Reinforcement Learning applied to finance

by

Amit Dutta
16101100
Md Sultan Parvez
16101079
Partho Talukdar
16101095

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
October 2020

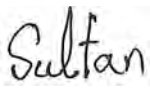
Declaration

It is hereby declared that

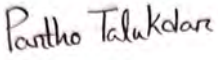
1. The submitted thesis is original solid work as a mandatory part for obtaining degree at Brac University.
2. The thesis material is not published or written by a third party,while providing due credit to the stakeholder with appropriate citation through full and accurate referencing.
3. The thesis or its any content material is not used or accepted for accomplishing any other degree or diploma at a university or other institution.
4. Proper acknowledgement of all main sources of help.

Student's Full Name & Signature:

Amit Dutta
16101100



Md Sultan Parvez
16101079



Partho Talukdar
16101095

Approval

The thesis/project titled "Reinforcement Learning Applied To Finance" submitted by

1. Amit Dutta (16101100)
2. Md Sultan Parvez (16101079)
3. Partho Talukdar– (16101095)

Of Summer, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on October 5, 2020.

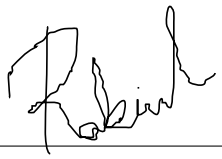
Examining Committee:

Supervisor:
(Member)

Prof. Mahbub Majumdar
Chairperson
Dept. of Computer Science & Engineering
Brac University

Mahbubul Alam Majumdar, PhD
Professor and Dean, School of Data and Sciences
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)



Md. Golam Rabiul Alam, PhD
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Prof. Mahbub Majumdar
Chairperson
Dept. of Computer Science & Engineering
Brac University

Mahbubul Alam Majumdar, PhD
Professor and Dean, School of Data and Sciences
Department of Computer Science and Engineering
Brac University

Abstract

The purpose of this work is to create an agent that can trade efficiently in the stock market. There is an implementation,proximal policy optimization (PPO) to train the agent and OpenAIGym to simulate a finical Market environment. The biggest problem of the trading market is there is no specific trading strategies, more often investor focuses on the risk and thus it becomes more of gambling. The deep learning community find research on Financial market less interesting because of the difficulty and the expensive nature of financial market. Main goal is to introduce a trading model using Reinforcement learning and neural network. The model will create a better solution of the current anomaly. The process gives confidence that this model will help the investor to find a safe yet profitable strategy.

Keywords: Machine learning, Reinforcement learning, OpenAiGym, Proximal Policy Optimization, Trading, Indicators , Sharpe Ratio , Sortino Ratio, Trading Indicators, RNN, LSTM, ANN, DNN

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our co-advisor Mahbubul Alam Majumdar,PhD sir for his kind support and advice in our work. He helped us whenever we needed help.
Thirdly, Md. Golam Rabiul Alam,PhD sir and the whole Thesis Coordinator panel of Summer2019 and Fall2019. All the reviews they gave helped us a lot in our later works.
And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

Chapter 1

Introduction

The trade market is the most important way of raising money for companies. The trade market is the basic idea of marketing a product through the value chain. Market volatility is perhaps the most misunderstood concept in investing. If the market is relatively stable, then it is low-risk volatility, but if it is changing now and then it is called high-risk volatility. Any types of investor can be benefited from high-risk volatility. However, conservative traders want a more stable market place. As the price fluctuates more, it is possible for an investor to buy share of the company, and with the company's growth, it is easier for them to profit.
Reinforcement learning is an area of machine learning which studies how an agent will take actions in an environment against a given set of states, thus maximizing the cumulative reward. In the operations research and control literature, reinforcement learning is called approximate dynamic programming, or neuro-dynamic programming. The main difference of Reinforcement learning and supervised learning is that supervised learning need to feed with many data and all those data have the correct action within them. On the contrary, the reinforcement agent decides what to do to perform the given task. In the absence of training data set, it's bound to learn from its experience. Open AI Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports various kind of environments. With the help of neural network sequence of timestep is followed to achieve an expected value point. Each timestep, the agent chooses an action, and the environment returns an observation and a reward. Stable Baselines is a python library, it is a set of improved implementations of Reinforcement Learning (RL) algorithms based on OpenAI Baselines. It has differences from the OpenAI Baselines. Although it is just a branch of OpenAIBaselines, there is significant structural refactoring in Stable Baselines. This toolset is well documented and comes with more code coverage. In trade market indicators are used for understanding the condition of finical or economic trends. In a real-life scenario, these indicators help an expert to understand the nature of the market. They often take many exciting decisions based on these indicators. There are many kinds of indicators − economic indicators, technical indicators, break-down indicators. Each indicator uses ratios and formulas that explain current gains and losses from a different perspective. Sharpe ratio is a technique used for understanding the risk of the trade market. The Trade market is very volatile, so it is essential to understand how much risk is taken. Sharpe ratio calculates the average return earned in more than the risk-free rate per unit of volatility or absolute risk.

$$SharpeRatio = (Rp–Rf)/\sigma p$$

Sortino ratio is another technique used for the same cause, but this technique calculates the risk in a different approach. It is a modification of Sharpe but penalizes only those returns falling below a user-specified target or required rate of the return. It computes the risk-adjusted return of an investment asset, portfolio, or strategy.

$$\text{Sortino Ratio} = (\text{R} - \text{T}) \ / \ \text{DR}$$

$$DR = \sqrt{\int_{-\infty}^{T} (T-r)^2 f(r) dr}$$

The created agent will enable the investor to find out strategies for maximizing profit. OpenAIGym is used to create a financial market environment. Then there is a use of a neural network consisting of an input layer followed by two 64 hidden layers and finally an output layer. At first, the effort was to implement the A2C algorithm but not satisfied with the result and experienced some major shortcomings. Then the PPO algorithm is applied over the environment and achieved a satisfactory solution.

1.1 Motivation

Much research has been conducted on making new strategies and techniques to find or build an Agent, that can efficiently trade in the market. The stock market has always been the centre of attention for finical market since it was first introduced in the early 16th century. Beyond a shadow of a doubt, it is basic to think about creating a trade Market Agent. But it is much more essential to create an Agent that can survive the worst of conditions. The majority of the existing model are based on Supervised learning. The limitation of those model are it can only match the performance of the existing human model. This also means that the models can be influenced by human made rules. It is already mentioned how volatile the stock market is. So, it is very normal for an Agent not to profit even though every condition suggested there will be profit. To overcome this major issue, the thought of creating an Agent that will not be subject to these sorts of influence. It may be possible to create such an Agent with the help of Reinforcement learning. Use of different Reinforce Learning Algorithm and compare them based on the result can be fruitful. The understanding of Market indicators will help greatly Since these indicators can detect the pace and nature of the Market and trade efficiently.

1.2 Problem Statement

The Stock Market is full of opportunities. But there have always been problems. The trade market can be influenced by all kinds of things. Weather, Political Instability, Political Changes, Threats, Boycott and strikes, economic trends even Pandemics are factors of Stock Market problems. In the early days of trading people dependent on an inside man or position to understand the condition on market. Since the

invention of the internet researching the market becomes very easy and various news about the companies and market were available. The Market was always volatile but this contributes to the volatility even further. As a result of such volatility trading become much riskier. [24]In the year 2008, the Dow Jones industrial average fell by 777.68 points. It was reported as the largest point drop in history until 2020. The Stock Market did recover from that recession. However, whether to invest in the market greatly remains a question mark. People or Companies always want to play safe so they are always in search of people or Models that can give give them security. Some even prioritize a consistent low-profit model over a high-profit inconsistent model. This paper proposes a Trading Agent based on Reinforcement Learning. This agent uses the ability of Reinforcement learning to learn and understand the market environment thus trade efficiently even in the worst of the conditions.

1.3 Objective and Contribution

In this thesis, it carried out a comparative study of different Reinforcement learning algorithms and studied different risk measurement techniques, various python libraries, and toolkits that help developing an artificial environment. This paper reflects some research about different Neural Networks and the indicators of the Stock market. Data set of different tech companies, such as Google, Microsoft, Apple is used here. Those data are collected from the website yahoo.finance. At first, using the given data set it performs the training part of the model. Then the agent is tested on 90 days fragment of the data set. Different fragments of the data-set were used in different iteration. There are many iterations of observation. In each observation different set of indicators were used. Either that or the same indicators is used on different data sets to figure out the efficiency of the proposed model. By comparing the result of each iteration it gives a pathway to compare the set of indicators. Comparing the algorithms based on the results is a good technique to get the expected outcome. Whether it is safe to invest or not, the stock market will always remain important in the world of finance. So, it is important to address the volatility of the stock market and come up with a consistent solution. Every time the market faces a recession all the companies across the globe suffer from it thus hampers the overall development of the world. Solving this problem of the Stock Market can be very beneficial for all the companies. Thus, bring a significant pace difference in development activities. Therefore, this model follows an approach to building an Agent that will consistently trade with profits or at least will not face a loss within a given period. This may address the insecurities companies have about the trading market.

1. Comparative study of Reinforcement learning, Risk management technique and various python library.

2. Collecting real data of companies like Apple and Microsoft from yahoo.finance.

3. Training and testing of agent iteratively.

4. Make a profitable and yet consistent agent.

1.4 Thesis Orientation

1. Chapter 1 is Introduction where motivation, problem statement, objectives, and contribution are discussed.

2. Chapter 2 is literature review where background and related work are discussed. In the literature review part, it reviews the previous work on Trade Market.

3. Chapter 3 is Proposed Method, here we discussed about our approach of building the agent.

4. Chapter 4 is the Algorithm, here the description of the used algorithm is discussed.

5. Chapter 5 includes performance analysis and results.

6. Chapter 6 is Conclusion and Future works where the conclusion consists of all work till now and future works include the scope of improvement.

Chapter 2

Literature Review

2.1 Background

Reinforcement learning in finance is an active research area as there are not many practical examples of reinforcement learning because it is mainly used in a simulated environment. It is already mentioned that stock market volatility is a major problem as machine learning is mainly used to learn the pattern from data. There are few works on machine learning in finance topics. A neural network is a powerful function approximator and if it is possible to train it for a large amount of time it can adopt different algorithms strategies and learn to exploit them like other traders. TD (0) is a machine learning algorithm that can learn from past experiences. [4] There is an application of TD (0) algorithm on the Korean stock market . The RMS error between value grades and the return grades is 3.07. [11]This paper applies the Adaptive Network Fuzzy Inference System (ANFIS) based on Reinforcement learning (RL)to detect change in trend for price and investment policy. It use momentum and moving average to count delay in a certain time period. This helps to find the best point to long go or short. [7]This paper uses (SARIMA) and discrete Markov process combindly to generate price series with period. Constructed scenario tree helps to planning model. [2]There is another approach instead of using the policy-based approach that paper has applied a value-based approach and using the popular Q-learning algorithm. The result is quite good but the main problem is it's taking a large amount of time to make a good profit. [15] Another paper have used an Recurrent deep neural network for real time trading and financial signal. Here Deep learning(DL) try to understand the market and Reinforcement learning(RL) take trading decisions to get the best reward for an environment. [17]Here this paper tried to figure out how hyper-parameter configuration affect the performance of DNNS and then compare the outcome for Naïve Bayes, k-nearest neighbor, random forest and support vector machines algorithms. [9]This paper tried to analyze different trading techniques. Then it detects hidden pattern for those techniques and calculate efficiency of each techniques by appiled them in real stock market to Buy and sell. [14]This paper proposes Multi-node Evolutionary Neural Networks for Deep Learning (MENNDL) for automatic network selection. It use genetic algorithms for computation and upgrade it through hyper-parameter optimization.

After studying these papers, it is understandable that RNN is the best approach to represent recurrent signals and through Q-learning, it can make a profit with a

long-term investment mind. But none of the mentioned papers represents the power of an Artificial Neural Network. So, this model tries to explore the financial domain with ANN (Artificial Neural Network).
A paper named (Go-Explore: a New Approach for Hard-Exploration Problems) [23] uses a different type of algorithm on different games and measure the accuracy and mention about a new Reinforcement Learning approach called Go-Explore. The current model also tried to implement this algorithm as this is a new technique but there is not much on the internet about this topic cause the paper itself has published in (30 Jan 2019). [9]Another paper suggest to use different technical indicator for buy,sell,hold strategy in stock market.

2.1.1 Financial Indicators

Accumulation/Distribution Index (ADI): Accumulation Distribution index is used to predict the tendency of reversals while using Tops and Bottom. It could be the beneficiary for identifying falling and the rising point when stats. It finds divergence from price movement. ADI mainly productive in a ranging market.

On-Balance Volume (OBV): On-Balance indicator makes a total of positive and negative stock volume. It rises when volume on up days is higher than the volume on down days and vice versa. OBV=Previous OBV+Current trading volume.

Chaikin Money Flow (CMF): Chaikin Money Flow calculate money volume in stock over a definite period. The lookback period 20/21 days. CMF varies above and below zero. Buying pressure can be positive and negative according to its value and this buying pressure determine to buy or sell.

Force Index (FI): Force index indicates power behind a move or turning point based on volume and price. The three key parts are Direction, Extent, volume for the price movement in this indicator. By comparing with moving to aver it identify changes in the power of bulls and bear.

Ease of Movement (EoM, EMV): EMV is a volume-based oscillator which works above and beyond zero lines. It calculates "ease" of price changing. The rise and fall depending on the box ratio. EMV tracks the price of underlying security based on the midpoint.

Volume-price Trend (VPT): Volume-price add or subtract of multiple percentage change in stock price and current volume. It helps to balance demand and supply. It's used in the long-term frame.

Negative Volume Index (NVI): Negative volume index observe the volume change and determine the activation time of smart money. Smart money and volume are opposite in nature. In NVI "255-day" term is used. When NVI above 255-day its bull market and when NVI below 255-day it's bear market

Average True Range (ATR): Average true ranges are mainly ranged over time. When there is a gap in price movement anyone can get an average line from this. It can

be divided into 14 periods like day, month, hour, year etc. Expanding ATR is used for recognition of volatility in the market.

Bollinger Bands (BB): Bollinger band is a composite of three-line where middle line is a simple moving average and the two other lines in the upper and lower band. By three-line it examines the relationship between brand and price then try to find the volatility where price move around the middle line (simple moving average)

Keltner Channel (KC): Keltner channel is quite similar to Bollinger Bands. It also has three lines. But here the middle line can be simple or exponential moving average. The middle line offers price based on user-defined period. KC is a lagging indicator. Overbought and oversold is not a big deal here.

