Applying Reinforcement Learning on Automated Cryptocurrency Trading

COMP4971C - Independent Work (Fall 2020)

December, 2020

CHONG, Cheuk Hei

Supervised by Dr. David Rossiter

Department of Computer Science and Engineering, HKUST

Abstract

In this research project, we examine the feasibility of automated cryptocurrency trading using reinforcement learning in order to learn an optimal policy by the machine itself. Technical indicators will also be added into the model to increase the chance of performing better action in every state. Evaluation on the model performance will also be conducted to verify the feasibility of implementing trading bot with reinforcement learning in the practical scenerio.

Contents

1 Introduction

With the rise of blockchain technology in recent years, more people are discussing on cryptocurrencies and starting to enter the trading market on cryptocurrencies. With its high volatility on price and 24/7 trading time characteristics, trading on cryptocurrencies might be a high-risk assessment when people do not have self-disciplines to construct strict policies on trading and do not monitor the pricing movement from time to time. Huge profit loss might occur to those FOMO traders when they observe the prices being decreasing rapidly.

Therefore, if people would like to use the optimal efforts to earn profits on cryptocurrency trading, it is necessary to implement automated trading algorithm which can execute buy, sell and hold actions at the right time by adopting "Buy low, sell high" strategy. In this project, we will adopt the concepts of reinforcement learning on trading. The well-trained model will replace humans to perform actions without affected by feelings. We will also examine the prediction power of reinforcement learning and evaluate its effectiveness quantitatively.

2 Disclaimer

The information presented in this research is not intended as, and shall not be understood as financial advice to enter in any security transactions or to engage in any of the investment strategies.

3 Related Work

There are some approaches which apply different methods on the price prediction of cryptocurrency. Based on those approaches, people will get the trend of the price movement or the prediction price of certain cryptocurrency by the well-trained algorithm. The following are some approaches to work on the trading price prediction:

3.1 Technical Analysis

Technical Analysis is a way to forecasting the general movement of the price in the future based on the previous price movements, which can be applied on any trading instruments. Chart analysis is frequent nowadays to know the trend of the price movement and people have invented different technical indicators to understand the price stock in a quantitative way. The following are the examples of some basic technical indicators:

Technical Indicators	Types	Description
Moving Average Convergence Divergence(MACD)	Trend	show the relationship between two moving averages on a price and know the momentum is increasing or decreasing
Relative Strength Index(RSI)	Momentum	measures the speed of the pricing movement and the trading strength, helps evaluate whether is overbought or underbought $$RSI = 100 - \frac{100}{1 + \frac{n_{up}}{n_{down}}} \quad (1)$$
Bollinger Bands(B.B.)	Volatility	define the upper and the lower rate boundaries of price in extreme stort-term, can take advantage during oversold condition $$UpperBB = MA + D\sqrt{\frac{\sum_{i=1}^{n}(y_j - MA)^2}{n}} \quad (2)$$ $$LowerBB = MA - D\sqrt{\frac{\sum_{i=1}^{n}(y_j - MA)^2}{n}} \quad (3)$$
On-balance Volume(OBV)	Volume	utilize the flow of trading volume to predict stock price change, bullish divergence and bearish divergence can predict whether it will break resistance $$OBV = OBV_{prev} + \begin{cases} vol, \ if close > close_{prev} \\ 0, \ if close = close_{prev} \\ -vol, \ if close < close_{prev} \end{cases} \quad (4)$$
Average True Range(ATR)	Volatility	discover the degree of price volatility $$ATR = \frac{1}{n}\sum_{i=1}^{n} TR_i \quad (5)$$

However, only using technical analysis is not adequate for earning profits as it might be too late to reflect the trend. For example, MACD, B.B. are the lagging indicators which a substantial portion of moving has been executed when those indicators are just starting to reflect the trend. Also, human bias might be invlvoed when analysing. It is possible to use different indicators to conclude different insights even with the same chart. Therefore, technical analysis is just a reference for the traders.

3.2 RNN

RNN stands for Recurrent Neural Network, which the nerual network will take the output from the previous step as the input of the current step to make decisions. It is commonly used in predictng sequential data and applied on Nautral Language Processing(NLP), stock prediction, and so on.

