment, allowing to infer multiple values (observations + reward) from a list of environments in each step.
- Using rllib multiple workers for asynchronously inferencing data from vectorized environments, which significantly speeds up the inference process.

Since reinforcement learning algorithms are known to be noisy and can diverge quickly, a checkpoint technique was used. This means that the weights and network configuration are saved every certain number of iterations. This lets us have models at each point in time that can be easily restored and tested. The training process is described in Figure 3.9.

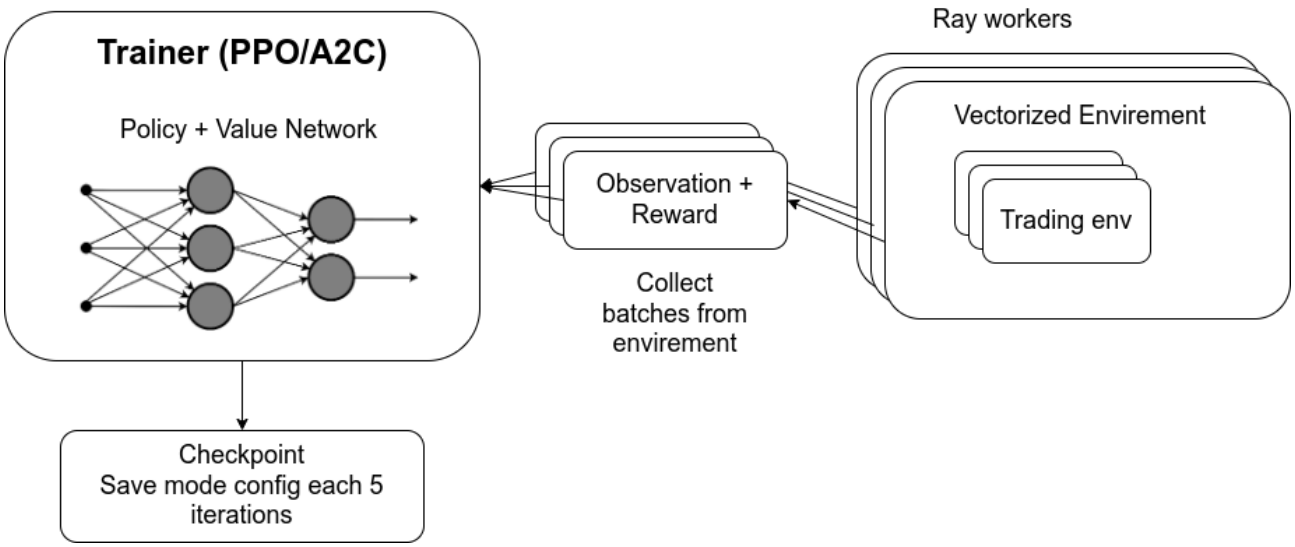


Figure 3.9: Training process.

The training was done in two identical virtual machine instances from Oracle cloud, their specs are described in Figure 3.10.

Shape summary ⓘ Edit shape

Shape: VM.Standard.E3.Flex
Memory (GB): 32
OCPU count: 6
Max total VNICs: 6
Network bandwidth (Gbps): 6
Local disk: -

Figure 3.10: Oracle Cloud VM shape

Since the training is conducted in a cloud server, it was more convenient to create a command-line interface (CLI) to train the agent easily as shown in Figure 3.11.

```
/opt/dev/trading/pfe/DRL-trading-bot | develop ↑1 ?7
└─ ./optimize.py ray_train -a PPO -r sortino -data dataset/binance-BTCUSDT-1h-2019-01__2021-01.csv --add-indicators
2022-06-08 11:53:15,650 INFO services.py:1456 -- View the Ray dashboard at http://10.0.0.254:8265
2022-06-08 11:53:17,571 INFO trial_runner.py:803 -- starting binance-BTCUSDT-1h-2019-01__2021-01
(PPOTrainer pid=20468) 2022-06-08 11:53:21,550  WARNING ppo.py:240 -- `train_batch_size` (4000) cannot be achieved wit
ment_length=200)! Auto-adjusting `rollout_fragment_length` to 363.
(PPOTrainer pid=20468) 2022-06-08 11:53:21,550  INFO ppo.py:268 -- In multi-agent mode, policies will be optimized seq
True if this doesn't work for you.
(RolloutWorker pid=20527) 2022-06-08 11:53:32,503       WARNING rollout_worker.py:498 -- We've added a module for chec
onment to fail if your environment is not set upcorrectly. You can disable check env by setting `disable_env_checking`
nt checking module standalone by calling ray.rllib.utils.check_env(env).
(RolloutWorker pid=20527) 2022-06-08 11:53:32,503       WARNING env.py:120 -- Your env doesn't have a .spec.max_episod
 dictionary, or `soft_horizon`. However, if you haven't, 'horizon' will default to infinity, and your environment will
(RolloutWorker pid=20525) 2022-06-08 11:53:32,794       WARNING rollout_worker.py:498 -- We've added a module for chec
onment to fail if your environment is not set upcorrectly. You can disable check env by setting `disable_env_checking`
```

Figure 3.11: Training reinforcement learning agent cli

- **-a**: Algorithm used to train the agent (EX: PPO, A2C ..).

- **-r**: Reward function used (EX: simple_profit, sharp, sortino).
- **-data** Dataset used to train.
- –**add_indicators**: This flag when set technical indicators are added to the observation space.
- **-lstm**: This flag wraps agent network with lstm layer.

Ray Dashboard was launched when starting a training session to monitor the VM's performances, ray dashboard is shown in Figure 3.12.

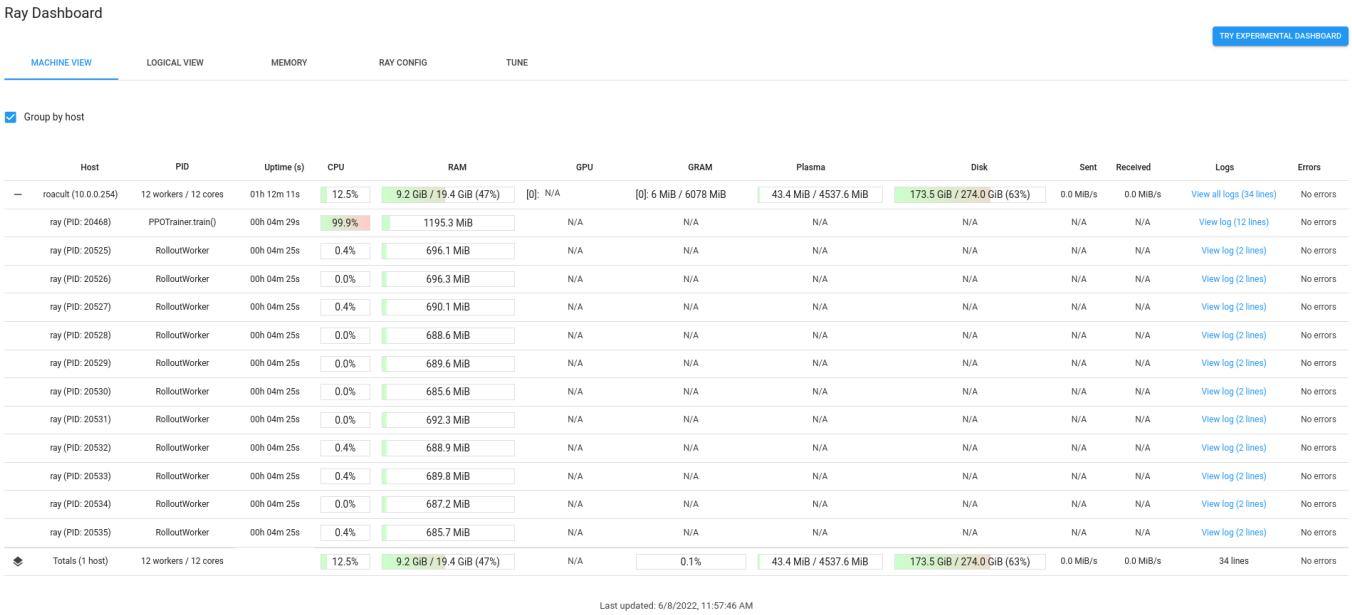


Figure 3.12: Ray dashboard

The training process is monitored by tensorboard to watch the progress of the training in real-time and helps identify the algorithm state (converging or diverging).

Figure 3.13: Tensorboard

3.4.2 Testing strategy

In this part, we will describe the decisions and steps taken to guarantee a good testing procedure and get the most accurate results possible.

Same as in training, testing is also done in the cloud server right after the training process is done. Figure 3.14 demonstrates the use of testing cli.

```
/opt/dev/trading/pfe/DRL-trading-bot | develop !10 ?3 ..... ✔ | crypto-venv Py
└─ ./optimize.py ray_test -a PPO -r simple_profit -data dataset/binance-BTCUSDT-1h-2019-01__2021-01.csv --add-indicators -id binance-BTCUSDT-1
h-lite_0_2022-05-13_17-53-55 --episodes 3 -ns 160 --render
2022-06-08 21:11:26,932 INFO services.py:1456 -- View the Ray dashboard at http://10.0.0.254:8265
2022-06-08 21:11:28,566 WARNING trainer.py:2503 -- You have specified 1 evaluation workers, but your `evaluation_interval` is None! Therefore,
 evaluation will not occur automatically with each call to `Trainer.train()`. Instead, you will have to call `Trainer.evaluate()` manually in
order to trigger an evaluation run.
2022-06-08 21:11:28,566 INFO ppo.py:268 -- In multi-agent mode, policies will be optimized sequentially by the multi-GPU optimizer. Consider s
etting simple_optimizer=True if this doesn't work for you.
2022-06-08 21:11:29,995 WARNING rollout_worker.py:498 -- We've added a module for checking environments that are used in experiments. It will
cause your environment to fail if your environment is not set upcorrectly. You can disable check env by setting `disable_env_checking` to True
 in your experiment config dictionary. You can run the environment checking module standalone by calling ray.rllib.utils.check_env(env).
2022-06-08 21:11:29,995 WARNING env.py:120 -- Your env doesn't have a .spec.max_episode_steps attribute. This is fine if you have set 'horizon
' in your config dictionary, or `soft_horizon`. However, if you haven't, 'horizon' will default to infinity, and your environment will not be
```

Figure 3.14: Testing reinforcement learning agent cli

- **-a**: Algorithm used to train the agent (EX: PPO, A2C ..).
- **-r**: Reward function used (EX: simple_profit, sharp, sortino).
- **-data** Dataset used to train.
- **–add_indicators**: This flag when set technical indicators are added to the observation space.

- **-lstm**: This flag wraps agent network with lstm layer.
- **-id**: Id of the model we want to test.
- **–episodes**: Number of episodes to test on.
- **-ns**: List checkpoints we want to test.
- **–render**: This flag trigger environment rendering.