Donchian Channel (DC): Donchian channel is a volatility indicator just work like Keltner channel. Here lower line in 20-day lowest price and upper channel in 20-day highest price where the simple middle average line is just average of the upper and lower line.

Moving Average Convergence Divergence (MACD): Moving average convergence divergence describe the relation among two moving average of the share market price. While calculating MACD 26 period EMA (exponential moving average) is subtracted from 12 periods EMA. There is a crossover in MACD. This indicates overbought or oversold nature of the market.

Average Directional Movement Index (ADX): Average directional movement is a trading model which is divvied into Plus directional indicator (+DI) and Minus directional indicator (-DI). From this average can be calculated. By dividing the market in the various period using this a trader can know about the strength of the market easily..

Vortex Indicator (VI): Vortex indicator is consisting of two oscillators that record positive and negative tendency of the market. It easily reveals about the bullish or Bearish market if the stock line crosses the positive tendency or negative tendency mark. Vortex crossover is the main function here which interacts with other tendency and give the opportunity to know about the strength.

Mass Index (MI): Created by Donald Dorsey, the Mass Index utilizes the high-low range to distinguish pattern inversions dependent on run extensions. In this sense, the Mass Index is an unpredictability pointer that doesn't have a directional predisposition. The Mass Index utilizes the high-low differential to give a smoothed unpredictability measure. The pointer commonly varies in the mid-20s; readings close to the high finish of the authentic range propose expanding unpredictability, which builds the odds for a pattern inversion.

Commodity Channel Index (CCI): The Commodity Channel Index (CCI) is a momentum oscillator that helps in technical analysis. CCI is applied to detect reversal and divergence of the stock price. It first of all remember security's change in price then compare it with average change in price. CCI is an unbound oscillator, which

means there is no upside or drawback. This makes translating an overbought or oversold condition abstract.

Detrended Price Oscillator (DPO): The Detrended Price Oscillator try to recognise pattern from cost to permit recognizable proof of cycles all the more effectively. Cycles longer than the period determined for the pointer are expelled, leaving just the shorter-term cycles. DPO is uprooted to one side so the marker is lined up with the pinnacles and troughs in cost. DPO helps to locate cycles with peaks and trougs. There is shorter and longer DPO settings to get the best fit. It compares last trading price with simple moving average and gives a zero line.From this it is clearly understandable that if it gives positive value the price high and negative value for low price.

KST Oscillator (KST): The Know Sure Thing pointer (KST) is an energy-based oscillator. KST depends on Rate of Change (ROC). Know Sure Thing takes four diverse periods of ROC and smooth's them out utilizing Simple Moving Averages. KST then computes the last worth that varies among positive and negative qualities above and beneath a Zero Line. There is likewise a sign line which is an SMA of the KST line itself. The Know Sure Thing Indicator quantifies the energy of four separate value cycles. Specialized Analysts utilize this data to spot divergences, overbought and oversold conditions and hybrids.

Ichimoku Kinkō Hyō (Ichimoku): Ichimoku is a specialized marker which is used to check force alongside future regions of help and obstruction. The across the board specialized pointer is involved five lines called the tenkan-sen, kijun-sen, senkou length A, senkou range B and chikou range. Ichimoku may look entangled to amateur brokers that haven't seen it previously, yet the intricacy rapidly vanishes with a comprehension of what the different lines mean and why they are utilized.

Parabolic Stop And Reverse (Parabolic SAR): The parabolic SAR is a technical indicator which is used to find price direction. It uses "SAR" method to get the best exit and entry points. It is visible as series of dots. If the dots are above the price its bullish and below the price line its bearish. The parabolic SAR pointer shows up on an outline as a progression of dabs, either above or underneath a benefit's value, contingent upon the course the value is moving.

Money Flow Index (MFI): The Money Flow Index (MFI) is a momentum pointer that estimates the progression of cash into and out of a security over a predefined time-frame. It is identified with the Relative Strength Index (RSI) yet fuses volume, while the RSI just thinks about the cost. The MFI is determined by aggregating positive and negative Money Flow esteems (see Money Flow), at that point making a Money Ratio. The Money Ratio is then standardized into the MFI oscillator structure. Oversold/Overbought levels are for the most part not reason enough to purchase/sell, and brokers ought to consider extra specialized examination or research to affirm the security's defining moment. Remember, during solid patterns, the MFI may remain overbought or oversold for broadened periods.

Relative Strength Index (RSI): RSI is a momentum oscillator which calculate change

of price movement. It ranges between 0 and 100. Market is overbought when value above 70 and market is oversold when value is below 30. Strength oscillators can become overbought (oversold) and remain so in a solid up (down) pattern. The initial three overbought readings foreshadowed solidification.

True strength index (TSI): The true strength indicator is a specialized momentum oscillator. The marker might be valuable for deciding overbought and oversold conditions, showing potential pattern course changes using centerline or sign line hybrids, and cautioning of pattern shortcoming through difference. The TSI changes among positive and negative domain. Positive domain implies the bulls are more in charge of the benefit. Negative domain implies the bears are more in charge. At the point when the marker divergences with value, that may flag the value pattern is debilitating and may turn around.

Ultimate Oscillator (UO): The Ultimate Oscillator indicator (UO) is a specialized investigation instrument used to gauge energy crosswise over three different periods. The issue with numerous energy oscillators is that after a fast advance or decrease in value, it can frame bogus disparity exchanging signals. For instance, after a fast ascent in value, a bearish disparity sign may introduce itself, anyway value keeps on rising. A definitive Oscillator endeavours to address this by utilizing various time allotments in its estimation rather than only one-time allotment which is what is utilized in most other momentum oscillators.

Stochastic Oscillator (SR):Stochastic is a momentum indicator. It's accuracy is much higher for buy and sell. SR generally takes trading period of 14 days to calculate. The Range is from 0 to 100. If the reading is over 80 then the market is overbought and below 20 is oversold.

Williams %R (WR): William shows the opening price,closing price,highest price of a share for a time period. The range is 0 to -100. If the value is above -20 then overbought and below -80 is oversold.

Awesome Oscillator (AO): The Awesome Oscillator is an indicator used to gauge market. AO figures the distinction of a 34 Period and 5 Period Simple Moving Averages. The Simple Moving Averages that are utilized are not determined utilizing shutting cost but instead each bar's midpoints. AO is commonly used to confirm patterns or to foresee potential inversions.

Kaufman's Adaptive Moving Average (KAMA): Created by Perry Kaufman, Kaufman's Adaptive Moving Average (KAMA) is a moving normal intended to represent showcase commotion or instability in the market. KAMA will intently pursue costs when the value swings are generally little and the clamour is low. KAMA will change when the value swings augment and pursue costs from a more prominent separation. This pattern following pointer can be utilized to recognize the general pattern, time defining moments and channel value developments.

Rate of Change (ROC): Rate of change is classified under momentum indicator. It gives percentage of price change. ROC can also give signal about overbought,oversold

and crossover. It's in the zero lie. so if the changes in the positive direction then market is booming or else it's shrinking.

shin Cumulative Return (CR): Cumulative return is an indicator of profit for speculation is the total sum that the venture has picked up or lost after some time, autonomous of the time frame included. Introduced as a rate, the aggregate return is the crude numerical return of the accompanying computation.

Exponential Moving Average (EMA): Exponential moving average significantly react to current data. Unlike simple moving average, it does not put equal weight on all data point rather it put greater weight on the most recent observed data points

Double Exponential Moving Average (DEMA): In the trade market, some traders identify lag as a problem, DEMA eliminates this lag by using two exponential moving averages. It is way quicker to react than the traditional EMA. However, with the increase of time frame it's reaction becomes less quick.

Hilbert Transform Indicator (HT_Trendline): This indicator considers the present and prior price also some feedbacks. This indicator itself is a filter used in digital signal processing. However, this indicator is always used in conjunction with other Indicators.

Moving Average (MA): This indicator is more of a tool, which takes the constantly updated prices and thus constructs the average page constantly. The average is based on a specific period based on the user.

MESA Adaptive Moving Average(MAMA): This indicator is considered as the replacement of the traditional Moving Average. It adapts to the rate change of phase so that the composite moving average can swiftly respond to the price changes.

Parabolic SAR Extended (SAREXT): Parabolic SAR Extended differ from the Classic Parabolic SAR although it derives from the classic one. This is indicator is designed in such a way that the opportunists trader finds it very useful. At the beginning of the trading, it is a lot reactive but over time it gets influenced by movements which however does not change the current trends. When the indicator is above 0 it means buy when the indicator is less than 0 it means to sell.

Simple Moving Average (SMA): As the name suggest this indicator is an arithmetic calculation of the moving average by taking recent price and the number of the period into consideration. SMA smooth out volatility so it is easier to view the price trend of the market.

Triple Exponential Moving Average (T3): This indicator allows a smoother plotting of curve than the simple moving average and it has less lag. Now if the price goes upward this indicator head and people should only do long trades and when the price is below the indicator goes down then we should limit our trade to short trades.

Triangular Moving Average (TRIMA): TRIMA gives the 3 moving average. It can

be generated from both price and volume. The Specialty or difference of TRIMA from other moving average is that it reacts slowly to price changes. It may keep the trader in a long trade however it produces a bigger profit.

Weighted Moving Average (WMA): Weighted moving average assigns a higher weight to the current data and less weight to the distant data. If the price goes beyond the WMA then it is an indication to exit the market but if the price is below or near the WMA it can be an indication to more trading.

Chaikin A/D Line (AD): A/D line calculate total cash flow of a market for a time period. AD can help to understand the security's underlying trend that is anticipating the reversal When the price fluctuate.

Chaikin A/D Oscillator (ADOSC): This indicator is a momentum indicator mot of the price but of the Accumulation Distribution Line. It looks at the pace the price is moving and the underlying pressure of buying and selling given a period.

Absolute Price Oscillation (APO): APO basically gives the difference between a fast-moving average and a slow-moving average. If APO goes above zero it is called bullish and if it goes beyond zero it is called bearish. Bullish indicate upward movement of the trend while bearish indicate the downfall of the trend.

Aroon (AROON): This indicator is the combination of two sperate indicators Aroon up and Aroon down. It focuses on time relative to price. So, it is quite different from other momentum indicators. A positive sign indication in Aroon up and a decline in Aroon down indicates Uptrend. Conversely, a decline in Aroon Up and a surge in Aroon down indicates a downtrend.

Balance Of Power (BOP): This indicator is quite simple. It is nothing but an arithmetic calculation of closing, opening, high and low prices. The value of the calculation can be shown as a smooth moving average. This indicator tells about the trend of a market. Usually,the divergence between price and BOP can help to identify a potential reversal of a trend.

Chande Momentum Oscillator (CMO): This indicator takes the recent gain and recent losses and price moment over the same period into consideration. It is pretty similar to RSI. It measures both the up and down momentum. Traders can easily spot the price divergence between the indicator and the market. A negative divergence means an upward trend and positive divergence means Negative divergence.

Directional Movement Index (DX): This indicator compares the prior highs and lows by drawing two lines and an optional third line known as directional movement shows the difference between the lines. When the positive DI is higher than the negative one there is an upward trend and when the negative DI is higher than the negative one it indicates a downward trend.

Percentage Price Oscillation (PPO): This indicator shows the relationship between two moving average in a percentage term. The PPO indicator is used to spot diver-

gence and compare volatility and asset performance which can lead to price reversal and help understand trend direction. It gives a signal to buy when the PPO line crosses the signal line from below and generate signal for sell when PPO crosses the signal line from above.

Stochastic (STOCH): Stochastic indicator compares the closing price and the previous trading range. In doing so it measures the momentum and speed of price. This means the indicator change even before the price itself thus it is considered of the leading indicator. It is used for anticipating price reversal and identify overbought, oversold price levels.

Ultimate Oscillator (ULTOSC): Ultimate Oscillator measures the momentum of price in different time frames. By using weight over different time frames, the indicator reduces volatility. The ultimate Oscillator generates fewer divergence compare to other Oscillators due to its unique construction. The range of this indicator is 0-100 where below 30 indicates oversold and above 70 indicates overbought.

Upon Studying these Indicators and knowing about its functionality current model will use a set of these indicators manually. Best-suited indicators can be chosen for every set that will make a set in a way such that one indicator will not contradict with others. Indicators that work well with others will be given more priority.

2.1.2 Artificial Neural Network

ANN was developed based on the idea of how the neurons of human brains are connected and how the nervous system works. The ANN is strongly interconnected with one another to perform the different task and come up with solutions. ANN is very popular in pattern recognition, data classification and clustering.
In the Neural Network, the neurons are masterminded into different layers. Every one of the layers is associated with different layers on both of their sides in which the layers on one side occupied with accepting information signals which are required by the network to learn or measure and the layers on the opposite side work with output and responses for the data. In the middle of these two layers, there are hidden layers. The interconnection in the middle of the hidden units and the output unit with the all other units in the layers are known as weight. The weight demonstrates the reliance/association quality in between units.
For every process in a layer of ANN, each info is increased by an initially perceived weight, and it will make the interior estimation of the activity. This worth is additionally modified by an at first created threshold value and sent to an activation function to map the output. After that, the output of that function is sent as the input for another layer, or as the last reaction of a function if the layer is the last. The weight and the threshold esteems are most typically improved to create the right and most exact value.
ANN learn through experience so there is a training phase. Through the training, ANN standardizes all of its weights. The ANN is consisting of two topologies. The feed-forward topology and the feed backward topology. In feed-forward topology, if a node sends a signal to another node then the sender will not receive any signal

from the receiver. In feed-backwards topologies, the sender will receive feedback signals from the receiver.