In RNNs, there is an "internal" state which will be updated when the sequence is processed. At first, sequence of input vector x will be fed into each RNN cell. Recurrence formula will be then applied to every time step, which will use the input x_t and the old state h_{t-1} as the parameters to evaulate the new state in the next RNN cell:

$$h_t = f_W(h_{t-1}, x_t) \tag{6}$$

Assume that we used tanh as the activation function, therefore the value of internal state h_t and output y_t can be expressed as:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \tag{7}$$

$$y_t = W_{hy}h_t \tag{8}$$

This procedure will continue until the all the time step is completed. Finally, backpropagation will be performed from state h_t to state h_{t-1} multiplying by W_{hh}^T. As RNN can store every information in each step time, it is commonly used on predicting time-series data. However, with computing the gradient, it invloves many W and repeated $tanh$ calculation in each time step. The gradient will vanish rapidly at last ($||\frac{\partial h_t}{\partial h_{t-1}}|| < 1$) and parameter update is not significant at all. Therefore, RNN is diffcult to process data with long sequences. Trading price prediction invloves long sequential data which is not desirable to adopt it.

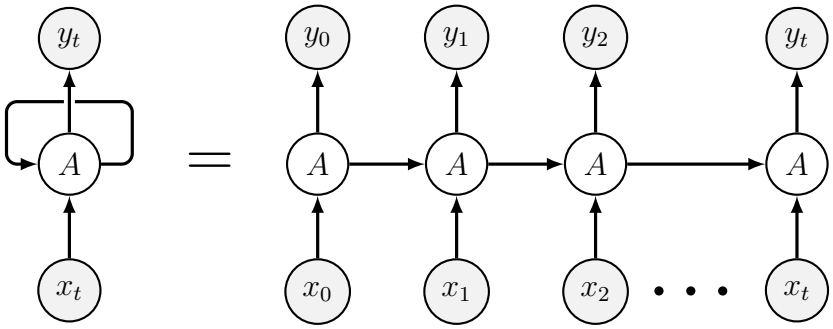

Figure 1: Workflow of Recurrent Neural Network (RNN)

3.3 LSTM

LSTM stands for Long Short Term Memory networks. It is a type of Recurrent Neural Network (RNN) which solves the gradient vanishing problem and it is capable of learning long-term sequential data. LSTM have a similar structure with RNN but LSTM have a

new cell state and gates to control the data flow.

Cell State c. It allows the information in the past several time steps passing through the network being remain unchanged.

Forget Gate f. It uses to determine whether the information will be earsen in the cell state. Data flow is controlled by the sigmoid function.

$$f_t = \sigma(W_f[h_{t-1}, x_t]) \tag{9}$$

Input Gate i. It is with a sigmoid layer which decides whether to write to the cell. Tanh layers will be then passed through to calculate $\hat{c}_t$ which will be used to update the cell state. Finally, C_t is calculated to replace the old cell state.

$$i_t = \sigma(W_i[h_{t-1}, x_t]) \tag{10}$$

$$\hat{c}_t = \tanh(W_c[h_{t-1}, x_t]) \tag{11}$$

$$c_t = f_t c_{t-1} + i_t \hat{c}_t \tag{12}$$

Output Gate o. It is to decide how much to reveal the cell. After passing the sigmoid leyer, it will be multiplied by the value of existing cell state with passing the tanh layer to calculate hidden state h_t.

$$o_t = \sigma(W_o[h_{t-1}, x_t]) \tag{13}$$

$$h_t = o_t \tanh(c_t) \tag{14}$$

For backpropagation, as the gradient will pass backward from c_t to c_{t-1} which do not need to do matrix multiplication with W, the gradient flow is uniterrupted so that it can prevent the gradient vanishing problem.

With this advantage, most of the trading prediction applications are using with LSTM. However, LSTM is just for predicting the price but wihtout the invlovement of experience replay. Therefore, we are going to use reinforcement learning approach to evaulate the effectiveness of each action taken in trading.