The training process is schematized in Figure 3.15.

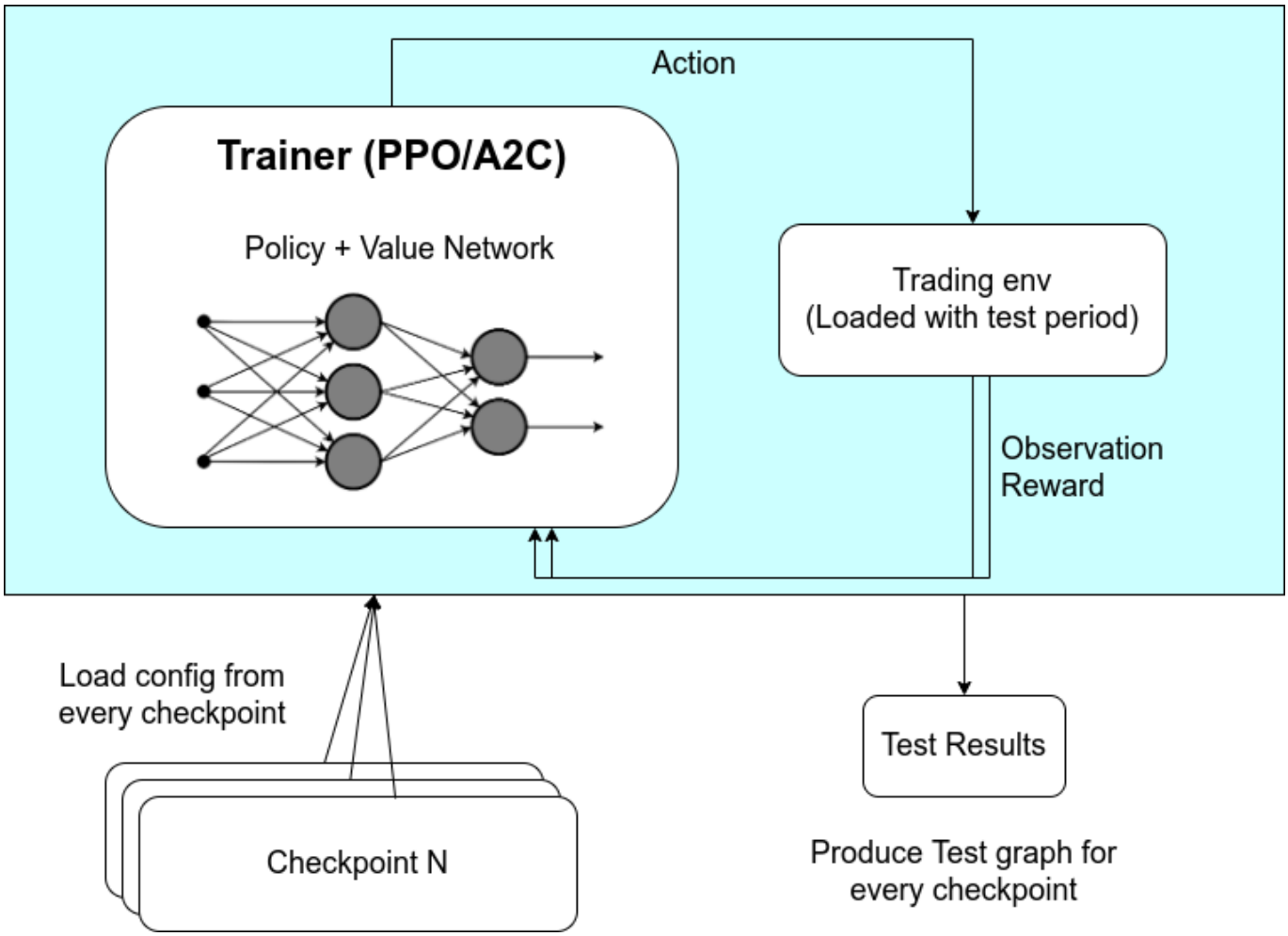


Figure 3.15: Testing process.

3.4.2.1 Benchmarks

When market prices are rising, even poorly timed trades may result in a profit; thus, net worth or earnings alone cannot be used to evaluate a trading strategy. To test, assess, and get a sense of how our model is doing, we required something to compare it to. After reviewing the relevant literature, we identified three trading strategies (benchmarks) that are often used as

evaluation measures. Following is a description of each benchmark used to analyze and test trading agents.

Buy and Hold (BH)

BH is a traditional passive investment approach, in which an investor purchases assets and holds them for a long period. Because some assets, such as stock shares for big companies, tend to increase in value over time (years, centuries), buying and holding for a long period often result in a profit. An investor who uses a BH strategy actively selects investments but has no concern for short-term price movements [6]. As the author of this study did [14].

Simple Moving Average (SMA) crossover

Simple Moving Average (SMA) crossover is a technique based on moving average indicators; it employs SMA indicators of different lengths (most often SMA 50 and SMA 200) and uses their crosses as a signal for entering or exiting trades. When the SMA 50 crosses above the SMA 200, it generates a buy signal (golden cross), and when the SMA 50 crosses beneath the SMA 200, it generates a sell signal (death cross).

Relative Strength Index (RSI) divergence

Relative Strength Index (RSI) divergence is a strategy based on RSI indicators. Generally, an RSI divergence means that the RSI indicator is moving in the opposite direction compared to the price. Therefore, while the price is moving, the RSI is telling us in advance to anticipate a change in the direction. There are two types of RSI divergence.

- **Positive RSI Divergence** When the price movement reflects lower lows and lower highs, but the RSI indicator is indicating the contrary, the price action is bearish. This divergence may function as a buy signal.
- **Negative RSI Divergence** It applies to positive trends characterized by higher closing highs and lows. This divergence could be used as a Sell signal.

3.4.2.2 Procedure

The following is the main important point followed to ensure a proper test and accurate results.

- Agents are evaluated independently throughout both the testing and training periods.
- Agents are evaluated on each saved checkpoint.
- Repeated runs of an agent on the same period might result in different actions each time; hence, multiple runs are essential to confirm the agent's stability and eliminate outlier results.
- Agent's accumulated net worth is plotted alongside the aforementioned benchmarks (BH, SMA crossover, and RSI divergence).
- Asset and the balance held over time are recorded and plotted on the same time ax.

Figure 3.16 shows an example of the test result of an agent trained on BNB-USDT pair for 190 iterations (loaded from checkpoint 190) using Sortino ratio reward function, the agent is evaluated on 3 episodes (runs). Each benchmark is plotted in a different color (BH green,...) and the agent's net worth in the 3 episodes is plotted in red.

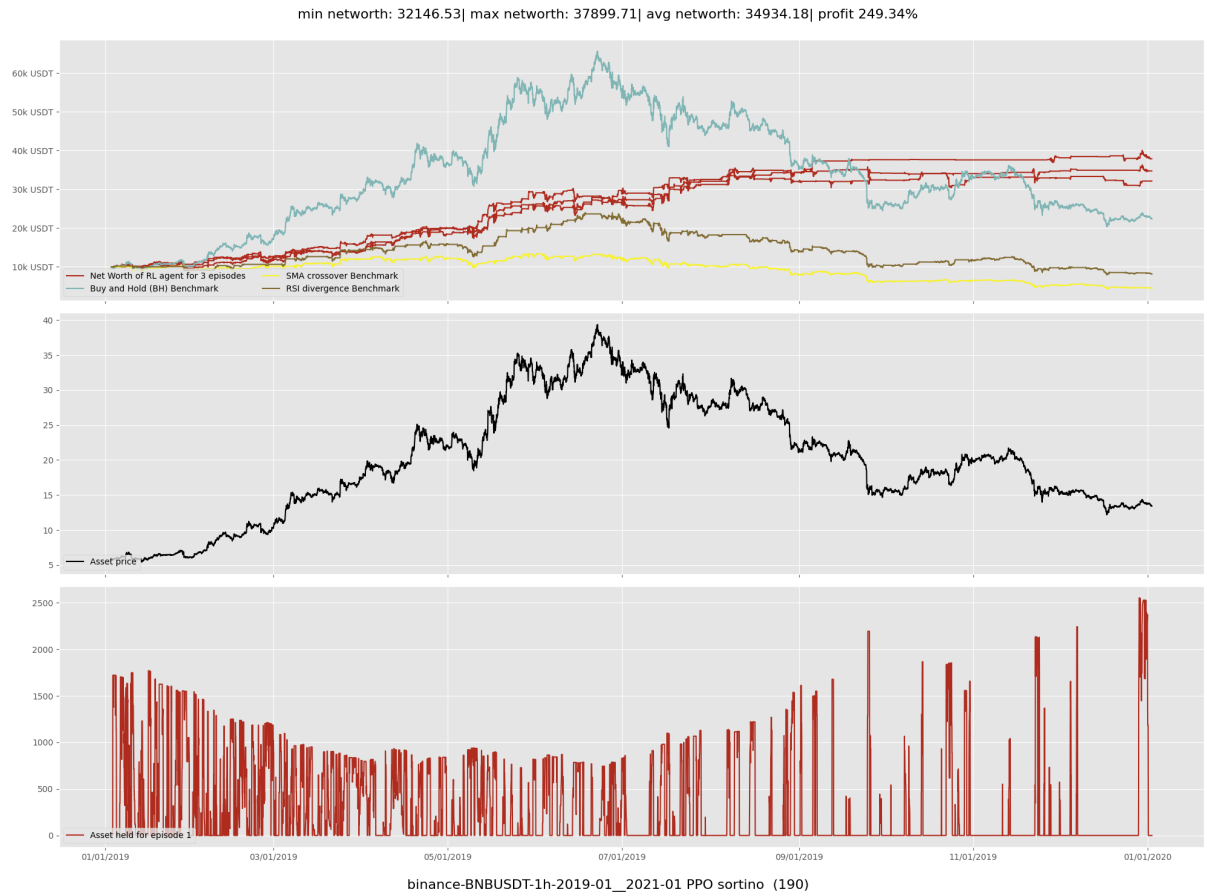


Figure 3.16: Test result example

CHAPTER 4

Results Analysis

As stated before, agents are used on an environment simulated with two years (2019-2021) of hourly historical OHLCV prices, the first year being used for training and the second year for backtesting.

The experiment was conducted on four currency pairs.

- **BTC-USDT**: Bitcoin and USDT
- **ETH-USDT**: Etheruim and USDT
- **BNB-USDT**: Binance coin and USDT
- **ADA-USDT**: Cardano coin and USDT

Each one of these pairs is combined with the three reward functions described before (Simple profit, Sharpe ratio, Sortino ratio). This gives us a total of 12 options, and each one of these options is tested using two distinct Deep Reinforcement Learning algorithms, A2C and PPO (explained previously in this section) which are the most promising Deep Reinforcement Learning algorithms as of the writing of this report.