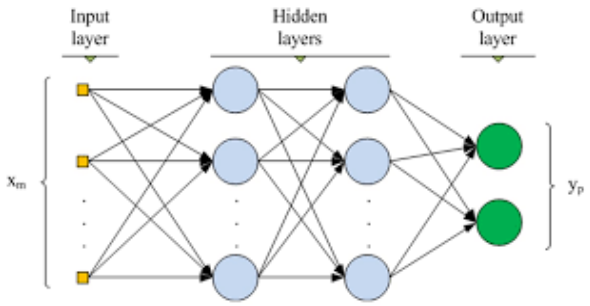


Figure 2.1: Artificial Neural Network

2.1.3 Deep Neural Network

Neural Network can be compared with a game of chess against a Computer. The computer follows an algorithm based on which it decides its tactics, depending on the moves and action of the human. Computer's database already has the data on how each figure moves, and the boundaries of the chessboard, also many different strategies the chess player follows. A different approach to this could be that the computer can learn while playing with others. By updating the neural network weights it can improve and get better from past experience
If multiple layers of node exist in a system and the system derives complex-level of functions from input information then the system is considered a deep neural network. The deep neural network is the process of building creating creative and abstract component from raw data. if a neural network is consisting of more than one hidden layer it is considered a deep neural network. However, in real practice, Deep Neural Network much consists of at least eight hidden layers.

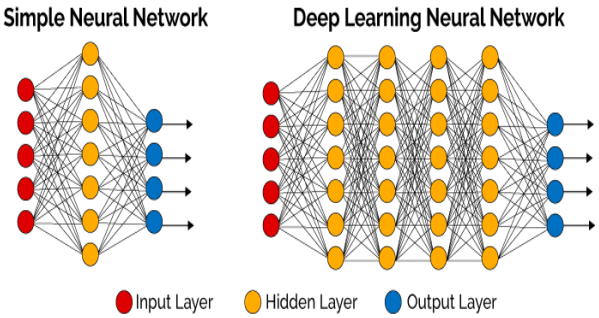


Figure 2.2: Deep Neural Network

2.1.4 Recurrent Neural Network

In a conventional Neural network assume that all data to be independent. But for many fields, this idea is proven bad. For predicting what someone is going to say next to someone needs to know what the person said before that. RNN process sequential

data. RNN perform a common task for every elements available in a sequence and the output is influenced by previous computation that is why it is called recurrent. RNN has a memory which captures what has been already computed.

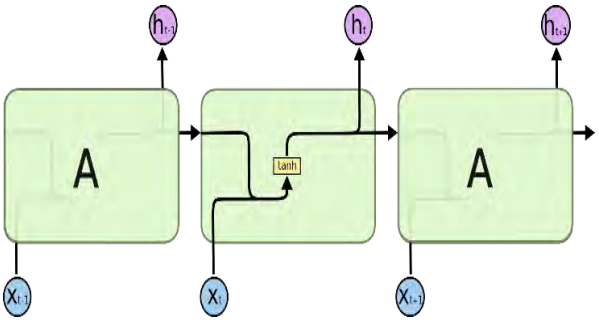


Figure 2.3: Recurrent Neural Network

Types of RNN

The center explanation that recurrent nets are additionally energizing is that they permit to work over successions of vectors: Mostly in both sequences in the input and the input.but it also works on them separately

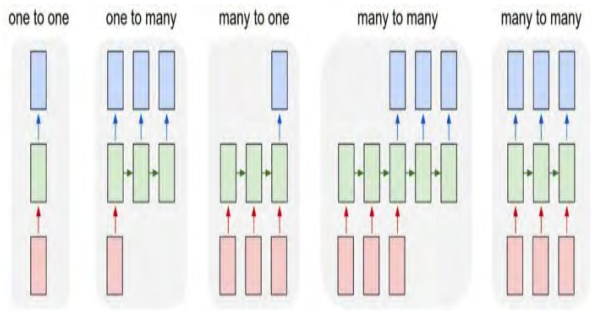


Figure 2.4: Types of RNN

One to one:
It is often referred as Vaniall or Plain neural network. If the input and output size is fixed. And each data is independent of the previous output,One to One RNN is used.
One to Many:
If the Input data is fixed but output gives a sequence of information then this method is used.
Many to One:
When the input data is a sequence of information but the output data is fixed then many to one is used.
Many to Many:
If both the input data and the output data are both sequences of information and the process is recurrent then it is called many to many RNN.
Bidirectional Many to Many:
If both the input and output is synced in addition to data being sequenced of information in both cases.

Deep View

In a traditional Neural Network information is processed independently and have no relation with the previous one. In addition to that, the weight and bias given to hidden units are also different and there is no scope to memorize any information. In RNN Hidden layers have the same weight and bias thus giving them the chance to memorize the information processed through them.
RNN uses a activation function to compute the output, Output $y_{t=}$ $W_{hy}h_t$. Here h_t is the activation function and W_{hy} is the weight of output state. The activation function take the weight of previous hidden state into consideration.

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}X_t)$$

Here,the single hidden vector is denoted as h,weight is denoted as W, current input state weight is denoted as W_hx andprevious hidden state weight is denoted as W_hh.

Pros and Cons of RNN

Pros:

1. RNN can model sequence of data that is dependent on previous one.
2. To extend the effective pixel neighbourhood, RNN can be used alongside convolutional layers

Cons:

1. Vanishing gradient problems

2.1.5 Long Short Term Memory

LSTM is consider as special kind of RNN because of its capability of learning long term dependencies. LSTM is proven to perform outstandingly well on various types of problems. LSTM networks are specially designed to nullify the problem of long-term dependency. They are by default designed to remember information for the long term. LSTM follows a chain structure of recurrent neural network with a changed structure of the repeating module. The repeating module has four very special interactive neural network layer Instead of having one single neural network.

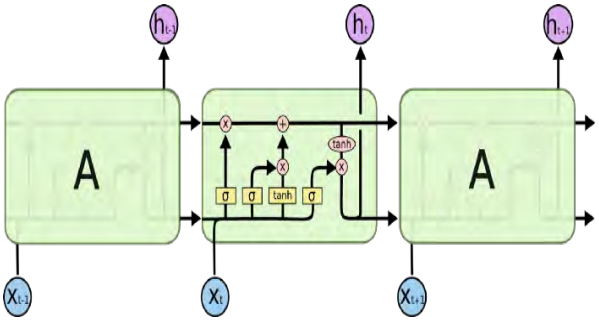


Figure 2.5: LSTM

Core idea behind LSTM

The key to LSTM is the the cell state, it is the horizontal line running through the top of the diagram. Through gates LSTM can remove or add information to the cell state. These gates are made of sigmoid neural net layer and have an output number between zero to one. these values control the adding or removing process. LSTM has three of these gates.

2.1.6 Hyperparameter of PPO

PPO Algorithm Updates its policy using a surrogate loss function. This loss function helps to avoid catastrophic performance drop. The loss function requires a proper initialization of hyper-parameters for better performance. Now, first, what is a policy in Reinforcement learning, set of actions that an agent can take is known as policy. Usually, the agent takes an initial policy and start interacting with the environment and get a reward as feedback. Based on that feedback the agent updates the policy thus establish a new policy. There two main issues in updating the policy, first amount of experience the agent required in order to update the policy. Second the correct manner of updating the policy. The lack of experience become an issues because the typical policy gradient methods are less transition efficient. PPO solves this problem by gathering trajectories and then performing stochastic gradient descent. PPO uses parameter such as Horizon Range, Minibatch Range, Epoch Range for this action.
Horizon Range: This hyper-parameter is also addressed as nstep, time_horizon , timesteps_per_actorbatch.Range: 32 to 5000
Minbatch Range: The range of this hyper-parameter can be higher based on implementations distribution. Usual Range:4 to 4096
Epoch Range: This hyper-parameter is popularly known epochs , optim_epochs , num_epoch ,num_sgd_iter. Range - 3 to 30
The second major issue occurs if the policy is updated in a large step, the performance collapse and never recover. The surrogate loss function of the PPO fixes the issue by keeping the step size in a safe range. The loss function uses hyperparameters such as Clipping Range, KL Target Range, KL Initialization Range, Gamma, GAE parameter Lambda Range.
Clipping Range: 0.1 or 0.2 or 0.3. it is also known as epsilon m, clip_param,clipping parameter epsilon and cliprange.
KL Target Range: The range of this parameter is between 0.003 to 0.03
KL Initialization Range: it is usually ranged between 0.003 to 1
Discount Factor Gamma Range:: 0.8 to 0.9997, In most common cases 0.99 is used. Is widely known as gamma also known as discount.
GAE Parameter Lambda Range: 0.9 to 1, This hyperparameter is most commonly known as lambda

2.1.7 Optuna

Optuna is a framework that optimizes the hyperparameters and was created to make hyperparameter optimization handier and extensible for the newcomers. In deep learning algorithms the rate of learning, training iteration count, batch size etc... are addressed by hyperparameters. These hyperparameters also consist of multiple

neural networks and channel and it doesn't consist of just numerical value, sometimes the deploying momentum SGD or Adam are also referred as hyperparameters. The problem with this hyperparameters are if the value of these hyperparameters are not tuned correctly the model face a performance drop. So it is essential to optimize these hyperparameters.
Optuna is an automated software framework that is invented for the machine learning-based task. This framework emphasizes on an authoritative, define by run approach user API. Optuna is set up based on some key features.
Define by run API:
In characterizes by-run API, clients don't have to characterize everything before an advancement approach that can be perceived with real code without any problem. Optuna arranges hyperparameter optimization with regards to a technique for augmenting or limiting target works that think about a gathering of hyperparameters as input and react back approval score as an output.
An objective function plans the inquiry space of engineering of neural network in the term of the number of existing layers and the summation of hidden units without relying upon characterized static factors remotely. Optuna Describe each process as study and trial.
Efficient Sampling and Pruning Mechanism:
The Sampling methods are two type relational sampling method and independent samplings. The first one handles the interrelationship amid parameters and the second one samples every parameter individually. Addition to this Optuna has a feature that supports customized sampling method.
The pruning mechanism operates in two parts regular monitoring and Adjorunes. The middle objective values are regularly monitored while the experiments that don't meet predefined expectations are Adjorunes.
Scalable and versatile System that is easy to setup:
Scalable system controls numerous tasks from the heavy experiments which typically needs many workers for trial-level to lightweight experiments via interactive interfaces.

2.2 Related Work

[20] This paper introduced the proximal policy optimization algorithm and discuss about its benefits and easy implementation process.
[22] In this paper the authors introduced a new agent. This agent analyze trading strategies and try to maximize invest return for 30 days stock price. Compared to Dow Jones Industrial Average and the traditional min-variance portfolio allocation strategy the agent showed a better performance. There proposed approach of deep reinforcement learning outperform the two baselines for Sharpe ratio and cumulative returns.
[4] This paper is all about applying reinforcement learning to the Stock market. It mainly discusses why reinforcement learning applies to the stock market. Since Reinforcement learning algorithm learns from experiences the algorithms try to learn the values of states for stock price trend at a given time session.
[3] This paper focuses on the problem of extracting the reward function given observed, optimal behaviour of Inverse Reinforcement Learning. The paper addresses the degeneracy issue and show the path of removing the existing issue.

[13] This paper is about Inverse reinforcement learning approach of identifying the strategies of traders based on their behaviours. The findings of there tests suggest that most frequencies used trading strategies can be accurately identified and profiled by observing the trading actions of individuals
[10] This paper depicts compound fortification learning (RL) that is an all-encompassing RL dependent on the compound return. Compound RL augments the logarithm of expected twofold exponentially limited compound return consequently based Markov choice cycles (MDPs). The commitments of this paper are (**??**)heoretical portrayal of compound RL that is an all-encompassing RL structure for augmenting the compound return in a return-based MDP and (**??**)xperimental outcomes in an illustrative model and an application to back.

[25] This paper explain how Reinforcement learning algorithms helps the agent to get the optimal policy for obtaining maximum cumulative result for some states of action. This paper also showed how different algorithms Of RL can maximize reward in term of financial application but takes a lots of time even 40 years. It propose a optimal Rl technique which is applicable in economics,games theory,operation research with optimal policy.

[8] This paper tells that investors over-extrapolate from their encounter when making reserve funds choices. Financial specialists who encounter especially fulfilling results from 401(k) saving a tall normal and/or mood change return as well as increase the reserve funds rate more compared to agents who have less fulfilling encounters. This finding isn't driven by total time-series stuns, pay impacts, judicious learning approximately contributing ability, speculator settled impacts, or different invest based of specific time frame which is related with the stock, bond, and cash resource classes. Let's talk about suggestions for the value premium astound and intercessions pointed at moving forward family money related results.

[5] This paper proposed a fully automated trading system using adaptive reinforcement learning(ARL). It works on a layered structure of risk management overlay,dynamic utility layer and machine learning algorithm. By using Recurrent reinforcement learning(RRL) this system ensure risk return for trading..

[6] Stock exchanging is an imperative decision-making issue that includes both stock choice and resource administration. Even though numerous promising comes about have been detailed for anticipating costs, selecting stocks, and overseeing resources utilizing machine-learning methods, considering all of them is challenging since of their complexity. In this paper, they show a modern stock exchanging strategy that consolidates dynamic resource allotment in a reinforcement-learning system. The proposed resource allotment procedure, called meta policy (MP), is outlined to utilize the worldly data from both stock proposals and the proportion of the stock support over the resource. Local traders are built with pattern-based different indicators and utilized to choose the buy cash per suggestion. Defining the MP within the reinforcement learning system is accomplished by a compact plan of the environment and the learning operator. Exploratory comes about utilizing the Korean stock advertise appear that the proposed MP strategy .

[18] In this case With the breakthrough of computational control and deep neural networks, numerous ranges that they haven't investigated with different methods that were inquired about thoroughly in past is attainable. In this paper, it walked through conceivable concepts to attain robo-like trading or prompting. In arrange to achieve a similar level of execution and simplification, like a human trader, the agents learn for themselves to make fruitful techniques that lead to the human-level long-term rewards. The learning show is executed in Long Short-Term Memory (LSTM) repetitive structures with Reinforcement Learning or Advancement Techniques acting as agents the strength and possibility of the system is confirmed on GBPUSD exchanging.

[21] This paper propose a system for trading in short term time period using Reinforcement learning. They appilied Q-learning algorithm along with three hidden layers of ReLU neurons as RL agent. The used framework has new state and reward signals, and a method for efficiently use of historical tick data. It helps to improve training quality and testing accuracy

[1] This paper focuses on optimized objective function for calculating trading performance. It uses recurrent reinforcement learning (RRL) algorithms. The performance function here used are differential sharpe ratio,profit,sharpe ratio and economic utility. Differential sharpe ratio gives more consistent better result for this system

[16] This paper proposes a strategy of applying reinforcement learning,which is appropriate for modelling and studying different actions in real situations, like stock price prediction. They propose a recurrent reinforcement learning method, with a coherent risk adjusted performance objective function, called Calmar ratio. It helps to identify both buy and sell signals and asset allocation weights..