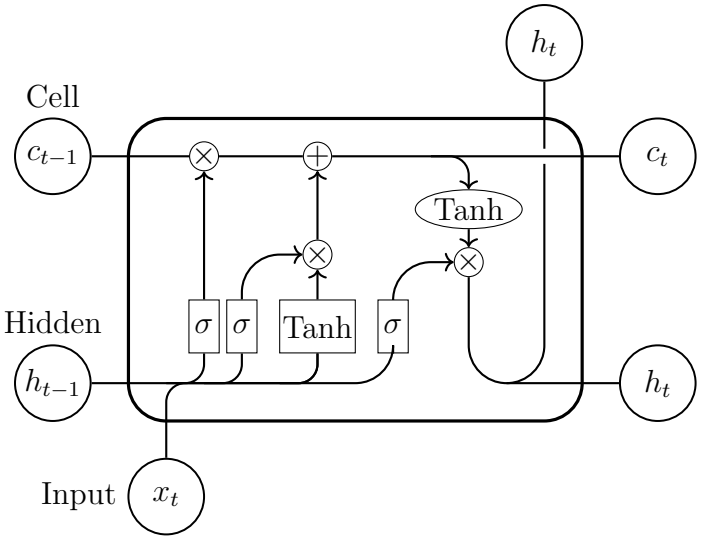

Figure 2: Workflow of Long Short Term Memory(LSTM)

4 Data

Before the introduction of the reinforcement learning methodology, data preparation process is with the same importance which can be explained into different parts.

4.1 Finding Price History Datasets

In this project, we are planning to consider the most popular cryptocurrencies in the portfolio, Bitcoin(BTC), which is in large trading volumes so that the price movement will not controlled by someone easily. The daily price movements of BTC has been collected in Yahoo Finance in CSV format and the data time range is from Aug 2015 to Dec 2020. The data includes the daily open price, close price, low price, high price, adj. close and the volume. For simplicity, we will consider the close price and the volume in this project.

Date	Open	High	Low	Close	Adj Close	Volume
2015-08-07	278.740997	280.391998	276.365997	279.584991	279.584991	42484800
2015-08-08	279.742004	279.928009	260.709991	260.997009	260.997009	58533000
2015-08-09	261.115997	267.002991	260.467987	265.083008	265.083008	23789600
2015-08-10	265.477997	267.032013	262.596008	264.470001	264.470001	20979400
2015-08-11	264.342010	270.385986	264.093994	270.385986	270.385986	25433900

Figure 3: Data Sample of BTC price from Yahoo Finance, from 07/08/2015 to 17/12/2020