4.1 Proximal Policy Optimization (PPO) agent

In this section, we will describe the results given by the PPO agent.

PPO Simple profit reward function

Table 4.1: Average profit of PPO agent and other benchmarks strategies.

Currency pairs	BH	SMAcrossover	RSIdivergence	PPOAgent
BTC-USDT	+ 319%	- 40%	- 19%	- 22%
ETH-USDT	+ 480%	- 59%	- 12%	+ 140%
BNB-USDT	+ 186%	- 56%	- 27%	- 53%
ADA-USDT	+ 455%	- 22%	- 1%	- 32%

As we can see from Table 4.1, "simple profit" reward function didn't manage to make a profit except on the ETH-USDT pair. And by looking at the test result in Figure 4.1, we can see that the profit made by the BH benchmark was three times more than the profit made by the PPO agent.

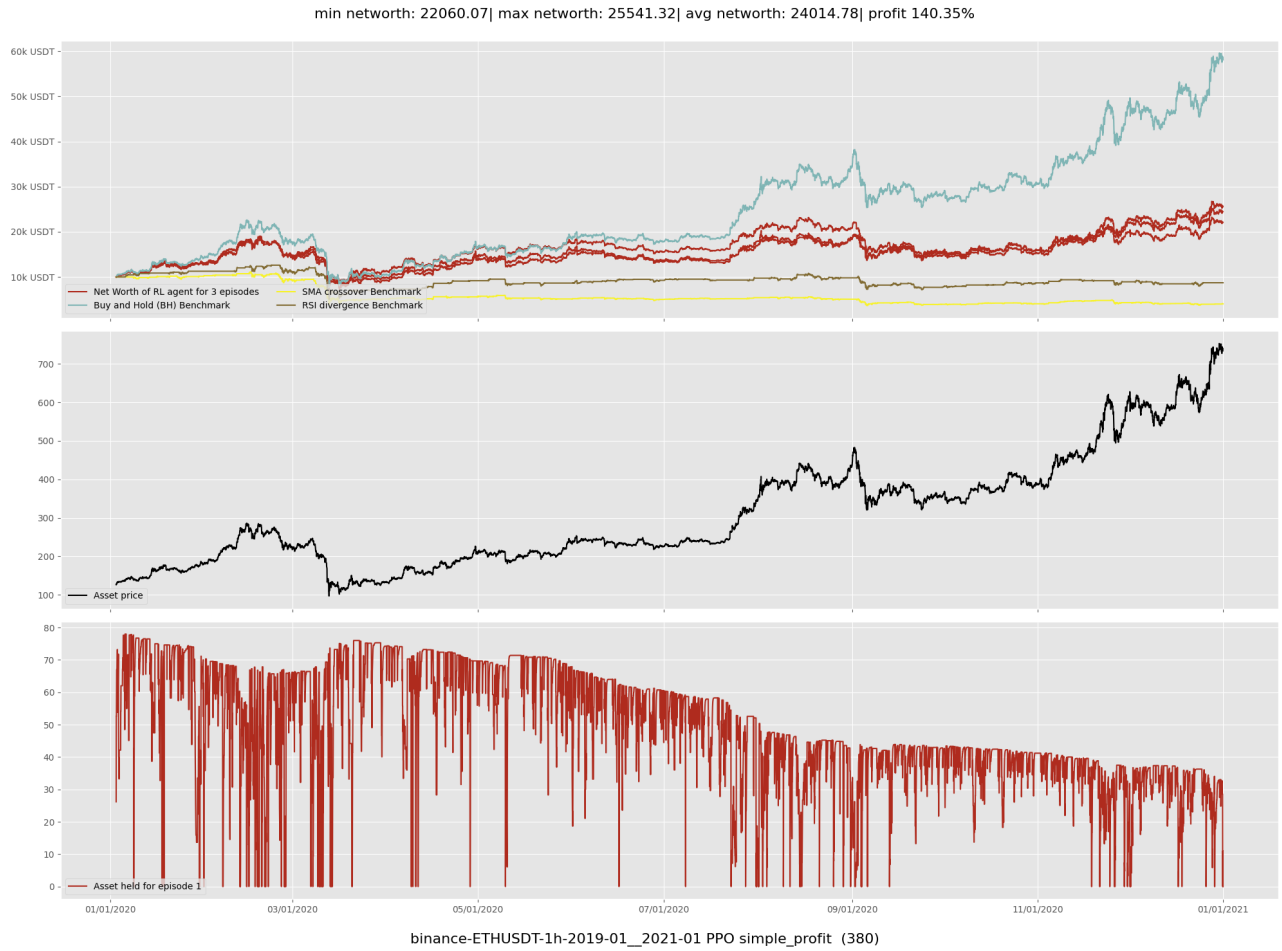


Figure 4.1: Test result of PPO agent, ETH-USDT pair, simple profit as a reward function.

PPO Sharp ratio reward function

Table 4.2: Average profit of PPO agent and other benchmarks strategies.

Currency pairs	BH	SMAcrossover	RSIdivergence	PPOAgent
BTC-USDT	+ 319%	- 40%	- 19%	+ 104%
ETH-USDT	+ 480%	- 59%	- 12%	+ 71%
BNB-USDT	+ 186%	- 56%	- 27%	+ 137%
ADA-USDT	+ 455%	- 22%	- 1%	- 37%

As we can see from Table 4.2, "Sharp ratio" reward manage to make a profit on all pairs except for ADA-USDT which made 37% loss from the initial balance. The most profitable pair was BNB-USDT, and by looking in Figure 4.2 we can see in some periods our agent manage to make more profits than BH benchmarks.

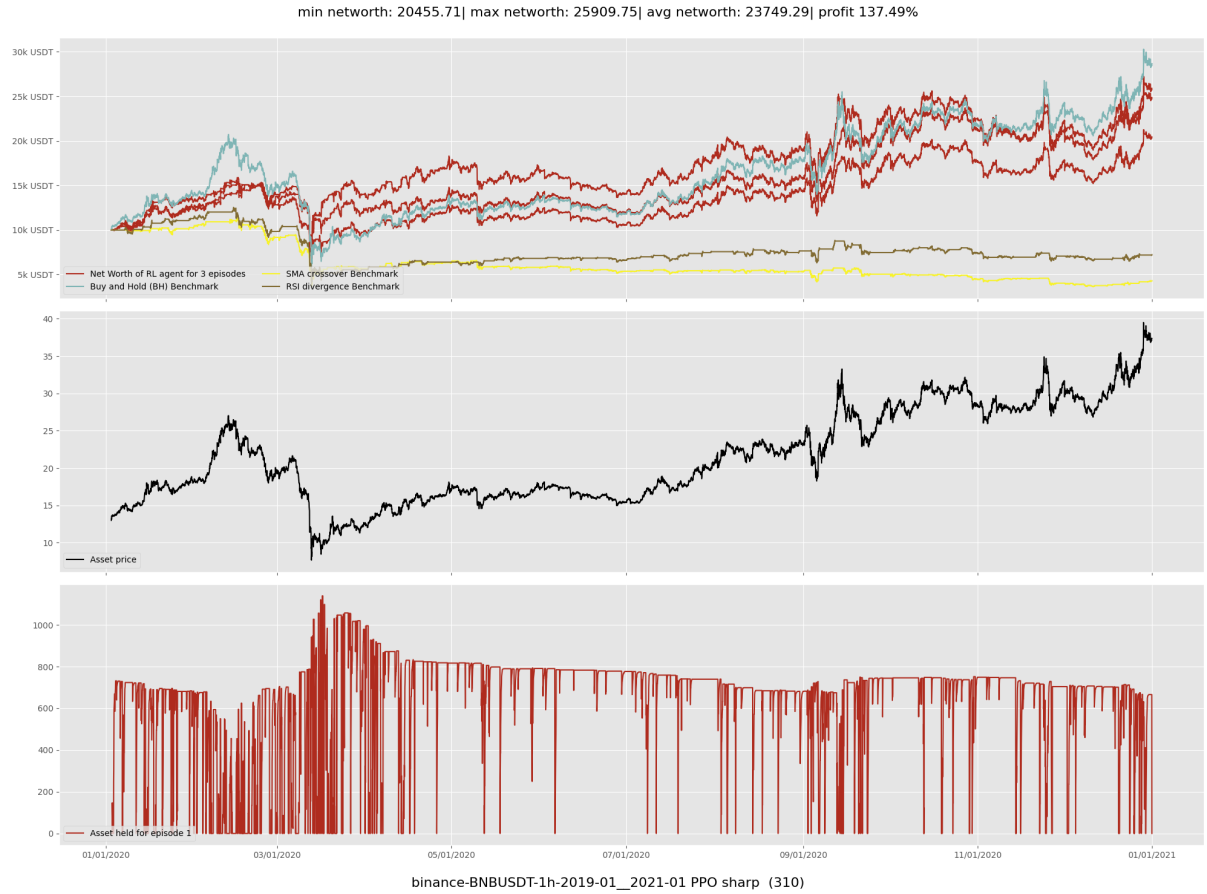


Figure 4.2: Test result of PPO agent, BNB-USDT pair, Sharp ratio as a reward function.

Figure 4.3 shows results of PPO agent with sharp ratio reward on BTC-USDT pair which managed to make 105% profit. Although profit was always less than the profit made by BH strategy.