Chapter 3

Proposed Method

The stock market is so volatile that to predict a particular day price is almost impossible cause it's completely random. There are so many supervised learning techniques out there to forecast the stock market price but those are not that much efficient. Because of its volatility, people can't predict it with some specific algorithm. There are many investors out there who are making money by buying and selling the stock, so if this model could make an agent that can work as human traders that will accomplish it.
Now, let's describe the environment as we cannot train the algorithm with the real trading environment because that would be costly so it creates a simulated environment so that the algorithm can learn over time. This system uses OpenAiGym to create a simulated environment with AAPL, MSFT stock data set. Now let's take a closer look at the parameter. As it is known to all that reinforcement learning is all about agent interacting with the environment and for that this model needs to specify some of the variables so that the agent can understand the environment.

1. State: As a state, it uses stock data points for the last 5 days and scaled them between 0-1. Then the balance, shareholders, total share sold total sales value and the additional set of indicators are added.

2. Action: As an action, it selects the number of shares to buy/sell/hold.

3. Reward: This is the most important function of any reinforcement learning approach by using this reward function an agent can learn and for the reward function this approach has used current account balance multiplied by delay modifier. Delay modifier takes the current time step and divided it by max steps.

$$Reward = \frac{Current\ timestep}{Max\ time\ steps} \times Balance$$

This function works such a way so that the agent will mainly focus on future profit less than the current profit. There is a have application of Multi layer perception network (2 layers of 64) to train the agent. Figure-2 describes the whole working process.

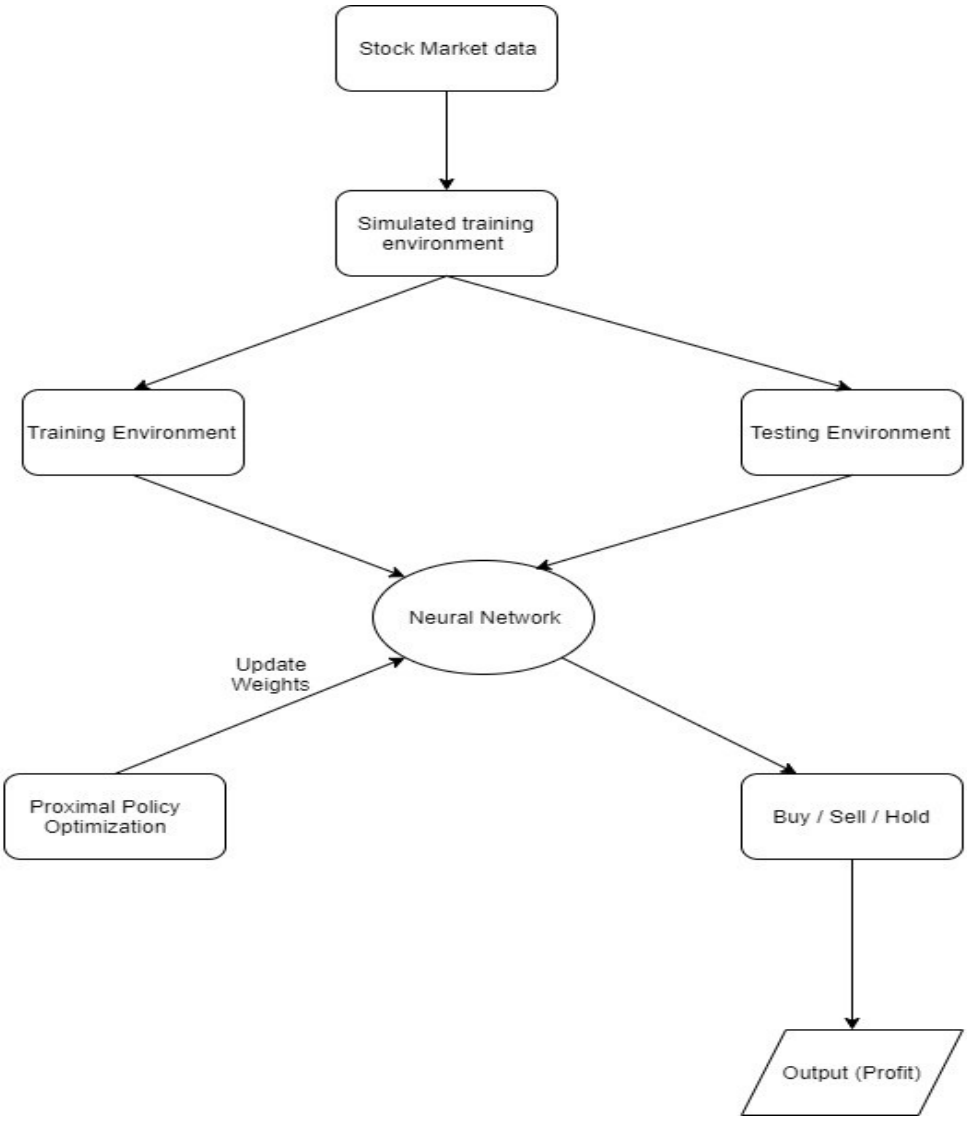


Figure 3.1: Working Process

3.1 Preliminary Phase

Reinforcement learning is a technique to train an agent to learn some specific task and in this case that is trading. There are so many reinforcement learning algorithms out there but the decision was to go with A2C(Asynchronous Advantage Actor Critic) at the initial stage and it is experimented with the A2C but due to some of the problems it is decided to drop the idea of using A2C as the results are not so good. we tested our model multiple times and the model was not consistent. After that we use RNN and the results are same then we use RNN-LSTM the model is giving us good profit for 8/10 times.we also used optuna to finetune the hyperparameters. And the results are same the model is not consistent.

3.2 Secondary Phase

At the first phase, it used the normal columns of the dataset, but as the result was fluctuating from observation to observation, then a decision was made to add some extra indicators. Study of some indicators can help to find a better and stable output. For observation and exploration, this model decided to pick a random number of indicators and those indicators were used randomly too. There is an observation of the results using 50 different sets of indicators. And for each set, the system has iterated 10 times. From all those observations we came to this conclusion that these ('volatility_dcl','volatility_bbm','trend_ema_fast','volume_fi','volatility_dchi','volatility_bbh') parameters are the best suited for convenient and stable outputs. We switch to Recurrent Neural Network from DNN. because RNN is more efficient in the sequential data processing.

3.3 Final phase

After those observations, Further study of the indicators were needed in order to gain a clear understanding of how the market in real-life work. That is how the traders use the indicators efficiently, what indicators go in conjunction with the others and how the indicators react to different scenarios. The goal was to find the best suited yet common set of indicators for all kind of scenario. It was planned to use this knowledge to manually select the indicators to gain a better profit.
After some time it was noticed that there was some noise in the model, that is why the model was not stable. It was changing drastically. Upon researching it was found that it was happening due to unoptimized hyperparameter. So, to optimize the value of the hyperparameter, it uses optuna. After optimizing the value of hyperparameter it was possible to reduce the noise of the model. And at last we use PPO(Proximal Policy Optimization) and for neural network architecture it uses RNN after that it is giving us consistent results. Then it applies indicators manually on historical data and record the profit then it uses those same indicators in our model and record the profit then we compare those two profits. For using indicators manually we use TradingView. Finally It's been found that our model is always beat the human profit.

Chapter 4

Algorithms

4.1 Actor-Critic Methodology

The problem with the vanilla policy gradient is that sometimes it generates a cumulative reward of 0. So, it is not possible to distinguish whether the Action was good or bad. Thus, the Actor failed to learn. The Equation for Vanilla gradient Policy −

$$\nabla_\theta J(\theta) = E_T[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t \mid s_t) G_t]$$

then it expands this equation in this manner −

$$\nabla_\theta J(\theta) = E_{s_0,a_0,....,s_t,a_t}[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t \mid s_t)] E_{r_{t+1},s_{t+1},\ ...,r_T,s_T}[G_t]$$

Here the 2nd part of the equation is the q value,it was already known for q learning and value iteration.

$$E_{r_{t+1},s_{t+1},\ ...,r_T,s_T}[G_t] = Q(s_t, a_t)$$

So,After rewriting the equation as –

$$\nabla_\theta J(\theta) \sim E_{s_0,a_0,....,s_t,a_t}[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t \mid s_t)] Q_w(s_t, a_t)$$

$$\nabla_\theta J(\theta) = E_T[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t \mid s_t) Q_w(s_t, a_t)]$$

Here, Q value can be learned from the Q function. That is by parameterizing the Q function with a neural network can easily get the q value. This is where Actor-Critic Method comes in –
1. The Critic estimates the value function. This could be Q value (Action Value) or V value (State Value)
2. The "Actor" updates the policies Based on the direct suggestion of the "Critic". The neural network parameterizes both the critic function and the actor function. In the above-derived equation, the q value was parameterized by the critic neural network, therefore it is known as Q Actor-Critic.
It is observed that both the Value and critic network are updated after each step.

Algorithm 1 Q Actor Critic

Initialize parameters s,θ and ω and learning rates α_θ andα_ω;sample a$\sim$ π_θ(a—s).
for t=1...T **do**
sample reward $r_t$$\sim$R(s,a) and next state s$\sim$ P(s'|s,a)
then sample the next action a$\sim$ π_θ(a| s)
update the policy parameter: $\theta \leftarrow \theta+\alpha\ \theta Q_w(s,a)\bigtriangledown_\theta\ log\pi_\theta$ (as);compute
the correction (TD error) for action-value at time t:

$$\delta_t=r_t+\gamma\ Q_w(s',a')-Q_w(s,a)$$

and use it to update the parameters of Q function:

$$\omega \leftarrow \omega+\alpha_\omega\ \delta_t\omega Q\omega(s,a)$$

move to a$\leftarrow$a and s$\leftarrow$s
end for

Now, if it uses the V function as the baseline function, it can subtract the Q value from the V value. This will help to understand how efficient it will be if the model takes a specific action compared to average or the general action. This is called advantage value.

$$A\left(s_t,a_t\right)=Q_w\left(s_t,a_t\right)-V_v(s_t)$$

But it's not necessary to construct separate neural networks for Q and V value in addition to the policy network, firstly because it will be very inefficient and there is a better way of solving the problem. The Bellman optimally equation can be used which builds a relationship between the two values.

$$Q\left(s_t,a_t\right)=E[r_{t+1}+\gamma V\left(s_{t+1}\right)]$$

Now after rewriting the Advantage value as –

$$A\left(s_t,a_t\right)=r_{t+1}+\gamma V\left(s_{t+1}\right)-V_v(s_t)$$

So,The v value can be found by using the neural network only once. Again,if it rewrites the equation as –

$$\nabla_\theta J\left(\theta\right)\sim\sum_{t=0}^{T-1}\nabla_\theta log\pi_\theta\left(a_t\mid s_t\right)]Q_w(s_t,a_t)(r_{t+1}+\gamma V\left(s_{t+1}\right)-V_v(s_t))$$

$$=\sum_{t=0}^{T-1}\nabla_\theta log\pi_\theta\left(a_t\mid s_t\right)]Q_w(s_t,a_t)A(s_t,a_t)$$

This is called the Advantage Actor-Critic.

4.2 Proximal Policy Optimization

Based on real impact PPO is way better than the conventional state-of-the-art approaches and it is very simple to implement and easy to tune. The convergence problem of the policy gradient is reduced by the natural policy gradient. Nevertheless the natural policy gradient can not scale for large scale problems because it

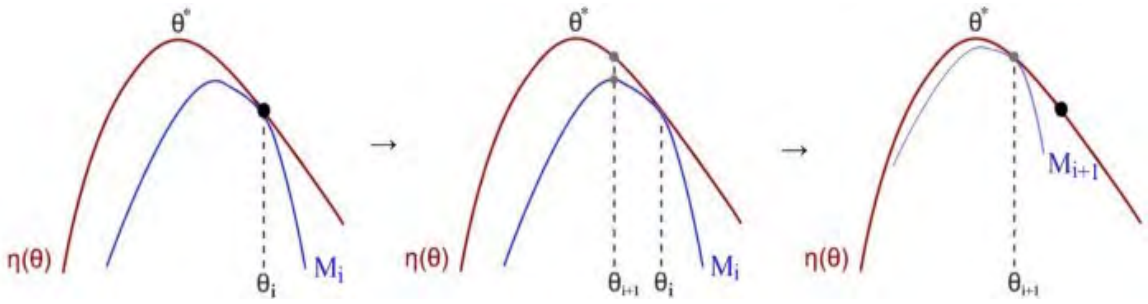
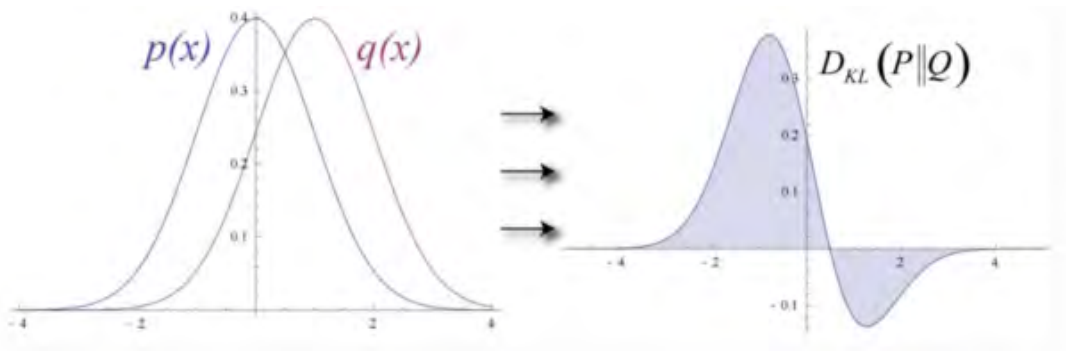