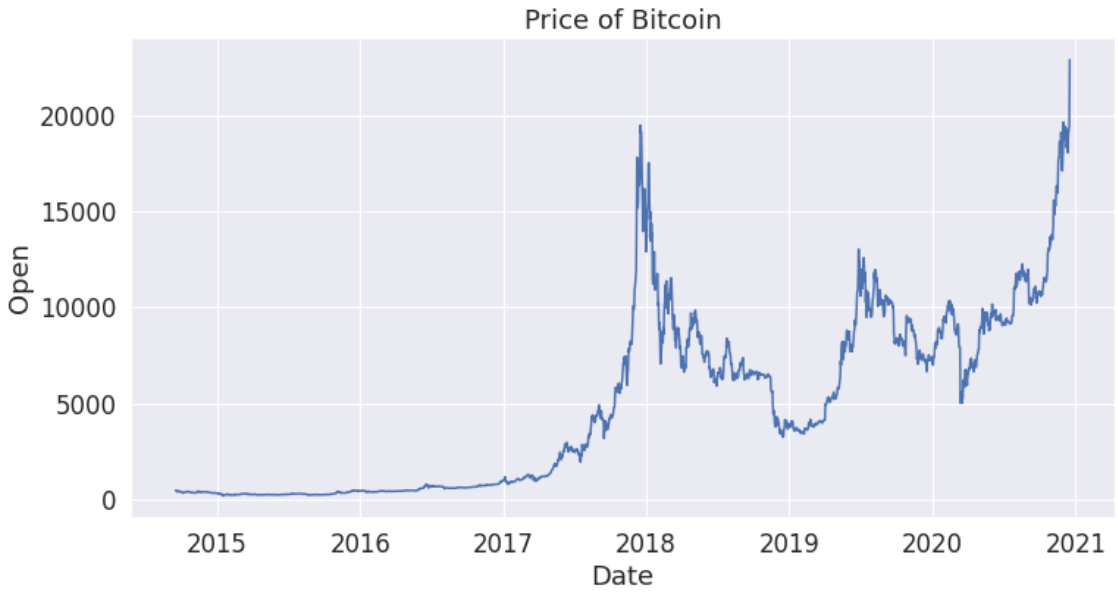

Figure 4: Price of Bitcoin, from 07/08/2015 to 17/12/2020

4.2 Time Interval of Dataset

In the initial plan, as the cryptocurrency price is more volatile than stock market, it is aimed to get price movement with shorter time frame. However, with the consideration of the limited access of history data and extreme high training time, the price dataset is in daily interval which simplifies the training process.

4.3 Adding Technical Indicators

Pricing data is not adequate to learn the patterns of the movement by the neural network of the agent, therefore we will calculate some relevant technical indicators based on the data in Yahoo Finance and add the values of those indicators as the input of the neural network. The included indicators are as follows:

Technical Indicators	Type
Moving Average Convergence Divergence(MACD)	Trend
Relative Strength Index(RSI)	Momentum
Bollinger Bands(B.B.) Low	Volatility
Bollinger Bands(B.B.) High	Volatility
On-balance Volume(OBV)	Volume
AverageTrueRange(ATR)	Volatility

Those indicators are from different types, some can analyze the trend of the movement, volatility of the pricing, and volume. Those data gives more clear image to the nerual network to learn the relationship between the movement trend and the price.

4.4 Data Normalization

It is needed to bear in mind that states of the agent, including the number of shares owned, coin prices and cash are in different ranges. With different ranges of value, the gradients may oscillate back and forth in the neural network which will affect the agent performance. This might cause the agent not performing the optimal actions and rewards might be lowered. With data normalization, we will standardize the data by removing the mean and scaling into unit variance. In this project, `sklearn.preprocessing.StandardScaler` has been used to perform the normalization. It can improve the gradient flow which makes the network much easier to train, and increase the rewards by performing optimal actions in each state.

$$z = \frac{x - \mu}{\sigma} \tag{15}$$

5 Methodology

In the "Related Work" session, we can see some interesting approaches working on the price prediction of the cryptocurrency. Although the result could tell humans whether the price will go up or down in the future, if we would like to implement a fully-automated robot which authorize the right to execute actions on "buy", "sell" or "hold", machine learning approaches might not be the best solution as it still needs to involve human's interruption. Therefore, this project is going to implement the automation with reinforcement learning.

5.1 Advantages of Reinforcement Learning

Reinforcement learning is a kind of simulation of human beings. When facing with unfamiliar enivronment, it will first try the possible actions randomly. After getting more experience, it will adjust the policy to correct the error made before and execute a more optimal action in the next state. The alogrithm will then keep improving with more training loops. After training, we can also know how the agent behaves in each state. Comparing with other prediction methods, RL can solve complex problems which traditional one would not be able to solve.

Cryptocurrency trading is a good example to implement RL as it could teach humans when is the best time to "buy" or "sell" the coin. Also, as pricing pattern could be changed in the future, RL can explore more possibilities which do not encounter in the previous data by learning from the mistake.

5.2 Architecture of Reinforcement Learning

Reinforcement Learning (RL) is the training of a model which machine can make different decisions at certain states. The role of the model can be explained as an agent to overcome a game-like problem. Unlike the RNN/LSTM approach which predicts the future results, The ultimate goal of the RL is to take actions at a right time for maximizing the rewards under a specified environment.

The reinforcement learning involves the following terms:

- Agent: responsible on performing actions
- Environment: The world setting for the agent to perform actions
- State: the current situation faced by the agent
- Action: set of actions which can be performed in the environment
- Rewards: a scalar feedback which reflects how well the action performs at a state s
- Neural Network: a process of learning on making decision used by the agent

For the workflow of the reinforcement learning, an environment will be first initialized for the agent to do observation. After that, states relevant to the environment will act as the input of the agent. After "thinking" by the neural network in the agent, agent will take an action according to its policy setting. The environment will be changed by the action executed and a reward value will be sent to the agent to know whether it is a good decision so as to update its current policy.