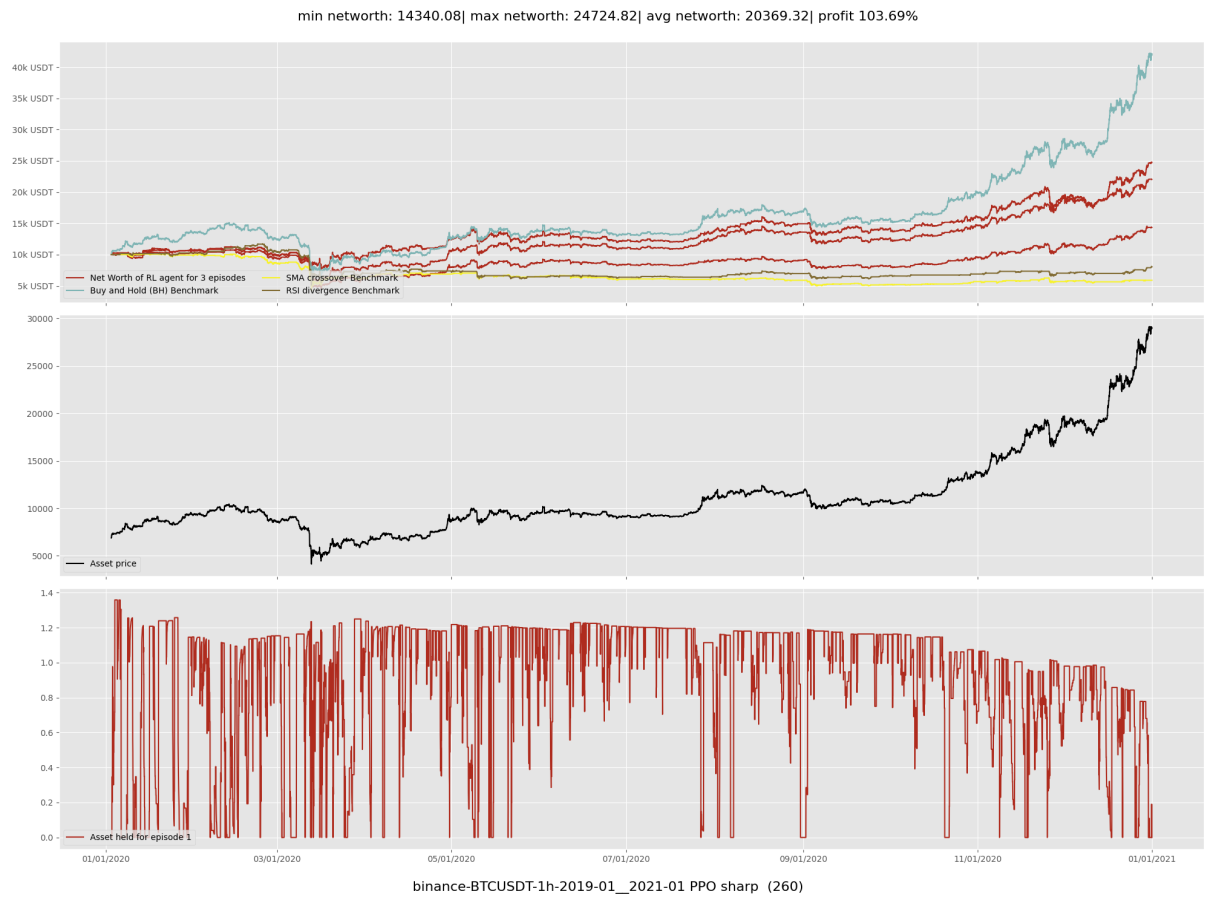


Figure 4.3: Test result of PPO agent, BTC-USDT pair, Sharp ratio as a reward function.

PPO Sortino ratio reward function

Table 4.3: Average profit of PPO agent and other benchmarks strategies.

Currency pairs	BH	SMAcrossover	RSIdivergence	PPOAgent
BTC-USDT	+ 319%	- 40%	- 19%	+ 47%
ETH-USDT	+ 480%	- 59%	- 12%	+ 57%
BNB-USDT	+ 186%	- 56%	- 27%	+ 152%
ADA-USDT	+ 455%	- 22%	- 1%	+ 67%

As we can see from Table 4.3, "Sortino ratio" reward manages to make a profit on all pairs, and same as in the "Sharp ratio" reward, pair BNB-USDT is the one with the highest profit where it made 152% profit. Agent performance on BNB-USDT pair over test period is shown in Figure 4.4.

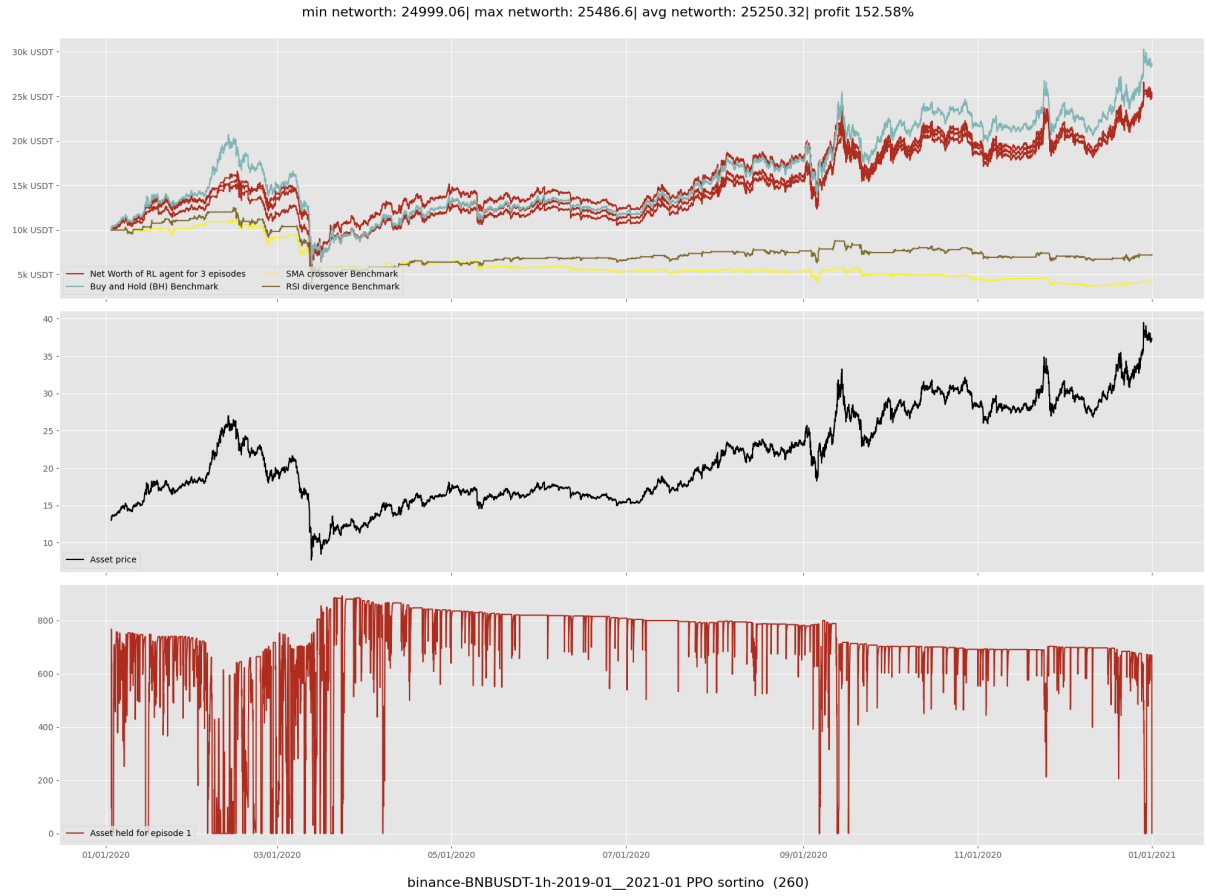


Figure 4.4: Test result of PPO agent, BNB-USDT pair, Sortino ratio as a reward function.

4.2 Advantage Actor Critic (A2C) agent

In this section, we will describe the results given by the A2C agent.

A2C Simple profit reward function

Table 4.4: Average profit of A2C agent and other benchmarks strategies.

Currency pairs	BH	SMAcrossover	RSIdivergence	A2CAgent
BTC-USDT	+ 319%	- 40%	- 19%	+ 94%
ETH-USDT	+ 480%	- 59%	- 12%	+ 143%
BNB-USDT	+ 186%	- 56%	- 27%	- 22%
ADA-USDT	+ 455%	- 22%	- 1%	+ 15%

From Table 4.4, we notice agent loss -22% of net worth in BNB-USDT pair, besides this pair agent manages to make profits but still lack behind BH benchmark, most profitable pair was ETH-USDT with 143% increase in net worth, Figure 4.5 shows agent performance over time on ETH-USDT pair.

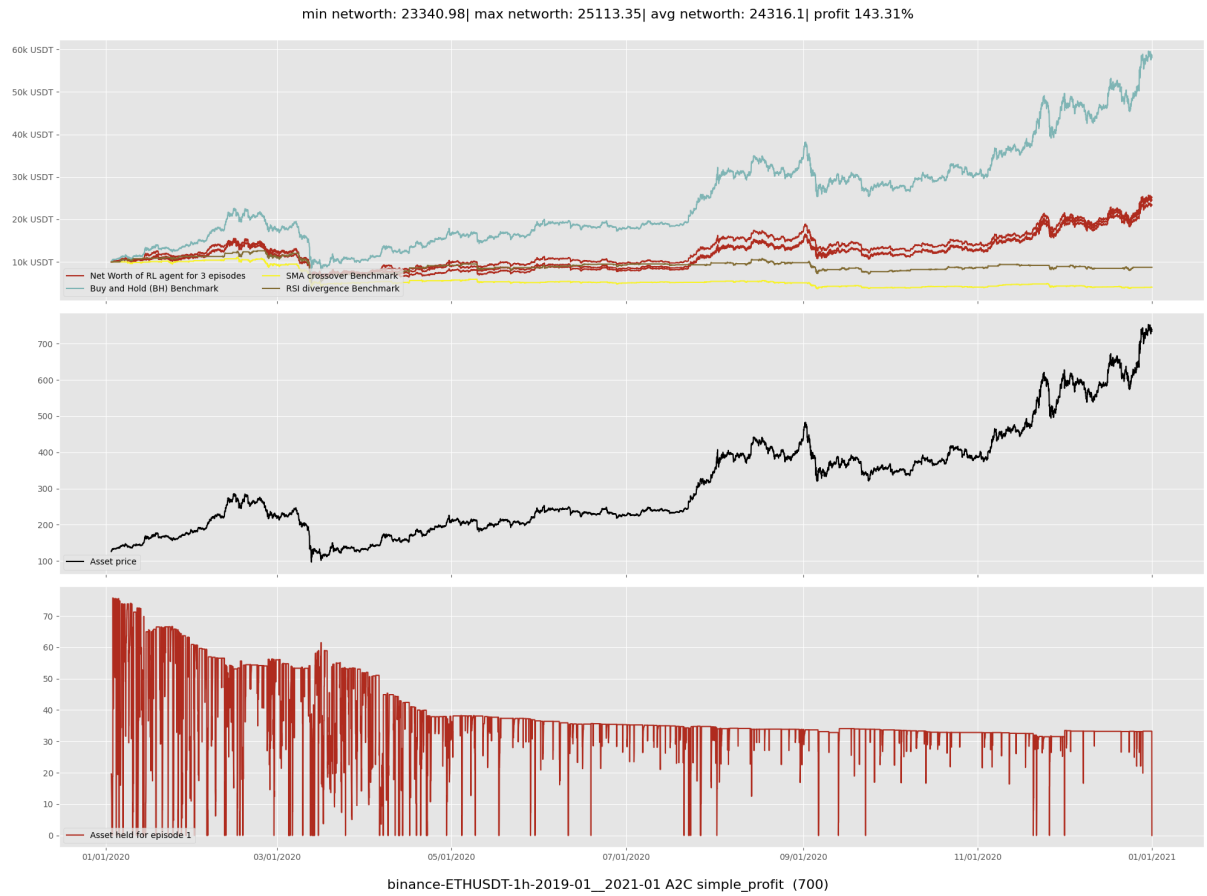


Figure 4.5: Test result of A2C agent, ETH-USDT pair, Simple profit as a reward function.