Figure 4.1: Minorize-Maximization Algorithm

needs a second-order derivative matrix to operate. In solving real tasks, the computational complexity is very high. By approximate the second-order derivate this complexity can be reduced to some degree. However, PPO uses different technic. In the object function, It formalizes the constraint instead of imposing strict constraints. The first order optimizer can be used as the gradient descent method if the constraint at all costs is not excluded. Thus, the method can be optimized. Even if the constraint is violated in some cases, the computation still is simple. There is a chance of maximizing the reward by using the Minorize-maximization algorithm iteratively. There are two major optimization method line search and trust region. Line search is like the gradient descent method which is very easy and popular. Line searching is like hiking, if it takes less step it is needed an eternity to reach the peak and if the step size is way too large, it may fall off. If somehow it survives the fall but since the policy gradient is based on the on-policy method it looks for action from the present state so it will only leave with bad state policy and will start exploration again from a bad state. This hurts performance. In the trust-region first, select the maximum step size that need to be explore. if it find the optimal point within that trust-region it will start exploring from there again. In trust region method by selecting the maximum step size with an initial guess after that it can be readjusted dynamically. If the divergence is getting large then there is a solution. It can shrink the trust-region so that it can avoid bad decision. And if the policy changes too much the regions can be shrunk also. In PPO, It can limit how far a model can change the approach in every cycle through the KL-divergence. The difference between two data distributions p and q is measured by KL-divergence

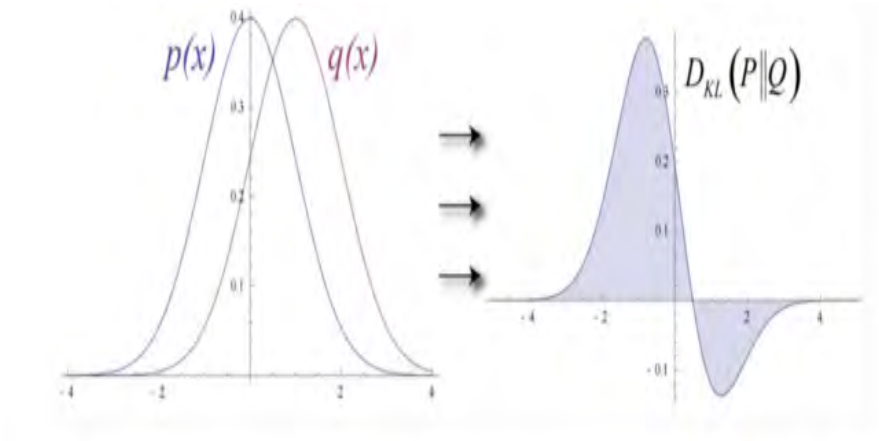
$$D_{KL}(P||Q) = E_x log \frac{P(x)}{Q(x)}$$

The goal is to not finding a new policy that is too much different from the old one



A lower bound function M can be used to limit the policy change, Where M is

$$M = L(\theta) - C.\overline{KL}$$



Here L(θ) equals

$$\widehat{E}_t[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}]\widehat{A_t}$$

and the second term is known as KL-divergence. In the new policy L is the Advantage function. It is assessed by a previous (or present) policy and afterwards re-calibrates utilizing the likelihood proportion of the new and the previous policy. Since L reduces the variance of estimation, it can be instead of the expected reward. the optimal policy will remain the same,given that the baseline is independent of the policy parameters. [12]The second term in M can be established as

$$\eta(\widetilde{\pi}) - CD\frac{max}{KL}(\pi, \widetilde{\pi}), \ where\ C = \frac{4\epsilon\gamma}{(1-\gamma)^2}.$$

$$D\frac{max}{KL}(\pi, \widetilde{\pi}) = max_S D_{KL}(\pi(. \mid s)) \,||\widetilde{\pi}(. \mid s).$$

$$\epsilon = \frac{max}{s,a} \mid A_\pi(s,a) \mid$$

The maximum of KL-divergence is the second term in M. But since it is too hard to find, the requirement can be independent by implying the average of KL-divergence. Locally, the advantage function is approximate to L at the current policy. But as it moves from the previous policy it gets less accurate.The upper bound of this inaccuracy in second term is denoted by M. After considering the upper bound of this error, the Model can ensure that the computed optimal policy within the trust region is better than the previous policy in every way.But The accuracy will be way too off,If the policy is not inside the trust region, So, even if the calculated value is better the policy can not be trusted. Let's summarize the objective below as

$$\underset{\theta}{\text{maximize}} \quad \hat{E}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{oud}}}(a_t|s_t)}\hat{A}_t\right]$$
$$\text{subject to } \hat{\text{E}}_t\left[\text{KL}\left[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)\right]\right] \leq \delta \text{ Or}$$

$$\underset{\theta}{\text{maximize}} \quad \hat{E}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t\right] - \beta\hat{E}_t\left[\text{KL}\left[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)\right]\right]\Big\}$$

Numerically, the two conditions above can be set out to a similar optimal policy. the theoretical threshold for is considered to be too conservative since it is very small. So, the condition can be selected one more time by setting the conditions as tuneable hyperparameters. It is already mentioned, M ought to be optimized. Therefore, by going one step further approximate M value to a quadratic equation. This quadratic equation is a convex function and it heavily studies about optimizing it in high dimensional space. The terms can be expanded up to the second-order by using Taylor-series. In any case,the KL-divergence term and will be overlooked if the second-order of $\mathcal{L}$ is a lot smaller than it.

$$\frac{maximize}{\theta}\ \widehat{E}_t\ [\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)}\hat{A}_t]$$

$$subject\ to\ \widehat{E}_t[KL[\pi_{\theta old}\left(.\mid s_t\right),\pi_\theta\left(.\mid s_t\right)]]\leq\delta$$

$$\mathcal{L}_{\theta_k}(\theta)\approx\mathcal{L}^0_{\beta\left(\bar{\theta}_k\right)}+\boldsymbol{g}^T\left(\theta-\theta_k\right)+\ldots$$

Therefore, the objective and the constraint can be approximate as:

$$\mathcal{L}_{\theta_k}(\theta)\approx g^T\left(\theta-\theta_k\right)\quad g\doteq\nabla_\theta\mathcal{L}_{\theta_k}(\theta)\big|_{\theta_k}$$
$$\bar{D}_{\kappa l}\left(\theta\|\theta_k\right)\approx\tfrac{1}{2}\left(\theta-\theta_k\right)^T H\left(\theta-\theta_k\right)\quad H\doteq\nabla^2_\theta\bar{D}_{kl}\left(\theta\|\theta_k\right)\Big|_{\theta_k}$$

Theb objective function becomes:

$$\theta_{k+1}=\arg\max_\theta g^T\left(\theta-\theta_k\right)\text{ s.t. }\tfrac{1}{2}\left(\theta-\theta_k\right)^T F\left(\theta-\theta_k\right)\leq\delta$$

The Solution of the quadratic equation analytic:

$$\theta_{k+1}=\theta_k+\sqrt{\tfrac{2\delta}{g^T F^{-1}g}}F^{-1}g$$

The challenges of this solution is that it involves calculation of second-order derivative and its inverse which is an over the top expensive operation

$$F=\nabla^2 f=\begin{pmatrix}\frac{\partial^2 f_1}{\partial x_1^2} & \frac{\partial^2 f_1}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2 f_1}{\partial x_1\partial x_n}\\ \frac{\partial^2 f_2}{\partial x_1^2} & \frac{\partial^2 f_2}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2 f_2}{\partial x_1\partial x_n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial^2 f_m}{\partial x_1^2} & \frac{\partial^2 f_m}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2 f_m}{\partial x_1\partial x_n}\end{pmatrix}$$

So, there are two ways to deal with address this issue:
1. to bring down the computational unpredictability, some calculations involving the second-order derivative and the inverse of it could be Approximated.
2. Make the first-order derivative solution, like the gradient descent, closer to the second-order derivative solution by adding flexible constraints.
ACKTR and TRPO adopt the first approach. PPO is similar to the second one. A bad policy decision once a while is buyable so it is safe to stay with the first-order solution like the stochastic gradient descent.However, by taking a risk to add a delicate limitation to the objective function as a result the optimization will have finer protection that model is trying to optimize within a trust region. Hence, the possibility of an awful choice become littler.

4.2.1 PPO with Adaptive KL penalty

The better approach to figure the goal is to change the constraint to a penalty in the objective function:

$$\underset{\theta}{\text{maximize}} \quad \hat{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] - \beta \hat{E}_t \left[\text{KL} \left[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t) \right] \right]$$

Weight of the penalty is controlled by . If the new policy is different from the previous policy it penalizes the objective. From the trust region it can be dynamically changed .The KL-divergence between the previous and current policy is denoted below as d. if than the target value is smaller than d we shrink β. And the trust region can be expanded if it falls beneath another target value

$$d = \widehat{E_t}[KL\left[\pi_{\theta old}(. \mid s_t), \pi_\theta(. \mid s_t)\right]]$$

Here is the pseudocode:

Algorithm 2 PPO with Adaptive KL Penalty

Input: intial policy parameter θ_0, intial KL penalty β_0,target KL-divergence δ
for k=0,1,2... **do**
Collect set of partial trajectories $\mathcal{D}_k$ on policy$\pi k = \pi(\theta k)$

$$\theta k + 1 = \arg\max \theta \mathcal{L} \theta k(\theta) - \beta k \bar{D} KL(\theta | \theta k)$$

by taking K steps of minibatch SGD (via Adam)
if $\bar{D}KL(\theta k + 1|\theta k) \geq 1.5\delta$ **then**

$$\beta k + 1 = 2\beta k$$

else if $\bar{D}KL(\theta k + 1|\theta k) \leq \delta/1.5$ **then**

$$\beta k + 1 = \beta k/2$$

end for

4.2.2 PPO with Clipped Objective

PPO with clipped objective can perform finer than previous one.This approach maintain two policy network in its implementation. The first one is the current policy that can be rectified if wanted :

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

The second one is the policy that that is used last time to collect samples

$$\pi_{\theta k}(a_t|s_t)$$

With the possibility of significance inspecting, it can assess another policy with tests gathered from an older policy. This improves test proficiency.

$$\underset{\theta}{\text{maximize}} \quad \hat{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right]$$

Yet, after refining the current policy, the contrast between the present and the previous policy is becoming bigger. The fluctuation of the assessment will increment and it can be a bad decisions as a result of the inaccuracy. So, the better option to synchronize the second network with refined policy again for every four iterations.

$$\pi_{\theta k+1}(a_t|s_t) \leftarrow \pi_\theta(a_t|s_t)$$

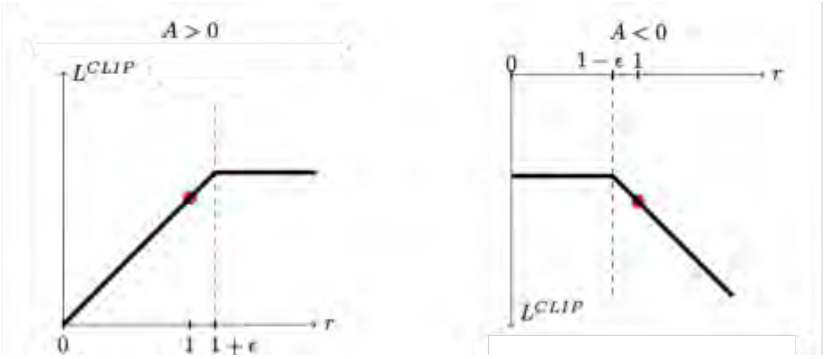
For computing a ratio between the new and the old policy:

$$r_t(\theta) = \pi_\theta(a_t \mid s_t) / \pi_{\theta k}(a_t|s_t)$$

This proportion gauges how the distinction between the two policies. if the new policy is far away from the previous policy , We build another objective function to cut the assessed advantage function. Hence, Our new objective function becomes:

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

If the probability ratio between the two policy falls outside the range (1 — e) and (1 + e), the advantage function will be clipped.[19]in the PPO paper e is set to 0.2.



Viably, this dishearten huge policy change in the event that it is outside our agreeable zone. Pseudocode:

Algorithm 3 PPO with Clipped Objective

Input: intial policy parameter θ_0, clipping threshold $\in$
for k=0,1,2... **do**
Collect set of trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$
Estimate advantage using any advantage estimation algorithm
compute policy update

$$\theta_{k+1} = \arg\max_{\theta} \mathcal{L}^{CLIP}_{\theta_k}(\theta)$$

by taking K steps of minibatch SGD (via Adam), where

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

end for

This new technique is basic and can utilize Gradient Descent like Adam to advance it. Truth be told, it is a disappointment for making so detailed investigation on the matter however concocts such a straightforward arrangement.

Chapter 5

Performance Analysis

5.1 Analysis

Throughout the phases of this thesis various experiment were done by using different Algorithm,different Neural network and lastly different set of indicators.Firstly, an effort to figure out which algorithm work best with which neural network were made. After that, fined tuned between combination of indicators were done to get the better results.fined tuning of the hyper parameters of Algorithm were also carried out in order to stabilize the model.Alongside with the extra combination of indicators, five basic observation space were used. Which are Open, High , Low , Close and Volume.

In the second phase of the work 50 observation were done and for each observation it has iterated over the same data for understanding how stable and consist the output is. There is another reason for changing parameters. The thing is the trade market trends differ from company to company. But the same kind of companies follows more or less the same pattern. For example- The tech-related companies follow a similar kind of trends, the garment companies follow different trends from the tech companies but follow the same finical trends within themselves. So, the effort was to explore and figure out whether it is possible to figure out a unique pattern for all kind of sectors.After completing the observations it shows that ['volatility_dcl','volatility_bbm','trend_ema_fast','volume_fi',' volatility_dchi', 'volatility_bbh'] this set of indicator alongside with the twelve basic observation space gives the best result.Our Findings of this phase is given below in a table.