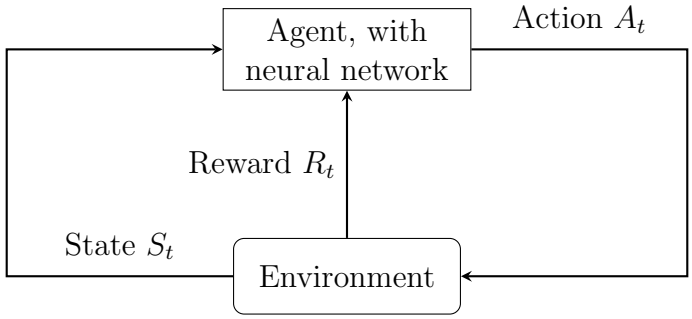

Figure 5: Illustration of Reinforcement Learning Workflow

5.3 Model Settings

5.3.1 Environment

In this project, in order to simplify the formation of investment portfolio, therefore in the environment setting, we assume that a user usually trade with only one type of cryptocurrencies at the same time, which is Bitcoin(BTC). However, commission fee will be also considered in the simulated environment.

5.3.2 State

State can be summarized into 7 types:

- Number of shares of BTC owned
- Open price of BTC
- Technical Indicator RSI
- Technical Indicator MACD
- Technical Indicator BB High
- Technical Indicator BB Low
- Technical Indicator OBV
- Technical Indicator ATR
- Cash remaining for purchasing more crpytocurrencies

For example, when we have 3BTC, the current price of the three cryptocurrency is $228.121, the value of those indicators are 39.667, -10.1075, 268.0265, 202.2588, -92971000, 9.9250 respectively, and the cash remaining is 0. We can combine those value into a vector of state: `[3, 228.121, 39.667, -10.1075, 268.0265, 202.2588, -92971000, 9.9250, 0]`. This set of vector will be fed into the neural network for training and it will explain more in the "Neural Network" part.

5.3.3 Action

Same with the real trading environment, we have set the actions into 3 types:

ID	Action	Description
0	Sell	$m = m + p_i * n_i * (1 - r), i = 1, 2, 3$
1	Hold	$m = m$
2	Buy	$m = m - p_i * n_i * (1 + r), i = 1, 2, 3$

- m: remaining cash
- r: rate of commission fee
- p_i: the price of each cryptocurrency
- n_i: the number of share of each cryptocurrency (minimum amount to execute the trade: 1)

It is understandable that users can partially buy or sell coins in the real world. However in this simulation. for selling, we have simplified the situation that the agent will sell all the existing shares of that coin that we own at once. For buying, the agent will buy that coin as many as possible unless cash is not sufficient. Therefore, we can explain this aggressive strategy as "all-or-nothing".

However, the commission rate `r` by exchange platform will be considered in this simulated environment. In summary, in each state, a set of actions can be expressed as: `[0], [1], [2]...`, which can form 3 possibilities to indicate actions perform on the coin.

It is also noticeable that the agent may not have sufficient money or number of share to perform "buy", "sell" actions in some scenarios, Therefore, if the remaining cash is not greater than the amount of buying the coins and the number of share is 0, no trading will be performed.