A2C Sharp ratio reward function

Table 4.5: Average profit of A2C agent and other benchmarks strategies.

Currency pairs	BH	SMAcrossover	RSIdivergence	PPOAgent
BTC-USDT	+ 319%	- 40%	- 19%	+ 105%
ETH-USDT	+ 480%	- 59%	- 12%	+ 210%
BNB-USDT	+ 186%	- 56%	- 27%	+ 191%
ADA-USDT	+ 455%	- 22%	- 1%	+ 247%

From Table 4.5 we can extract the following information's:

- A2C using "Sharp ratio" as the reward function is profitable on all pairs.
- ETH-USDT is the most profitable pair with 210% increase in net worth. From Figure 4.6, we can note that the agent chose to hold Asset for the majority of the time and sell when the market price began to fluctuate or change quickly in both directions (increasing or decreasing) to minimize risks.
- Most interesting results are for the BNB-USDT pair which manages to outperform the BH benchmark, Figure 4.7 shows agent performance over time.
- Figure 4.8 shows that the BTC-USDT agent outperformed the BH benchmark the most of the time, except for the last portion of time when prices started increasing rapidly agent sold all assets to minimize risks, which is a good approach overall.

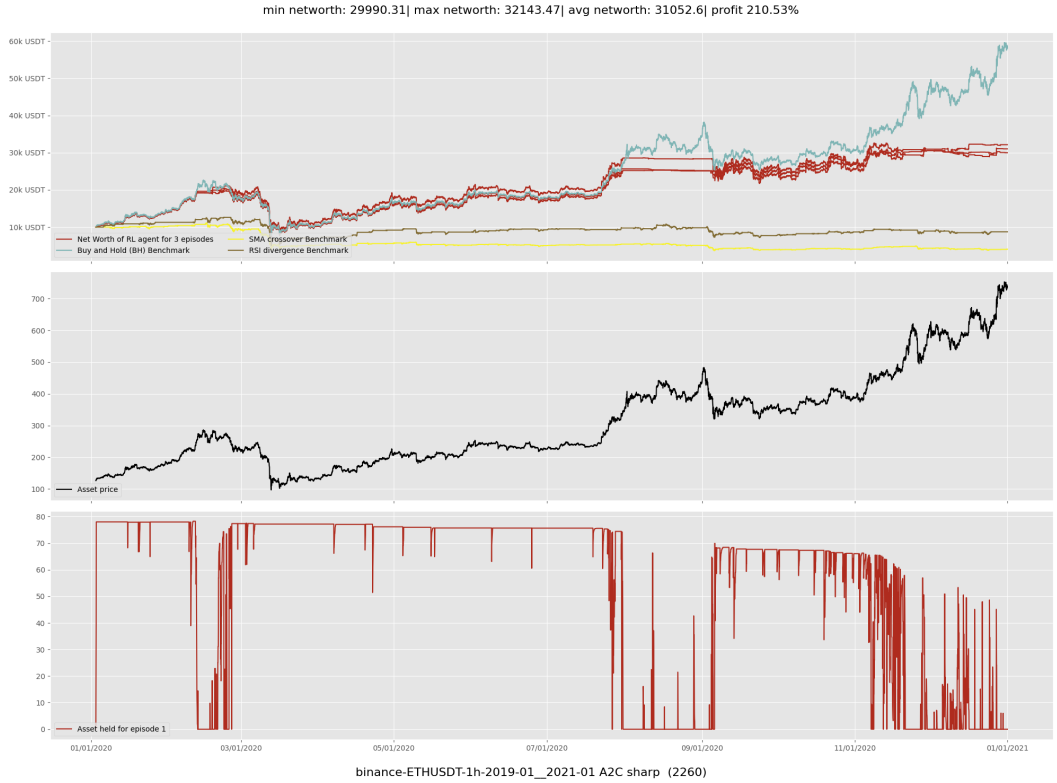


Figure 4.6: Test result of A2C agent, ETH-USDT pair, Sharp ratio as a reward function.

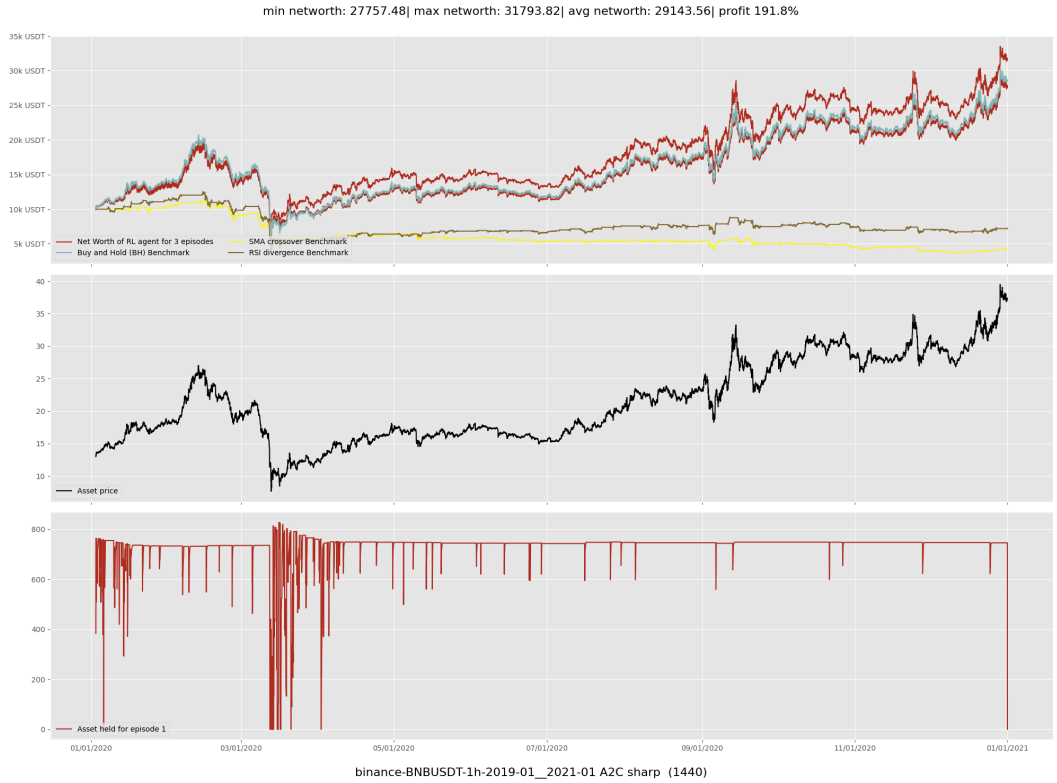


Figure 4.7: Test result of A2C agent, BNB-USDT pair, Sharp ratio as a reward function.

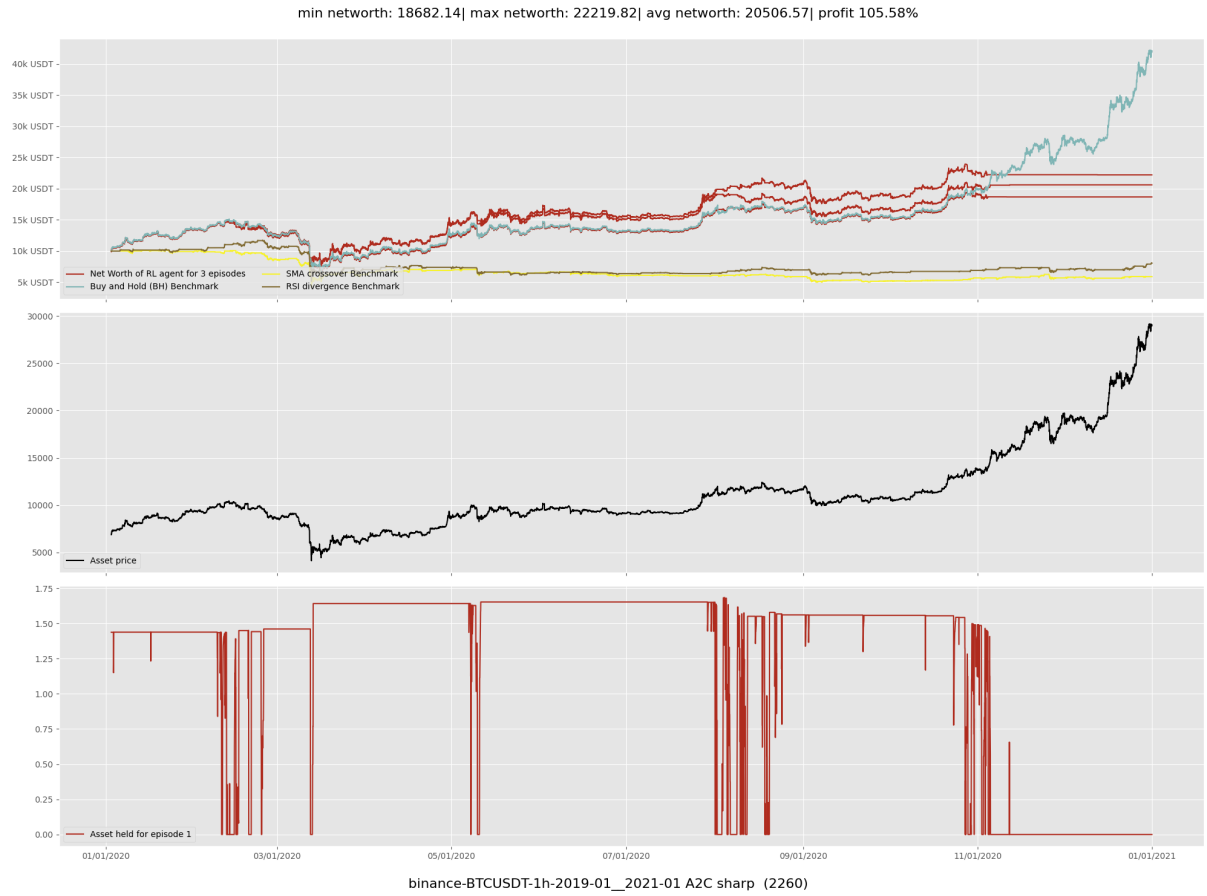


Figure 4.8: Test result of A2C agent, BTC-USDT pair, Sharp ratio as a reward function.

A2C Sortino ratio reward function

Table 4.6: Average profit of A2C agent and other benchmarks strategies.

Currency pairs	BH	SMAcrossover	RSIdivergence	PPOAgent
BTC-USDT	+ 319%	- 40%	- 19%	+ 106%
ETH-USDT	+ 480%	- 59%	- 12%	+ 69%
BNB-USDT	+ 186%	- 56%	- 27%	+ 159%
ADA-USDT	+ 455%	- 22%	- 1%	+ 45%

From Table 4.6, we can notice that all results were positive with small gains if we compare it to the BH benchmark, especially the ETH-USDT and ADA-USDT pairs which achieved almost 10 times less profit than the BH benchmark, most interesting results were in BNB-USDT pairs which manage

to make 159% profits achieving almost the same results as BH benchmark, Figure 4.9 shows the performance of A2C agent over time on BNB-USDT pair.