Observation	Profit($)	Observation Space
1	1760.56	'volatility_dcl','volatility_bbm','trend_ema_fast','volume_fi',' volatility_dchi', 'volatility_bbh'
2	1466.56	trend_ema_fast','volatility_kch','trend_ema_slow','volume_vpt', 'volatility_bbhi','volatility_dch',,'volatility_kcl','volatility_bbli', 'volume_cmf','volume_sma_em','volume_em','volatility_bbl', 'volume_fi','volume_nvi','volatility_dcli'
3	889.00	volatility_bbl','trend_macd_signal','volatility_atr','volatility_bbh', 'volatility_kchi','volatility_bbli','trend_macd','volatility_dcl', 'volatility_dch','volatility_kcli', 'volume_fi','volume_em'
4	741.61	volatility_bbl' ,'trend_macd_signal','volatility_bbh', 'trend_macd_diff', 'volatility_dchi','volume_adi','trend_macd', 'trend_ema_fast','volatility_dcl','volatility_bbm','volatility_dch', 'volatility_kch','trend_adx', 'volume_vpt', 'volume_cmf', 'volatility_kcli' , 'volume_obv'
5	737.86	volume_vpt','volume_sma_em','volatility_bbh','volatility_dch', 'trend_ema_slow', 'volume_nvi', 'volatility_bbli', 'volatility_dcl' 'volatility_dcli' ,'trend_adx'
6	734.64	'volume_sma_em','volatility_bbh','volatility_dch', 'trend_ema_slow ', 'volume_nvi','volatility_bbli','volatility_dcl', 'volatility_dcli','trend_adx'volatility_atr','volatility_kcc', 'volatility_dcli','volatility_bbli','volatility_kchi','volatility_dch' , 'volatility_bbm', 'trend_adx' ,'volatility_kcl','trend_macd' 'volatility_bbl', 'trend_ema_fast', 'volatility_kch'
7	711.81	volume_obv','volatility_kcc','volume_cmf','trend_macd_signal', 'volatility_bbl','volatility_bbh','volatility_atr','volatility_kcli', 'volume_em','volatility_bbm', 'volatility_dcl'
8	669.82	volume_adi','volatility_kch','trend_macd_signal','trend_adx', 'volatility_bbl' ,'volatility_bbli'
9	655.01	volatility_bbhi','volatility_bbli','trend_macd_diff','trend_macd', 'volume_cmf','trend_macd_signal','volume_sma_em', 'volatility_kcl',volume_obv','volatility_kcli','volatility_kchi', 'trend_ema_slow','trend_ema_fast','volatility_dcli','volume_fi', 'volume_nvi'

Table 5.1: Result Of 2nd Phase

In the Final phase of the work, after optimizing the hyperparameter and studying the indicators thoroughly set of indicators are applied manually. This time first calculated the profit manually then the same set of indicators are used to test and train the model. For each set of indicators, it was iterated several times and then counted the average profit as profit. The findings are given below.

Observation	Manual Profit ($)	Profit($)	Observation Space
1	1266	1904.8	'volatility_atr','volatility_bbm','volume_obv', 'trend_mass_index',momentum_mfi'
2	1196	2037	'volume_cmf','volume_fi','volume_em','volatility_kcc', 'momentum_ao', 'momentum_tsi','others_dr,'momentum_roc'
3	1168	1605.2	'volume_adi','volume_vpt','momentum_mfi','volatility_kch', 'trend_trix','momentum_tsi', 'momentum_ao'
4	1118	1471.4	'volume_fi','volume_nvi','volatility_kch','trend_adx','trend_kst', 'momentum_kama','momentum_wr', 'others_cr'
5	1042	1751.6	'volume_fi','volume_nvi','volatility_kch','trend_adx','trend_kst', 'momentum_kama','momentum_wr', 'others_cr'
6	968	1958.4	'volume_em','trend_vortex_ind_pos','Weighted_Price', 'momentum_rsi','momentum_stoch','volume_fi','trend_dpo', 'others_dlr'
7	882	1492.2	'volume_cmf','volume_obv','volatility_dcl','trend_macd', 'trend_kst','momentum_kama','trend_cci','momentum_roc', 'others_dr','momentum_uo'
8	742	1751.8	'volume_vpt','volume_nvi','volume_sma_em','volatility_kchi', 'trend_vortex_ind_pos','momentum_ao','momentum_tsi', 'trend_psar','trend_trix','momentum_ao','momentum_mfi', 'others_dlr','trend_mass_index'

Table 5.2: Result Of Final Phase

Different Algorithms along with different Neural Network were used in this experiment. Firstly, Actor critic method was implemented with DNN as the Neural network. But as the result was not satisfactory it was decided to apply A2C With RNN architecture. The result were better than the previous but the problem of instability remained. A lot of fine tuning using different algorithm and different neural network were made. At the end, Proximal Policy Optimization along with RNN-LSTM produced the best performance. The findings of the experiment are given below in a Table.

Algorithm Name	Neural Network	Profit($)
Proximal Policy Optimization	RNN-LSTM	2037.00
Proximal Policy Optimization	DNN	1854.34
Proximal Policy Optimization	RNN	1421.33
Asynchronous Actor Critic	DNN	1254.72
Asynchronous Actor Critic	RNN-LSTM	854.14
Trust Region Policy Optimization	DNN	712.29

Table 5.3: Result Using Different Algorithm

5.2 Result

In the second phase of the observation it was founded that, taking 'volatility_bbli','volatility_dcl', 'volatility_bbm', 'volume_cmf', 'trend_ema_fast','volume_fi', 'volatility_dchi', 'volatility_bbh' These observation space alongside the five regular observation space (Open , High , Low , Close , Volume) can get most convenient and stable outputs. Some interesting phenomena were noticable during the experiment. For some indicators, the model never invested in the trade market. As a result, the profit was zero. However, an AI built for maximizing the rewards deciding not to take any sort of risk is interesting.

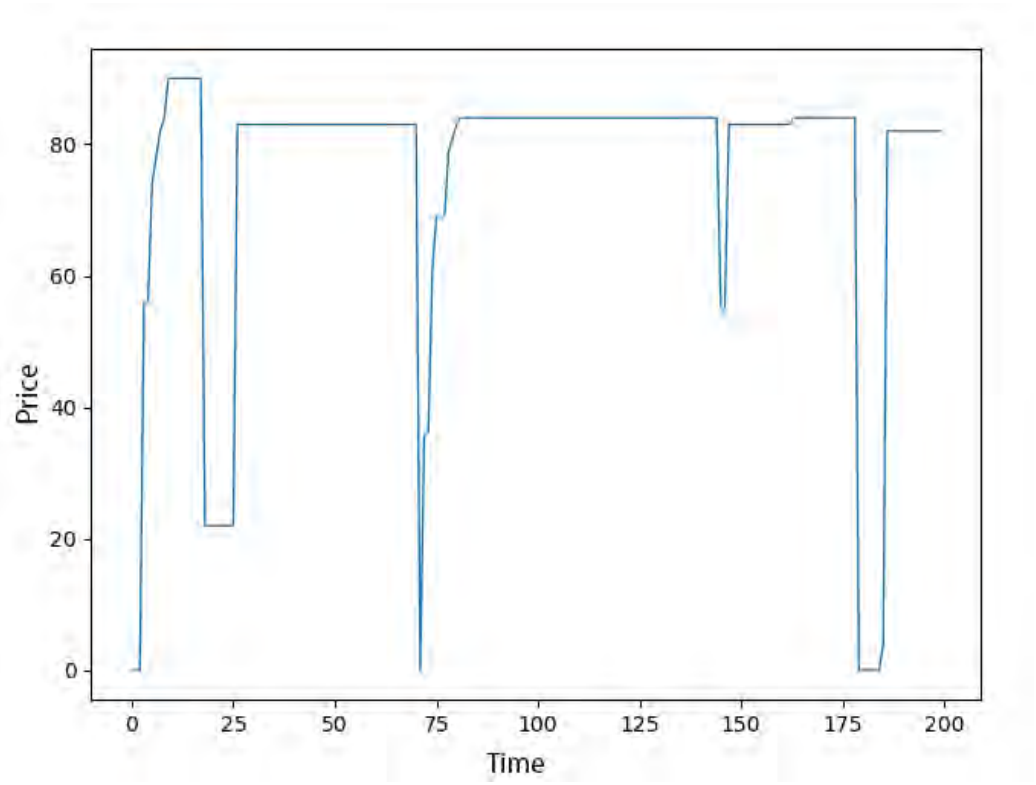


Figure 5.1: [Buy, Hold, Sell] share (2^{nd} phase)

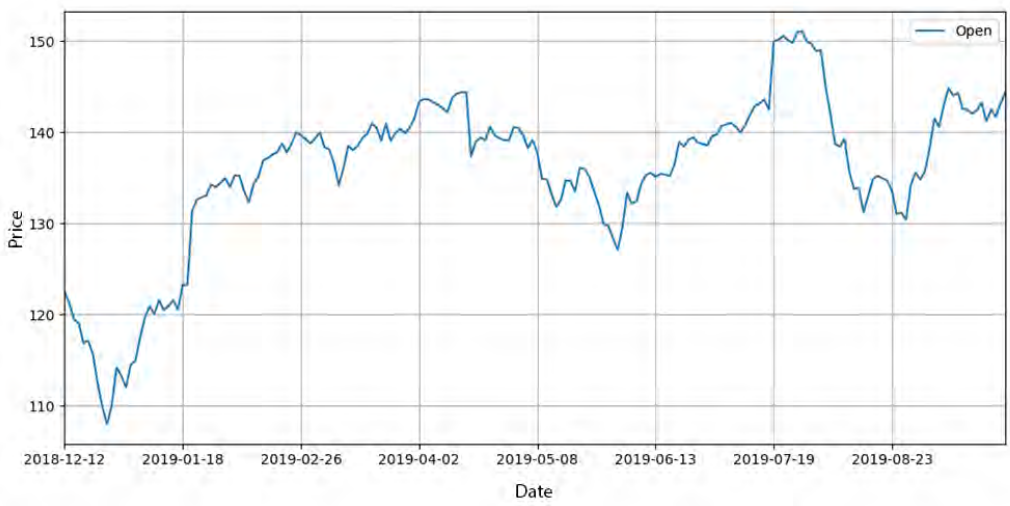


Figure 5.2: Stock market data (input of 2^{nd} phase)

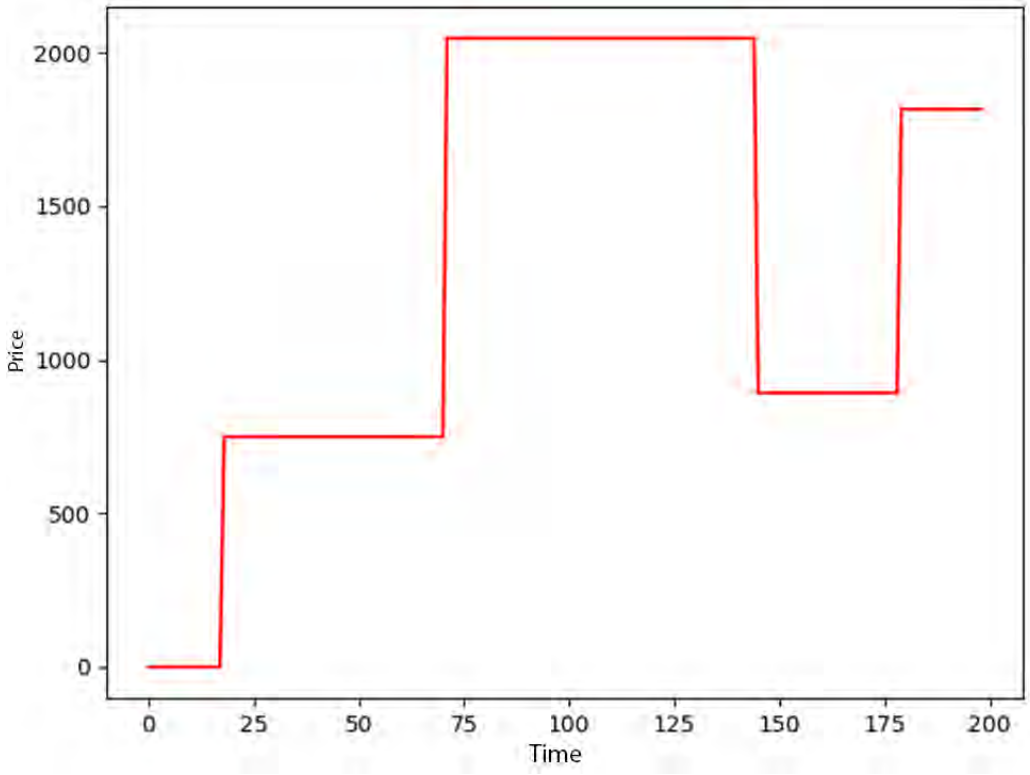


Figure 5.3: Profit over time (2^{nd} phase))

Based on the findings of 2^{nd} phase it was decided to change the hyperparameter values and used a new set of indicators manually. Previously, The default values of the hyperparameter were used.

Hyperparameter	Value
Horizon (T)	2048
Num. epochs	10
Minibatch size	64
Discount (γ)	0.99
GAE parameter (λ)	0.95
Clip Range	0.2
Learning Rate	0.00025

Table 5.4: Default Hyperparameter of PPO

After the optimization of the values,we get the values given in the below table

Hyperparameter	Value
Horizon (T)	2048
Num. epochs	16
Minibatch size	128
Discount (γ)	0.91
GAE parameter (λ)	0.94
Clip Range	0.2
Learning Rate	0.0021

Table 5.5: PPO Hyperparameters after optimization

In this phase(Final Phase) we used total 8 sets of indicators and we have observed that if we take ['volume_cmf', 'volume_fi', 'volume_em','volatility_kcc', 'momentum_ao', 'momentum_tsi', 'others_dr, 'momentum_roc'] as the set of observation space alongside with the five basic observation space it get the best result and most stable model.This model outperform the model of the second phase too. Graphical representation of the profit,the data used and the buy,sell,hold is given below

Figure 5.4: Apple Data (2018-2019)

The figure represents the input data for the algorithm and this model is using IBM 200 days stock data.

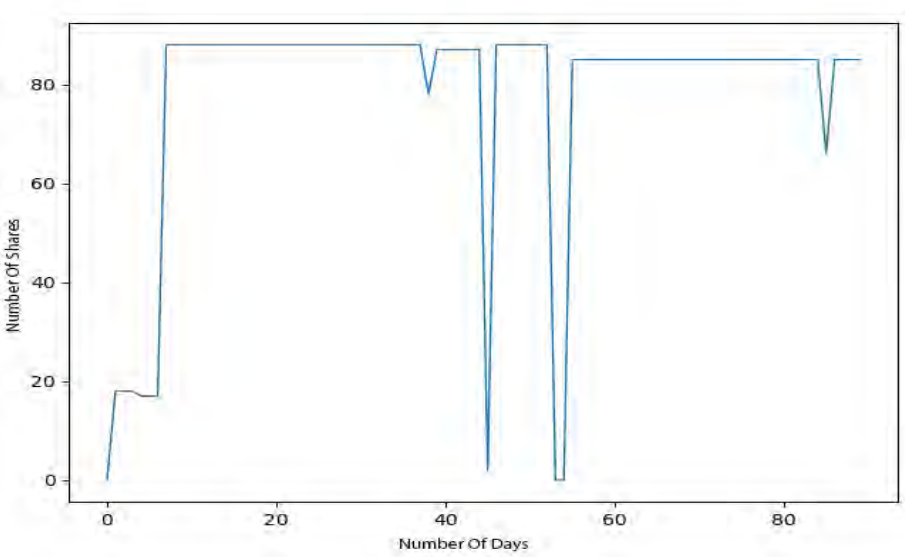


Figure 5.5: Buy,Sell,Hold

Figure: represents about the corresponding graph for buy sell and hold 0 to 600 means at this day selected algorithm decided to buy 600 stock here in this graph x= number of days, y = Number of shares.