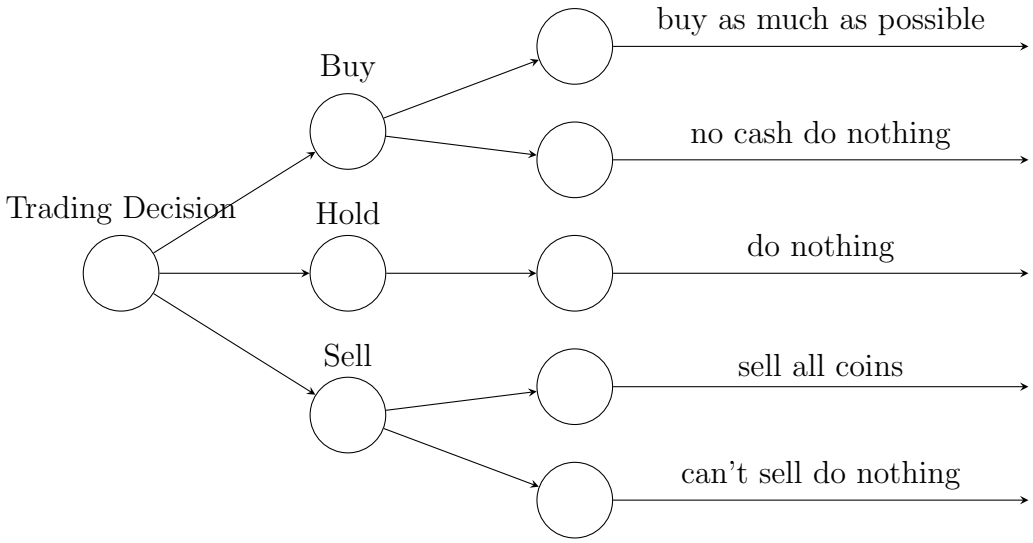

Figure 6: Illustration of Trading Decision

5.3.4 Rewards

The reward is the change in portfolio value between the current state and the previous state:

$$Reward(s, a, s') = (\sum_{i=0}^{2} n_i' * p_i' + m') - (\sum_{i=0}^{2} n_i * p_i + m) \tag{16}$$

5.3.5 Agent

The agent will follow the Epsilon-Greedy Algorithm. In this algorithm, we can decide which action to take in terms of exploration and exploitation.

$$\pi(s) = \begin{cases} \arg\max_{a \in Actions} Q(s, a), \ p = 1 - \epsilon \\ a \in actions, \ p = \epsilon \end{cases} \tag{17}$$

Exploration: agent will make an action which does not try before, with the probability ϵ. It can improve the current knowledge and increase the diversity of actions made by the agent in the long term.

Exploitation: agent will evaluate by using current action-value estimates and choose the most greedy action with getting the highest rewards. However, if it keeps using this approach to choose the action, that action might not be the most optimal one. Therefore, we have set the probability ϵ high initially to allow agents trying something new in the beginning. With getting more "experience", ϵ will decrease by multiplying a decay rate r so that the agent will choose the action by considering more on action-value estimates.

For the calculation of current action-value estimates, we will use the Bellman Optimality Equation:

$$Q(s,a) = \gamma \arg\max Q(s',a') + r(s,a) \tag{18}$$

which Q is the utility, γ is discount factor, r is reward.

5.3.6 Neural Network

The neural network is formed by a multi-layer perception, which consists of 8 inputs, corresponding to the 8 states on a single-cryptocurrency trading environment. A hidden layer with ReLU activation is created between the input layer and the output layer. The number of hidden neurons has configured to 64. Finally, output layer is formed which is corresponded to the three different actions defined previously. MSE will be used on calculating the loss and Adam will be used for the optimizer.

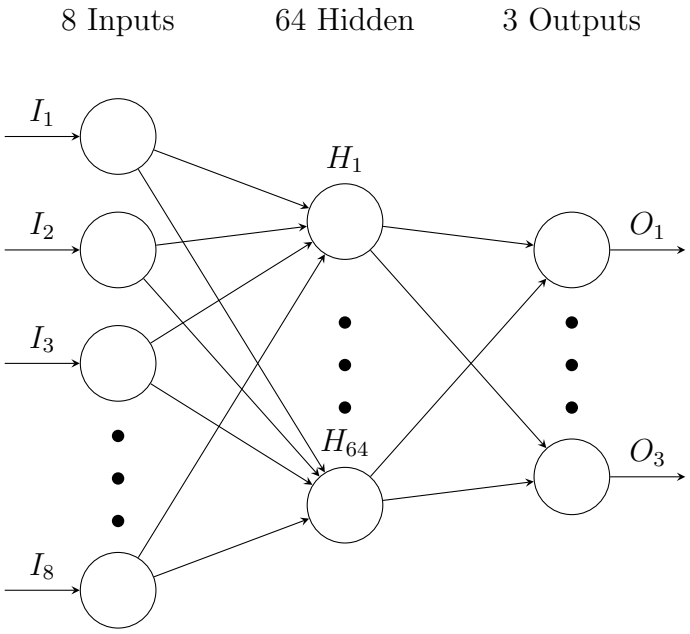

Figure 7: Illustration of the neural network architecture

6 Experiment

In this section, we will first train the datasets with our reinforcement learning algorithm, then proceed to analyze and evaluate the performance with using training and testing data. Results will be summarized quantitatively to determine the effectiveness of using reinforcement learning on trading.