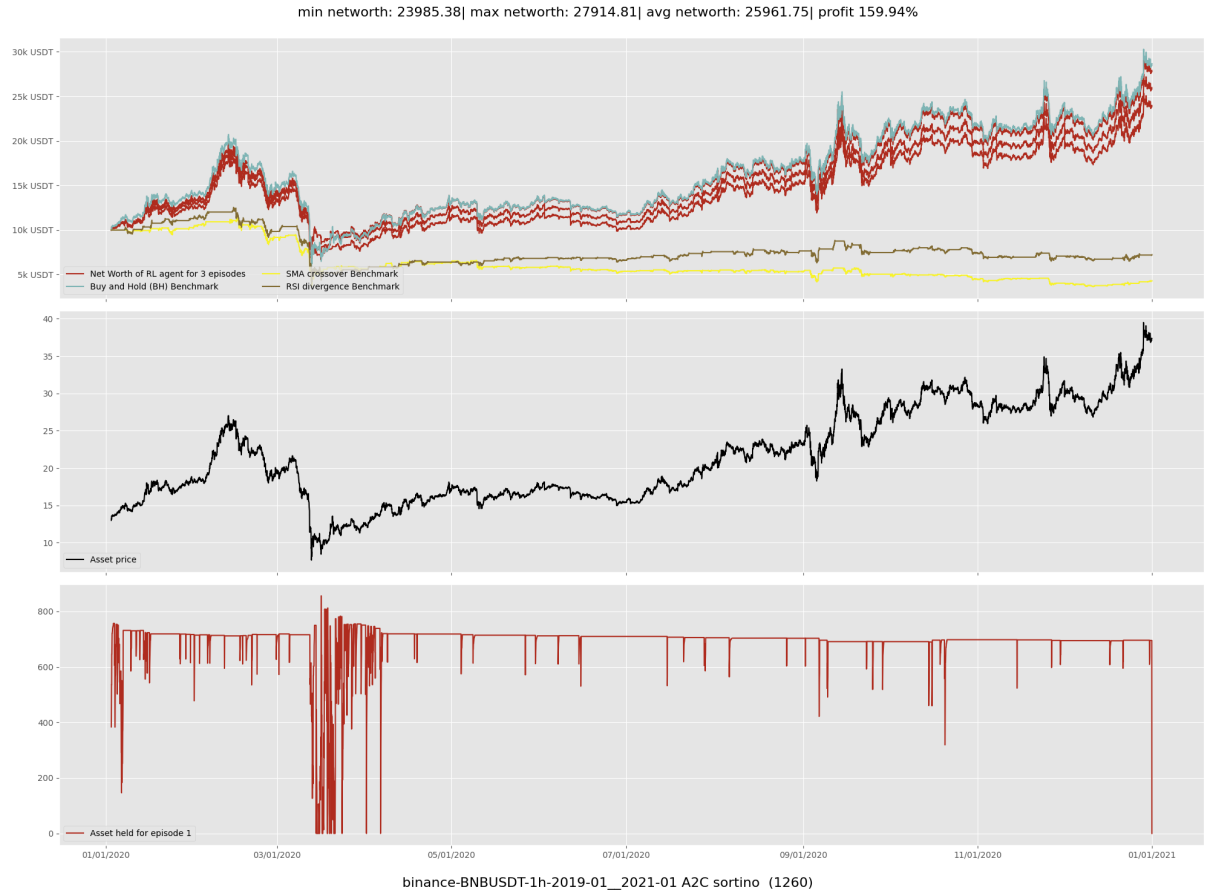


Figure 4.9: Test result of A2C agent, BTC-USDT pair, Sharp ratio as a reward function.

4.3 Conclusions drawn from results

From the results shown above, of both Advantage Actor Critic (A2C) and Proximal Policy Optimization (PPO) agents and with the three rewards (simple_profit, sharp ratio, Sortino ratio), and with the four cryptocurrency pairs (BTC-USDT, ETH-USDT, BNB-USDT, ADA-USDT), the following notes can be extracted.

- Non-risk-based reward functions such as simple profit didn't perform well and resulted in loss or small gain on most pairs and with both algorithms (PPO, A2C).
- From figures of simple profit functions results (Figure 4.1, Figure 4.5) we can note that the agent was buying and selling constantly and the

hold action was rarely used which indicates that the agent is always looking for small gains rather than overall net worth.

- Risk-based rewards such as sharp and Sortino ratio performed much better and most results were positive.
- We can note that the agent with the Sharp ratio reward function constantly buys and holds assets and start selling when prices fluctuate or change rapidly in both directions
- From the three benchmarks used, the BH benchmark was by far the most positive and profitable one, although some agents managed to outperform the BH benchmark.
- Based on the outcomes of these tests, A2C agents performed better than PPO agents.
- Based on the outcomes of these tests, the sharp ratio reward function with A2C agents is the most effective and promising solution. It consistently outperforms the BH benchmark (if we exclude market fluctuation in late 2020) and effectively avoids market swings while maintaining a steady profit over time.

CHAPTER 5

Back-testing platform

Before deploying a trading bot or strategy to live markets, intensive back-testing and simulations and advanced analysis by trading experts are required to identify beforehand any flaws or problems in the bot or strategy since any problem or wrong action can result in money loss. To easily simulate trades and test RL agent we built a backtesting web platform. The main features of this platform are listed below.

- List all trained agents in different checkpoints.
- Deploy any trained agent.
- Simulate trades over historical prices.
- Display agent net worth and actions over time in a visually and easily interpreted format.
- Display ratio of Balance/Asset over time in a visually and easily interpreted format.
- Display agent net worth over time in a visually and easily interpreted format.
- Pause and Resume simulations to easily analyze agent actions.

This platform consists of two components: API (Backend) and Web App (Frontend)

5.1 API (Backend)

API is responsible for deploying and testing trained models and communicating results to the frontend in real-time over WebSockets protocol since loading and launching trained agents is a major effort due to data retrieval and preparation, which demand substantial CPU and GPU processing capacity.

This part of the platform needs to be fast (performant), simple since it does not need a lot of features, and reliable because it would be handling a significant workload. Considering these criteria, the best option was the Python FastApi framework. Which checked all the requirements needed.

5.2 Web App (Frontend)

WebApp is accountable for displaying available agents and giving traders the ability to choose and deploy any trained agent. Provide an interpretable representation of agent behavior and performance over time. This part of the platform is developed using the famous web framework ReactJS.

5.3 Deployment

The platform was deployed in one of the Swissdigilab servers under the Swissdigilab domain name, it can be visited with this link:
http://rl-trading.swissdigilab.com

As shown in Figure 5.1, the platform was deployed using docker with three containers.

- **Api container** Contains FastApi app, served with UVICORN server.
- **Web container** Contains ReactJS app, served with "react serve" server.
- **Reverse proxy** This container is using Nginx as a reverse proxy (gateway) to provide access to the API and web app. It is the only container exposed to the outside.

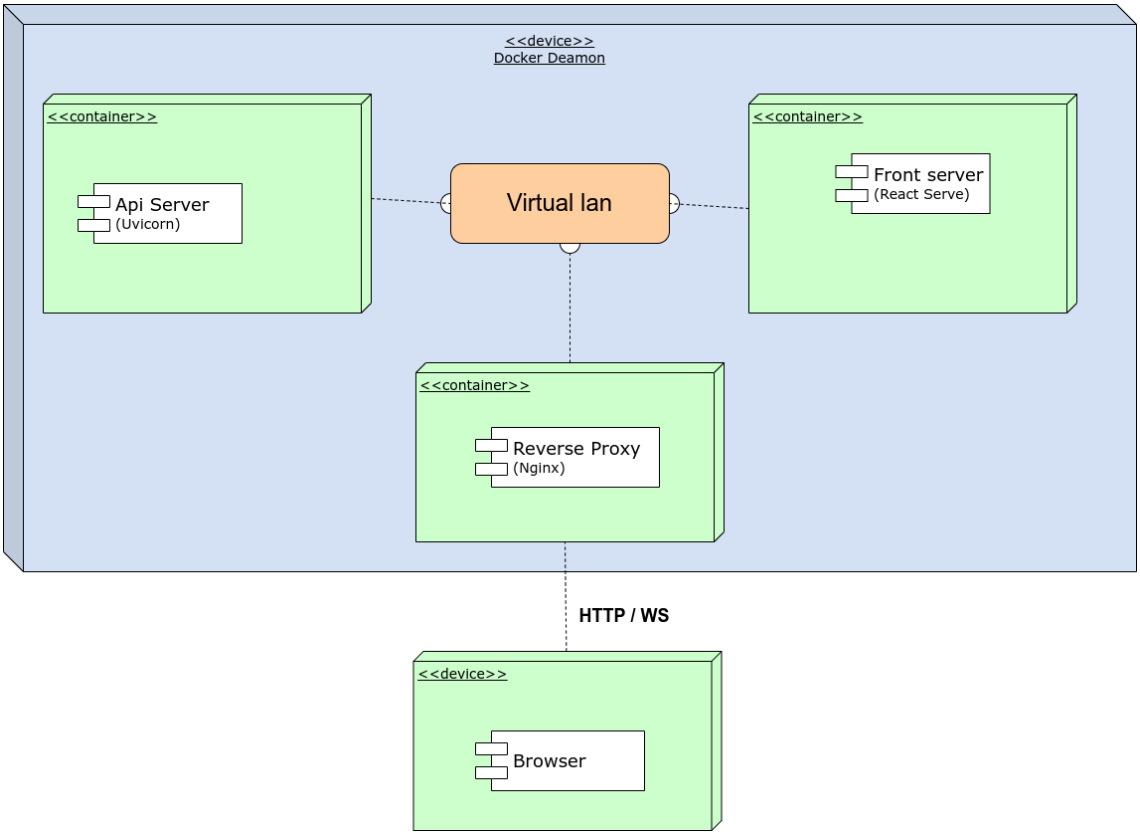


Figure 5.1: Deployment diagram.