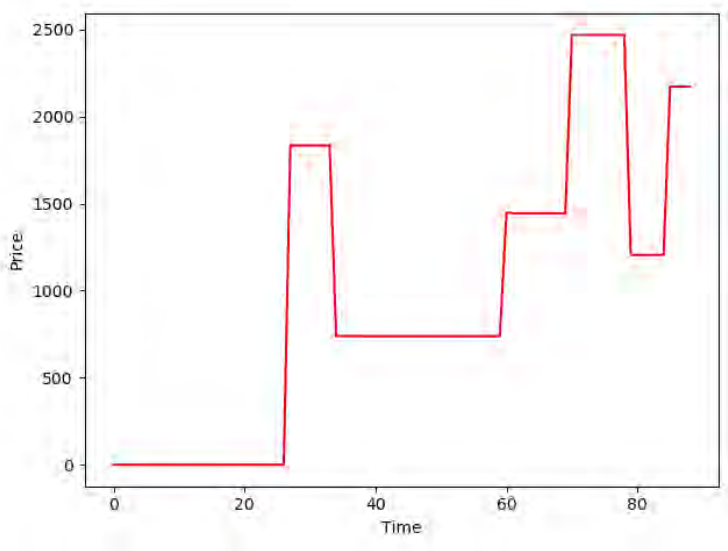


Figure 5.6: Profit over time

Figures of the indicator used in the best scenario is given below

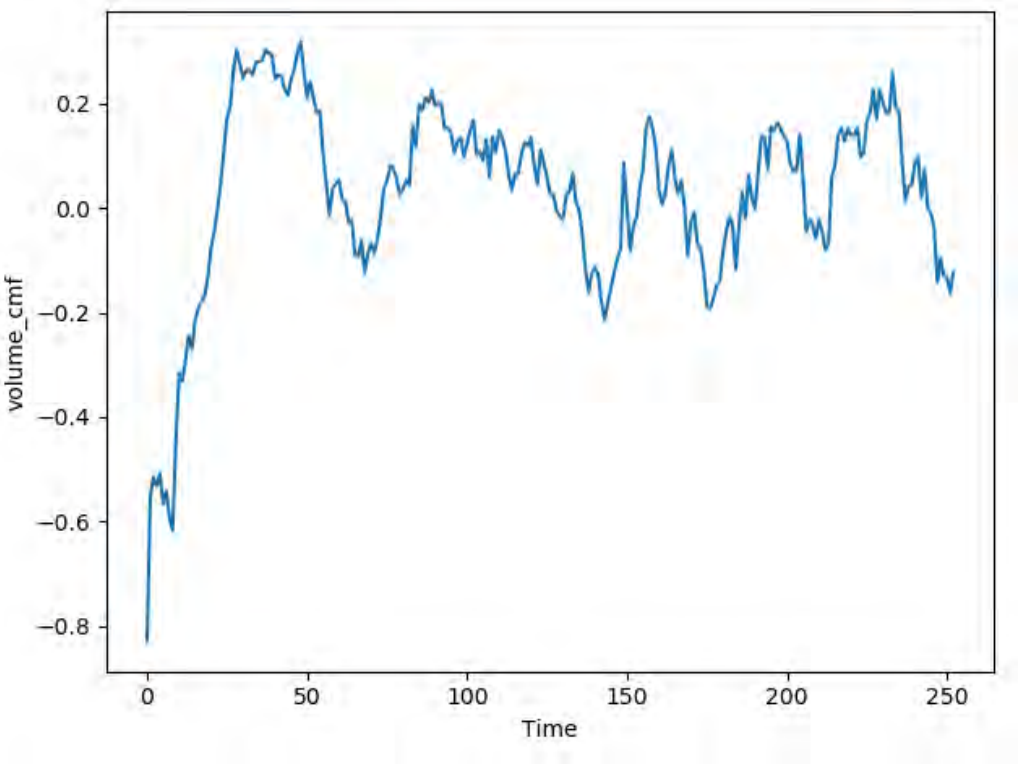


Figure 5.7: Chaikin Money Flow

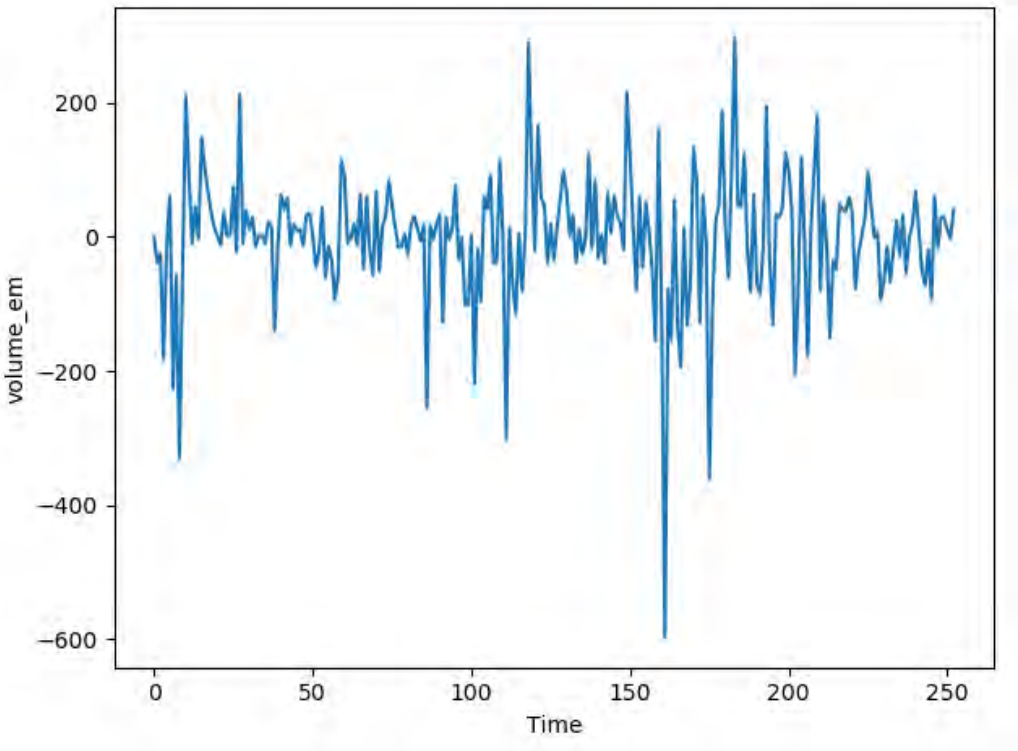


Figure 5.8: Ease of Movement

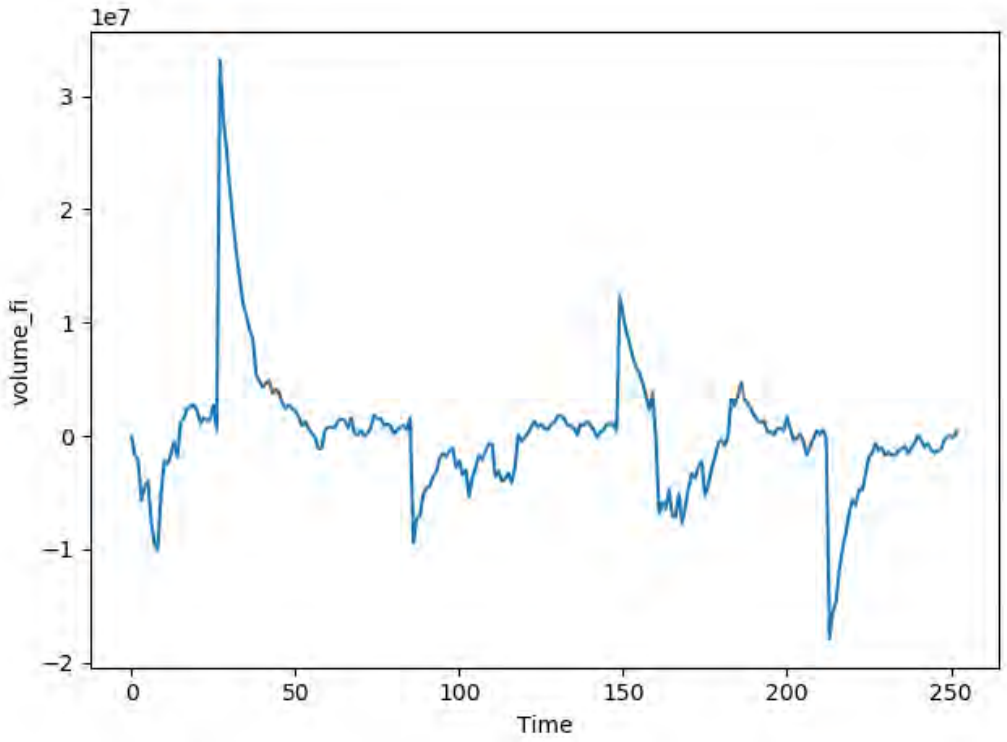


Figure 5.9: Force Index

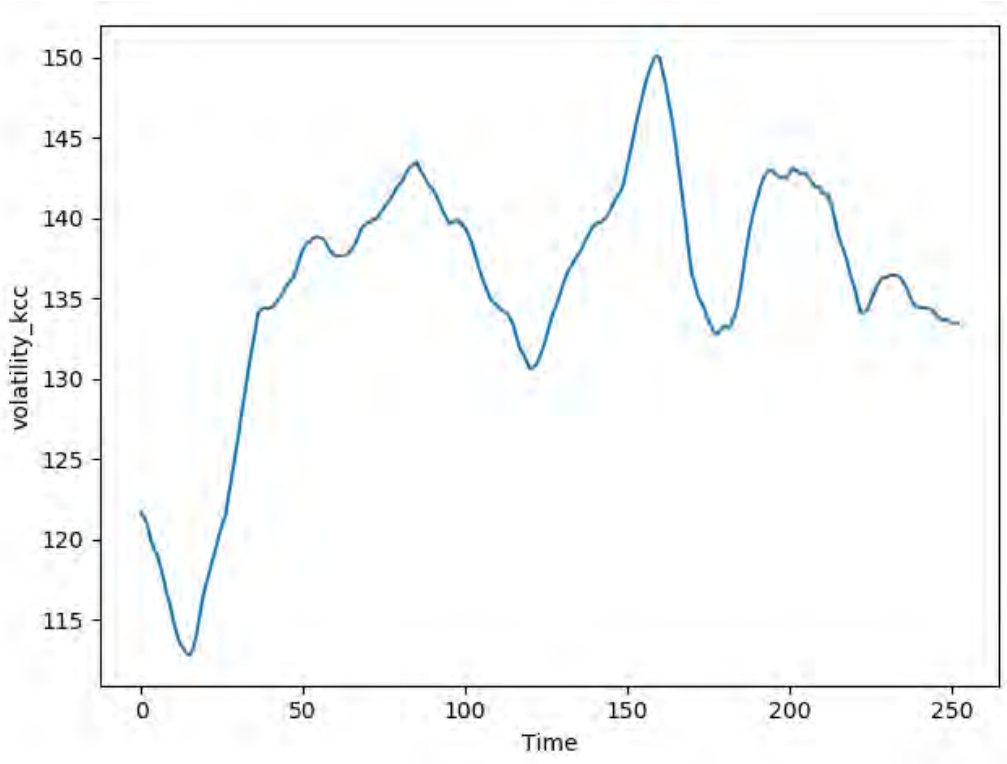


Figure 5.10: Keltner Channel

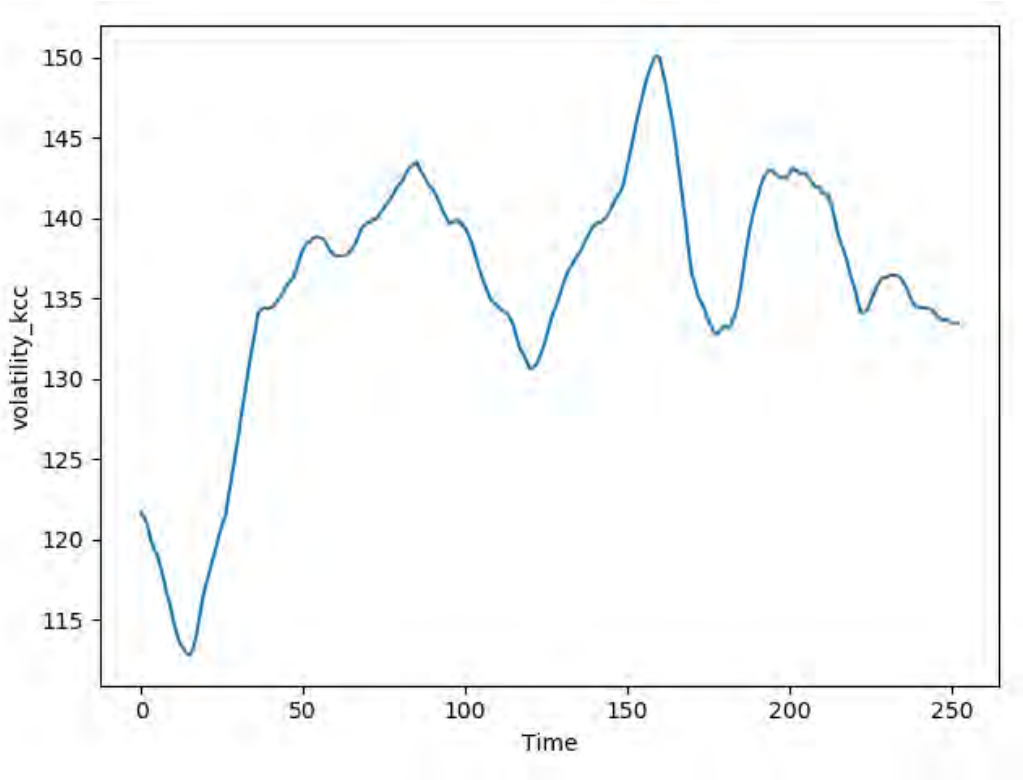


Figure 5.11: Awesome Oscillator

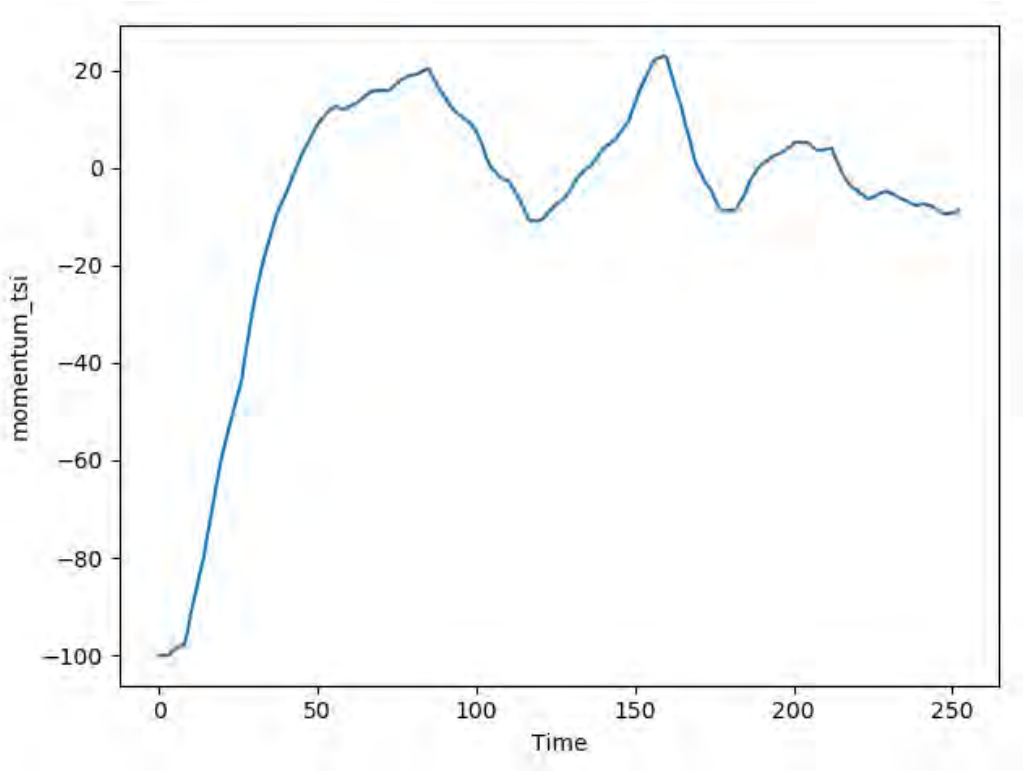


Figure 5.12: True Strength Index

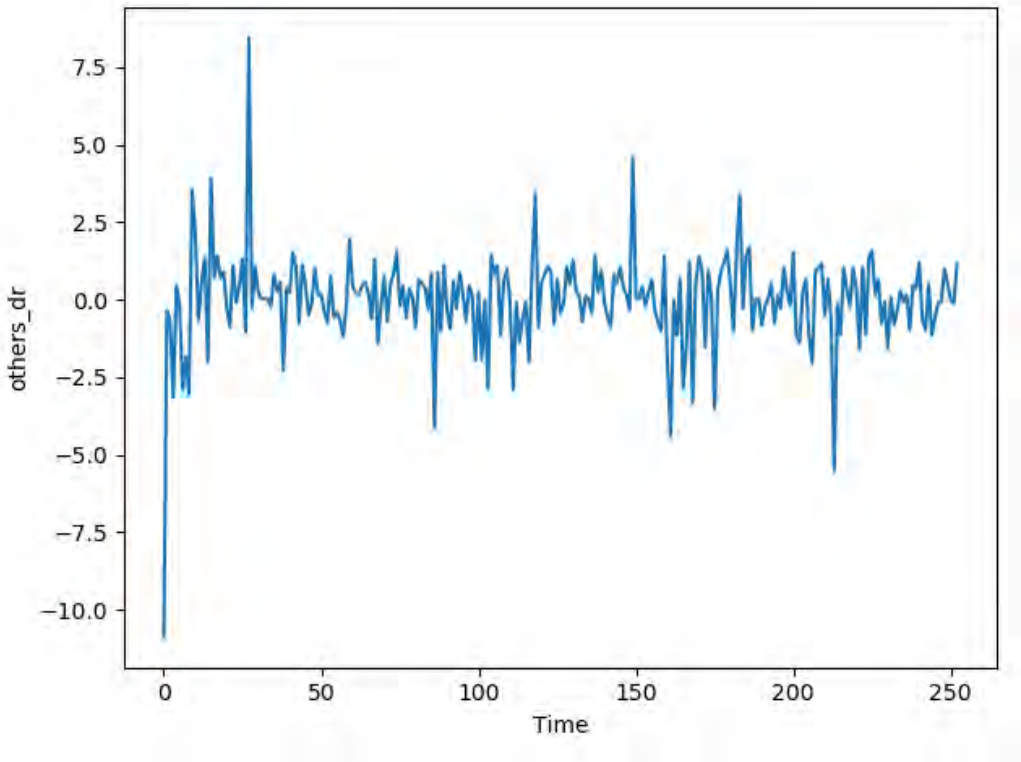


Figure 5.13: Daily Return

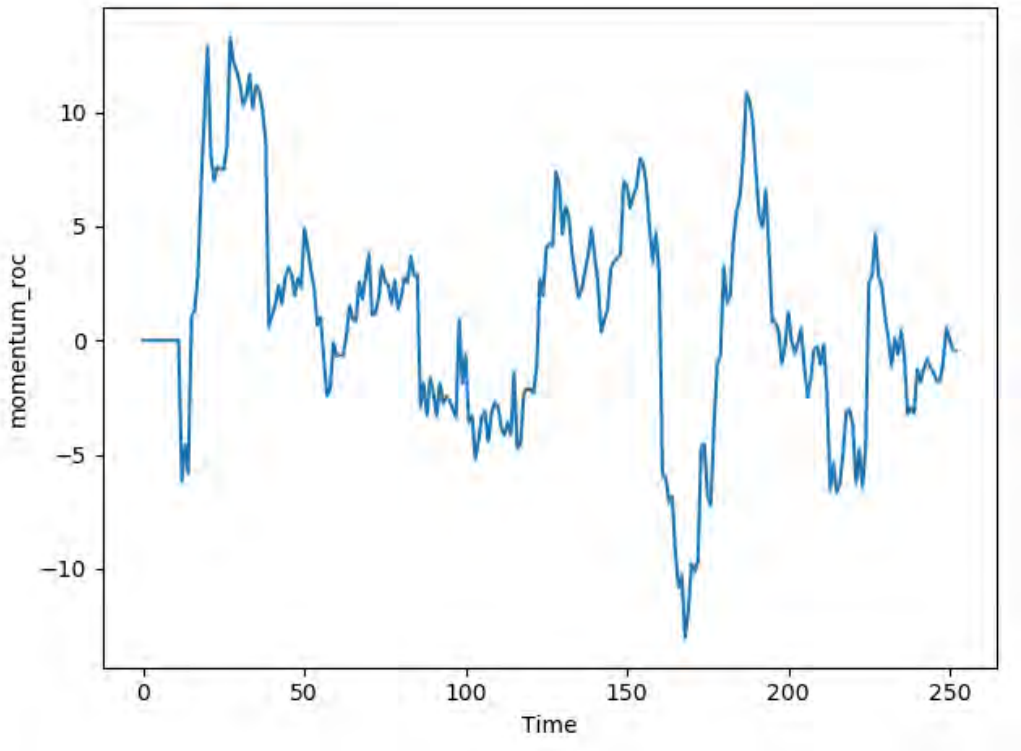


Figure 5.14: Rate of change

Chapter 6

Conclusion

6.1 Conclusion

The effort was to make an agent that can trade in the financial market using reinforcement learning. Due to cost and efficiency, it has used OpenAIGym to simulate a finical Market environment. This neural structure consists of two 64 hidden layers with one input and output layer. Applied algorithm is trained with AAPL, MSFT stock data set which was taken from Yahoo Finance. For the basic structure of (RL), last 5 days data was used and scaled between 0-1. Share buy/sell/hold is decided as an action for the agent. For the most important reward function, it has used account balance multiplied by delay modifier. First, A2C algorithm was applied but not satisfied with that. Then it comes the PPO algorithm. With normal Colum's of the dataset indicators for stable observation is added. There are 50 different sets of indicators which are used with each set iterated for 10 times. Finally, for the better result, manual implementation of the indicators with the real environment and then compared it with previous outcomes was done.

6.2 Future Plan

With the limited domain of knowledge in reinforcement learning and finance and by the guidance of the respected supervisor, this model has come to this point. In the future, Multi-agent Reinforcement learning can be used in order to get better performance. It creates a more accurate model by comparing between themselves. Think about the game GO. Contesting between the agent and more training it get best strategy model precise in steps for this game. Moreover, Inverse reinforcement learning can be applied. By this IRL it is possible to extract reward function from the observed behaviour of an agent. Neural Architecture search can be a better option for the optimal neural network. There are also effects of political decision, pandemics, weather in the share market. Future work should put emphasis on it how those factor in the model can give more accuracy.

Bibliography

[1] J. Moody, L. Wu, Y. Liao, and M. Saffell, "Performance functions and reinforcement learning for trading systems and portfolios", *Journal of Forecasting*, vol. 17, no. 5-6, pp. 441–470, 1998.

[2] J. E. Moody and M. Saffell, "Reinforcement learning for trading", in *Advances in Neural Information Processing Systems*, 1999, pp. 917–923.

[3] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning.", in *Icml*, vol. 1, 2000, p. 2.

[4] J. W. Lee, "Stock price prediction using reinforcement learning", in *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)*, IEEE, vol. 1, 2001, pp. 690–695.

[5] M. A. Dempster and V. Leemans, "An automated fx trading system using adaptive reinforcement learning", *Expert Systems with Applications*, vol. 30, no. 3, pp. 543–552, 2006.