6.1 Experiment Environment

All the process, including data prepossessing and reinforcement learning are operated in Google Colab Pro environment, configured with a Nvidia T4/P100 GPU, 16GB DDR6/ 12GB HBM2 memory capacity. Also, TensorFlow v2 has been adopted in order to construct the neural network for the agent during training.

6.2 Training and Testing Data

The history price data has been split into training data and testing data, which occupies 70 % and 30% of the total data respectively.

Data Type	Percentage(%)	Time Range
Training	70	2015-09-01 to 2019-04-20
Testing	30	2019-04-21 to 2020-12-17

6.3 Hyper-parameter Setting

The following table are the values of the hyper-parameter for the training:

Hyper-parameter	Value	Description
Money	`100000`	Initial capital
Epoch	`100`	Number of passing through the entire training dataset
Batch size	`32`	Number of samples propagating through the network
Discount rate γ	`0.95`	Importance of considering rewards in the distant future
Exploration rate ϵ	`1.0`	Probability of choosing actions in random ($\epsilon_{min} = 0.01$)
ϵ decay rate	`0.9027`	Decrease the probability of choosing actions in random, ϵ converged when epoch = 45
hidden layer size	`64`	Number of hidden neurons

6.4 Evaluation

6.4.1 Training Result

After training the agent within 100 epochs, the value of rewards increased steadily with some oscillations. However, the trend of the rewards is generally increasing. With \$100,000 investment in the beginning, the final profolio value is with an average of \$4577886.16 and a maximum of \$6363199.08, which the average reward is 45.78 times of the orginial values, which is an excellent result.

Average Rewards	Min Rewards	Max Rewards	Average APY
4577886.16	3012720.10	6363199.08	1208.9%

Figure 8: Trend of Training Data Rewards within 100 epochs

Looking at the actions taken on trading in different epochs, it is found that the agent has not ideas when to sell the BTC in the 10th epoch. However, with more epochs, it is learnt to sell BTC at a higher price to earn profits. However, it is needed to notice that the agent might be too quick to sell BTC in the later epochs when the price increased to about $19,000 at around late 2017.

All in all, the results are satisfactory in the training data which could reuse the model to test on the testing data.