Platform showcase

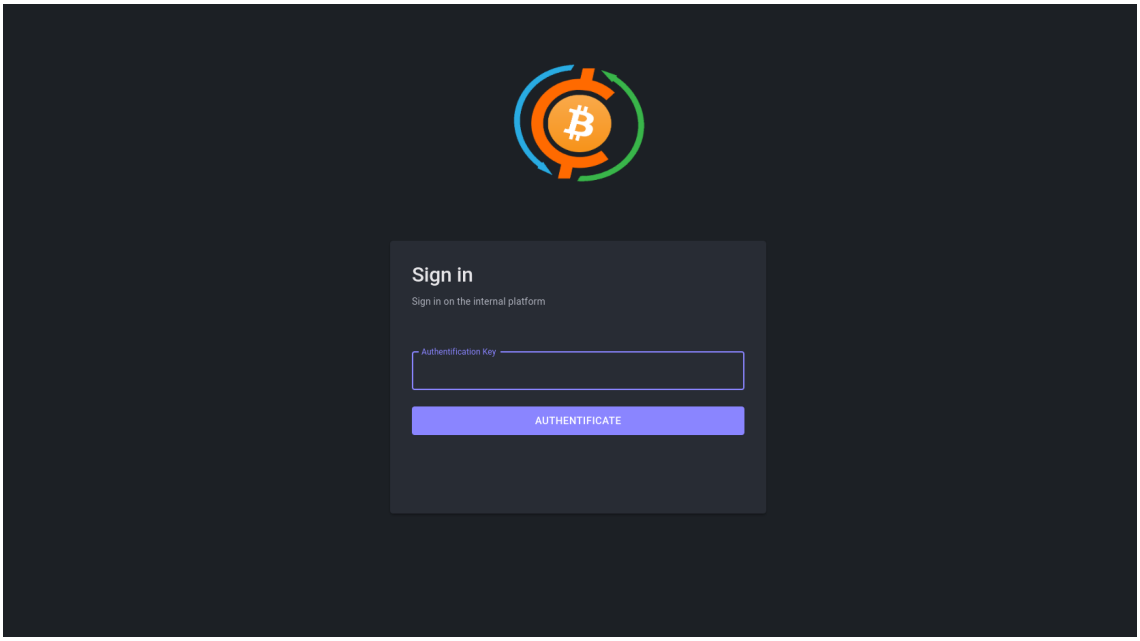


Figure 5.2: Authentications page.

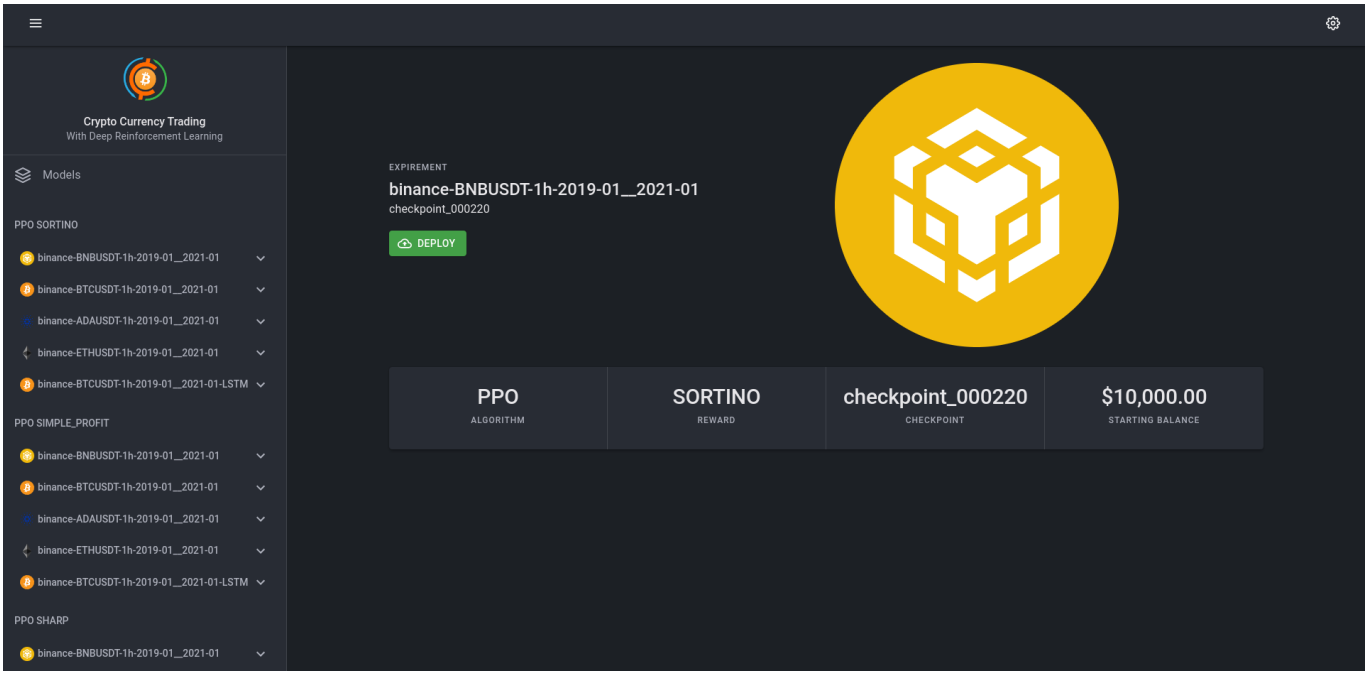


Figure 5.3: Agent listing page.

Figure 5.4: Monitoring deployed agent dashboard.

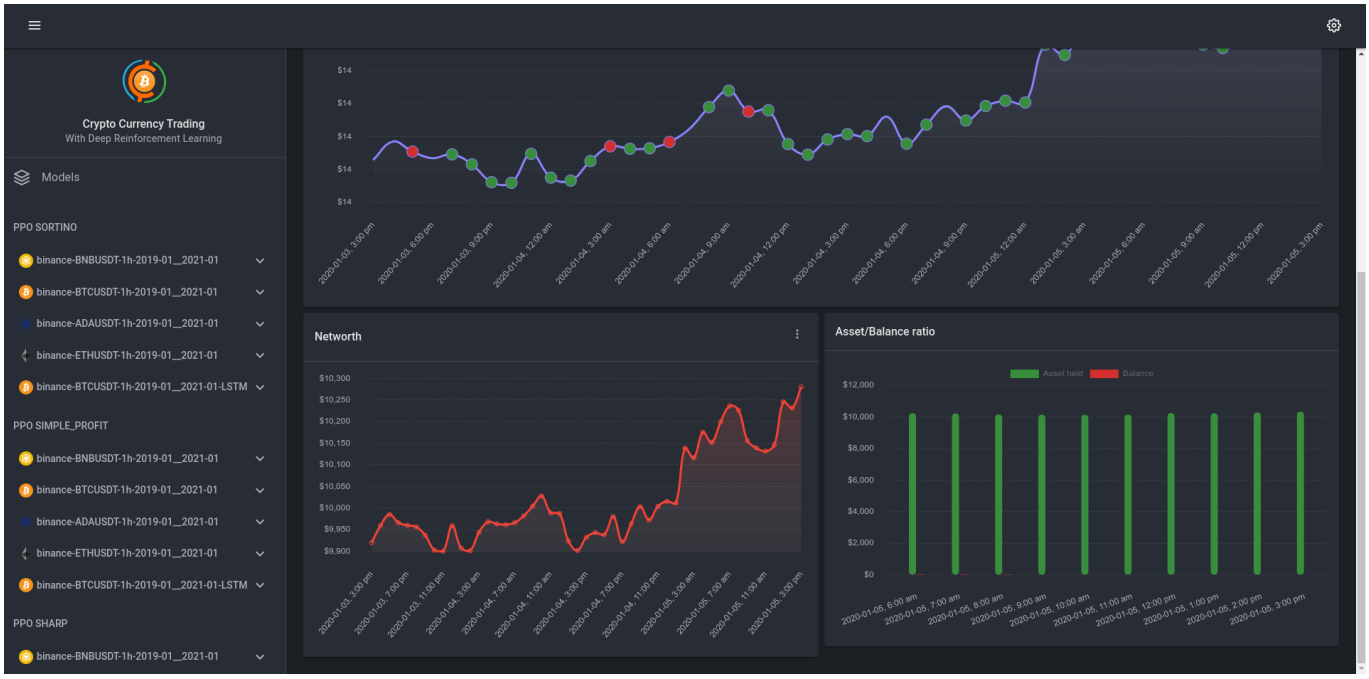


Figure 5.5: Monitoring deployed agent dashboard.

CHAPTER 6

Project management

6.1 Collaborative Work

To implement any project and ensure its success, collaborative work is a must. Parallel tasks are also a great way to save time and speed up the completion of a project, but it requires tools to organize and make sure everything's in order. Thus, we will discuss these tools and project monitoring in this chapter.

6.1.1 Tools

Git

Git is a free and open-source distributed version control system designed to efficiently manage small to extremely big projects. Git is simple to understand, has a small footprint, and offers lightning-fast speed. It surpasses SCM solutions like Subversion, CVS, Perforce, and ClearCase with capabilities such as inexpensive local branching, practical staging zones, and various workflows [1].

[1] https://git-scm.com

Git requires a server to host the source code, to be able to share and collaborate with others. Many services provide hosting git repositories such as GitHub or GitLab. Swissdigilab has provided us with private repo on the GitLab platform.