[6] O. Jangmin, J. Lee, J. W. Lee, and B.-T. Zhang, "Adaptive stock trading with dynamic asset allocation using reinforcement learning", *Information Sciences*, vol. 176, no. 15, pp. 2121–2147, 2006.

[7] M. Olsson and L. Soder, "Modeling real-time balancing power market prices using combined sarima and markov processes", *IEEE Transactions on Power Systems*, vol. 23, no. 2, pp. 443–450, 2008.

[8] J. J. Choi, D. Laibson, B. C. Madrian, and A. Metrick, "Reinforcement learning and savings behavior", *The Journal of finance*, vol. 64, no. 6, pp. 2515–2534, 2009.

[9] E. Hurwitz and T. Marwala, "Suitability of using technical indicator-based strategies as potential strategies within intelligent trading systems", in *2011 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, 2011, pp. 80–84.

[10] T. Matsui, T. Goto, K. Izumi, and Y. Chen, "Compound reinforcement learning: Theory and an application to finance", in *European Workshop on Reinforcement Learning*, Springer, 2011, pp. 321–332.

[11] Z. Tan, C. Quek, and P. Y. Cheng, "Stock trading with cycles: A financial application of anfis and reinforcement learning", *Expert Systems with Applications*, vol. 38, no. 5, pp. 4741–4755, 2011.

[12] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization", *CoRR*, vol. abs/1502.05477, 2015. arXiv: 1502.05477. [Online]. Available: http://arxiv.org/abs/1502.05477.

[13] S. Y. Yang, Q. Qiao, P. A. Beling, W. T. Scherer, and A. A. Kirilenko, "Gaussian process-based algorithmic trading strategy identification", *Quantitative Finance*, vol. 15, no. 10, pp. 1683–1703, 2015.

[14] S. R. Young, D. C. Rose, T. P. Karnowski, S.-H. Lim, and R. M. Patton, "Optimizing deep learning hyper-parameters through an evolutionary algorithm", in *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments*, 2015, pp. 1–5.

[15] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading", *IEEE transactions on neural networks and learning systems*, vol. 28, no. 3, pp. 653–664, 2016.

[16] S. Almahdi and S. Y. Yang, "An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown", *Expert Systems with Applications*, vol. 87, pp. 267–279, 2017.

[17] A. Koutsoukas, K. J. Monaghan, X. Li, and J. Huan, "Deep-learning: Investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data", *Journal of cheminformatics*, vol. 9, no. 1, p. 42, 2017.

[18] D. W. Lu, "Agent inspired trading using recurrent reinforcement learning and lstm neural networks", *arXiv preprint arXiv:1707.07338*, 2017.

[19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms", *CoRR*, vol. abs/1707.06347, 2017. arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347.

[20] ——, "Proximal policy optimization algorithms", *arXiv preprint arXiv:1707.06347*, 2017.

[21] J. Carapuço, R. Neves, and N. Horta, "Reinforcement learning applied to forex trading", *Applied Soft Computing*, vol. 73, pp. 783–794, 2018.

[22] Z. Xiong, X.-Y. Liu, S. Zhong, H. Yang, and A. Walid, "Practical deep reinforcement learning approach for stock trading", *arXiv preprint arXiv:1811.07522*, 2018.

[23] A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, and J. Clune, "Go-explore: A new approach for hard-exploration problems", *arXiv preprint arXiv:1901.10995*, 2019.

[24] R. Batra, "The Anatomy of Stock Market Bubbles and Crashes", in *COMMON SENSE MACROECONOMICS*, ser. World Scientific Book Chapters, World Scientific Publishing Co. Pte. Ltd., Oct. 2020, ch. 10, pp. 215–245. [Online]. Available: https://ideas.repec.org/h/wsi/wschap/9781786348401_0010.html.

[25] A. Charpentier, R. Elie, and C. Remlinger, "Reinforcement learning in economics and finance", *arXiv preprint arXiv:2003.10014*, 2020.