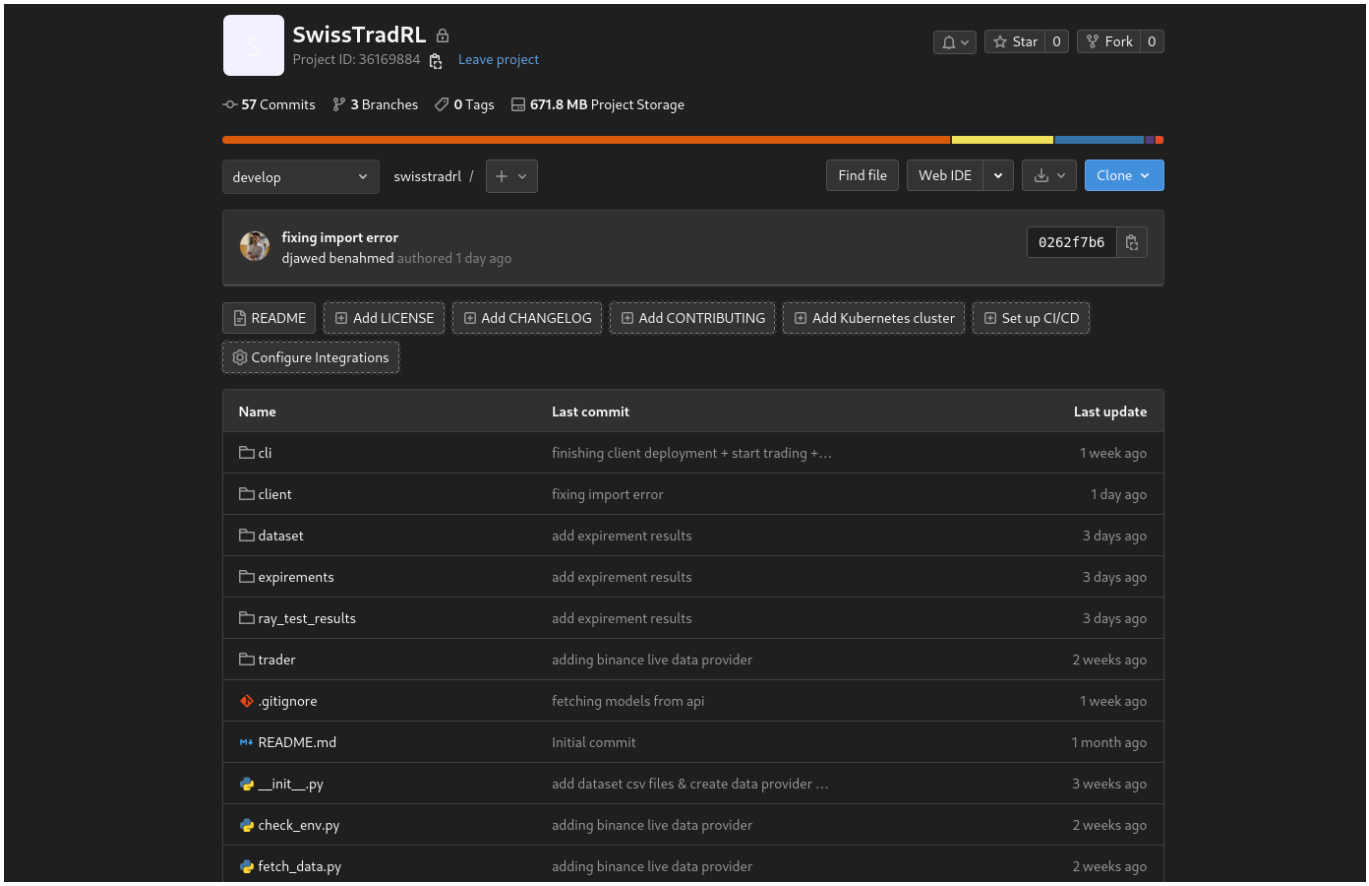


Figure 6.1: Git repository on Gitlab.

If we count agent size together with all of the checkpoints and with different reward functions, the amount of storage space that will be necessary will be more than what can be sent over email or via any of the other communication platforms. To distribute these models, we need a storage service, and after giving it some thought, we've opted to utilize Google Drive as our storage provider

Google drive

Google Drive is a service established by Google for the storing and synchronization of files. Google Drive, was introduced on April 24, 2012, it enables users to store files in the cloud (on Google's servers), synchronize files across devices, and share files [2].

[2]https://www.google.com/drive

6.1.2 Project monitoring

This project was monitored by two entities: the National School of computer science of Sidi Bel Abbes (ESI-SBA), represented by our instructor (Mr. KHALDI Belkacem), and Swissdigilab company, represented by the project's supervisor (Mr. ALAOUI MDAGHRI Abdellah).

Using Google Meet, the pedagogical supervisor (Mr. Khaldi) monitored all of the completed and upcoming assignments, the results produced, and the theoretical arguments behind them. The ESI-Sba supervisor was also responsible for validating the primary guidelines, and the theoretical concepts, proofreading and validating the thesis, and ensuring that the work done complied with the ESI-Sba standards. And he was always available providing us with guiding and answering our questions regarding this project.

Mr. ALAOUI MDAGHRI Abdellah, our supervisor from Swissdigilab, was in constant contact with us. We held weekly meetings to display and explain our work via Discord. He also provided us with the tools we required such as servers and domain hosting and advised us in areas where we lacked experiences, such as finance and trading.

6.2 Project Management Methodology

At the beginning of each project, it is necessary to choose a management method to follow that allows the project to be supervised to meet deadlines and achieve the objectives set. And when it comes to project management there are two options conventional ones like PMP and Prince 2, as well as agile approaches like SCRUM, Kanban, and DSDM.

The main goal of project management methodology is to be able to standardize, structure, and organize work methods. This helps keep all projects on the same track and lets us repeat what works and learn from what doesn't, which leads to a process of continuous improvement. In other words, a management methodology is a great tool for making things more efficient.

Figure 6.2: Project Management Methodology.

We went with Incremental Project Management Methodology, for certain reasons we will describe some of the themes.

- It's an ideal methodology when the project details are not clear at the beginning, which is the case in this project.
- This approach is more flexible and easy to change scope and requirements based on research findings.
- This project is about machine learning, which needs a step-by-step approach (build, train, test, build, train, test, build, etc.), which fits perfectly with the incremental methodology.
- This method works well when you want to explore a new technology, which is what this project is doing.

Conclusion

Despite the fact that deep reinforcement learning is a relatively new branch, it has had a lot of success, particularly in the financial and trading worlds, thanks to the rise of deep learning and years of research into the fundamental theories of reinforcement learning.

An experiment done in this project shows that reinforcement learning models can learn to trade and invest and make profits even in the chaotic and unstable market of crypto-currency. And all results shown pave the way for more research and experimentation that could one day transform the trading sector and lead this industry to become fully automated

6.3 Future work

Despite numerous tests and experiments conducted in this project, we only scratch the surface and still, a lot more experiments could be done and tested. There are thousands of cryptocurrencies in the crypto market right now and there are dozens of state-of-the-art reinforcement learning algorithms and there are many other options that could be changed and tuned, all this opens the door for more research to be conducted and explored.

One of the interesting ways that need to be explored is combining time series forecasting in the crypto market with a deep reinforcement learning agent in order to provide a price prediction which has a big potential of improving trading strategy.

Exploring the crypto futures market, where traders can profit from falling prices by employing Short and Long bets instead of Buy and Sell.

Bibliography

[1] Reinforcement learning : Markov-decision process towardsdata-science.com.

[2] Charu C. Aggarwal. *Neural Networks and Deep Learning*. Springer, Cham, 2018.

[3] Roohollah Amiri, Hani Mehrpouyan, Lex Fridman, Ranjan K Mallik, Arumugam Nallanathan, and David Matolak. A machine learning approach for power allocation in hetnets considering qos. In *2018 IEEE international conference on communications (ICC)*, pages 1–7. IEEE, 2018.

[4] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

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

[6] Brian Beers. Buy and hold definition. https://www.investopedia.com/terms/b/buyandhold.asp.

[7] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[8] biocompare. Molecular details of brain injury revealed.

[9] Yash Bohra. The challenge of vanishing/exploding gradients in deep neural networks.

[10] capitalindex team. Different types of trading strategies. https://www.capitalindex.com/bs/eng/pages/trading-guides/different-types-of-trading-strategies/.

[11] John Carpenter. A guide to company shares. https://www.1stformations.co.uk/blog/a-guide-to-company-shares/.

[12] James Chen. Foreign exchange (forex). https://www.investopedia.com/terms/f/foreign-exchange.asp.

[13] James Chen. Forex market. https://www.investopedia.com/terms/forex/f/forex-market.asp.

[14] Lin Chen and Qiang Gao. Application of deep reinforcement learning on automated stock trading. In *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, pages 29–33. IEEE, 2019.

[15] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[16] R.W. Colby and T.A. Meyers. *The Encyclopedia of Technical Market Indicators*. Dow Jones-Irwin, 1988.

[17] B.J. Copeland. artificial intelligence. https://www.britannica.com/technology/artificial-intelligence/.

[18] corporate finance institute team. Financial markets.

[19] Pamela de la Fuente Dayana Yochim, Dayana Yochim. A guide to company shares. https://www.nerdwallet.com/article/investing/stock-trading-how-to-begin.

[20] Sebastian Dittert. Reinforcement learning: Value function and policy medium.com.

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[22] Kevin Gurney. *An Introduction to Neural Networks*. Taylor and Francis, Inc., USA, 1997.

[23] Ikhlaas Gurrib et al. Performance of the average directional index as a market timing tool for the most actively traded usd based currency pairs. *Banks and Bank Systems*, 13(3):58–70, 2018.

[24] Adam Hayes. How does the stock market work? https://www.investopedia.com/articles/investing/082614/how-stock-market-works.asp.

[25] Adam Hayes. Physical asset. https://www.investopedia.com/terms/p/physicalasset.asp.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[27] Debasish Kalita. A brief overview of deep reinforcement learning.

[28] Yuxi Li. Deep reinforcement learning: An overview. *CoRR*, abs/1701.07274, 2017.

[29] 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. *arXiv preprint arXiv:1509.02971*, 2015.

[30] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11–26, 2017.

[31] Wei Luo, Qirong Tang, Changhong Fu, and Peter Eberhard. Deep-sarsa based multi-uav path planning and obstacle avoidance in a dynamic environment. In *International Conference on Swarm Intelligence*, pages 102–111. Springer, 2018.

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

[33] Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

[34] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In

International conference on machine learning, pages 1928–1937. PMLR, 2016.

[35] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[36] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[37] nirmalbang team. Trading vs. investing. https://www.nirmalbang.com/knowledge-center/trading-and-investing.html.

[38] Christopher Olah. Understanding lstm networks.

[39] Kayur Patel, James Fogarty, James A Landay, and Beverly Harrison. Investigating statistical machine learning as a tool for software development. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 667–676, 2008.

[40] David Petersson. supervised learning. https://www.techtarget.com/searchenterpriseai/definition/supervised-learning/.

[41] Mary K. Pratt. unsupervised learning. https://www.techtarget.com/searchenterpriseai/definition/unsupervised-learning/.

[42] Tidor-Vlad Pricope. Deep reinforcement learning in quantitative algorithmic trading: A review. *CoRR*, abs/2106.00123, 2021.

[43] Jonathan Sadighian. Extending deep reinforcement learning frameworks in cryptocurrency market making. *arXiv preprint arXiv:2004.06985*, 2020.

[44] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[45] Semyon Sergunin. How rpa and machine learning address business use cases. https://www.automationanywhere.com/company/blog/rpa-thought-leadership/how-rpa-and-machine-learning-address-business-use-cases/.

[46] William F Sharpe. Mutual fund performance. *The Journal of business*, 39(1):119–138, 1966.

[47] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

[48] smartera3s. Customer micro segmentation. https://www.smartera3s.com/products/customer-segmentation/.

[49] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, zweite edition, 2018.

[50] THE INVESTOPEDIA TEAM. 7 technical indicators to build a trading toolkit. https://www.investopedia.com/top-7-technical-analysis-tools-4773275/.

[51] The Editors of Encyclopedia Britannica. *stock exchange*. September 2021.

[52] Mike Thomas. How ai trading technology is making stock market investors smarter. https://builtin.com/artificial-intelligence/ai-trading-stock-market-tech/.

[53] William Wai Him Tsang, Terence Tai Leung Chong, et al. Profitability of the on-balance volume indicator. *Economics Bulletin*, 29(3):2424–2431, 2009.

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

[55] 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.

All links in the bibliography were accessed between 2022-02 and 2022-06; if any of the links are dead, you can access them using the web archive at https://web.archive.org