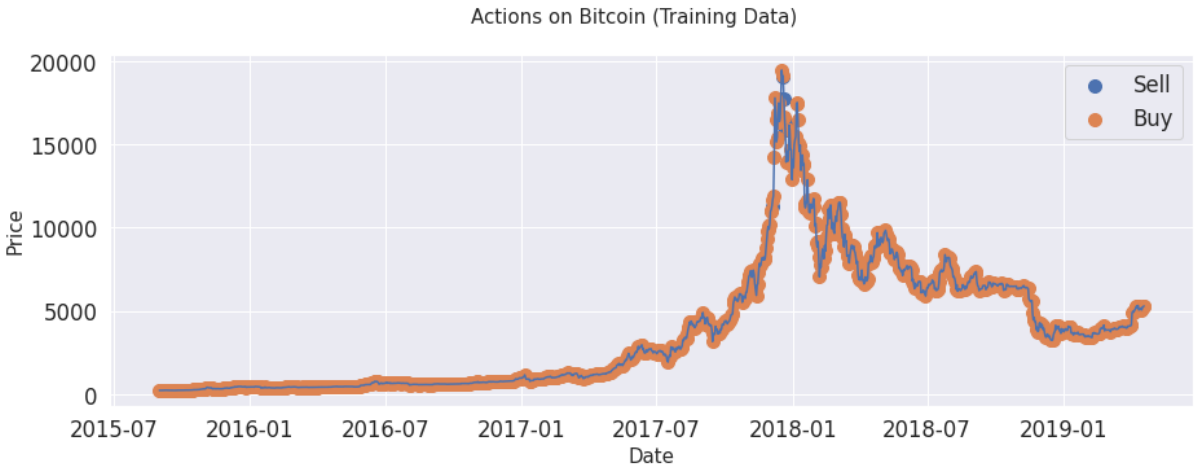

Figure 9: Actions on Bitcoin (Testing Data) in the 10th epoch

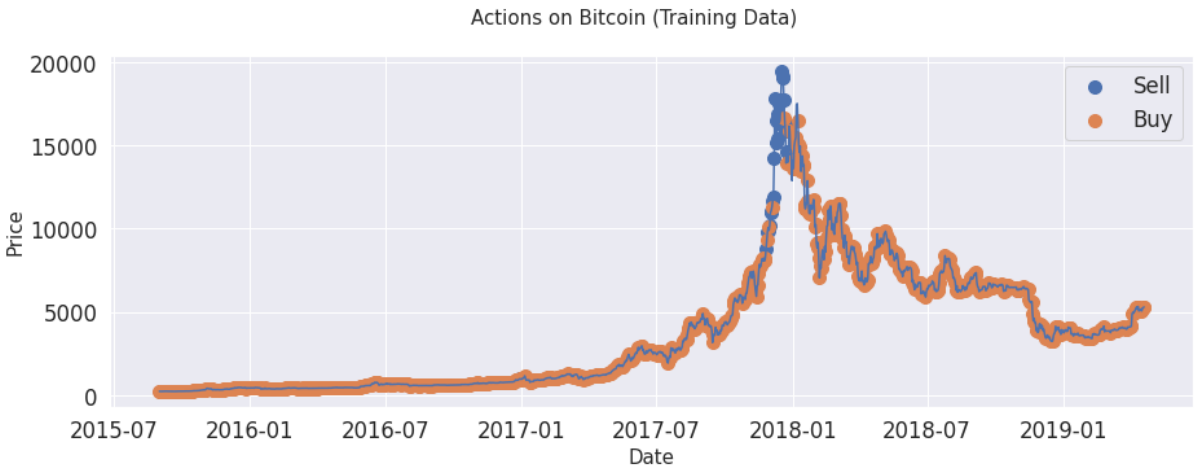

Figure 10: Actions on Bitcoin (Testing Data) in the 55th epoch

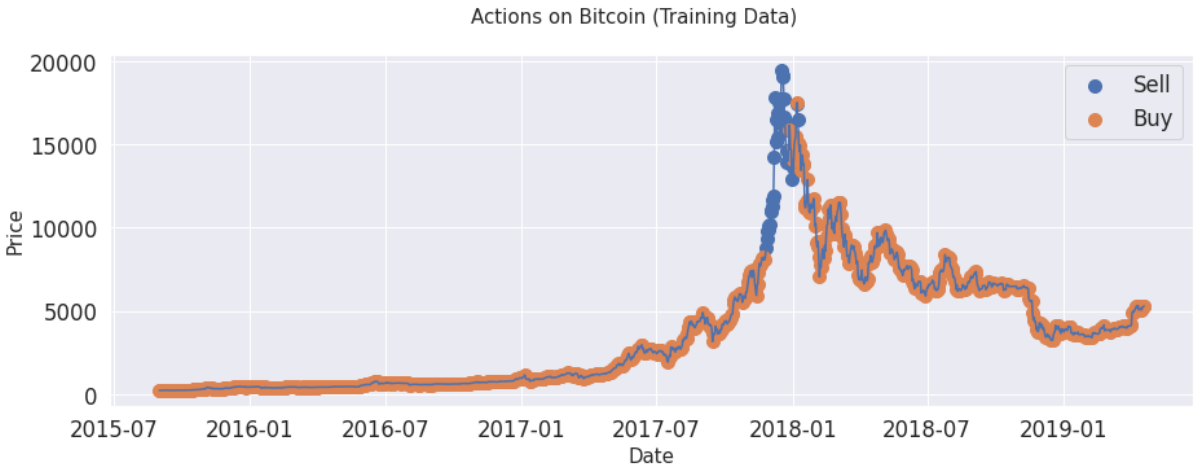

Figure 11: Actions on Bitcoin (Testing Data) in the 80th epoch

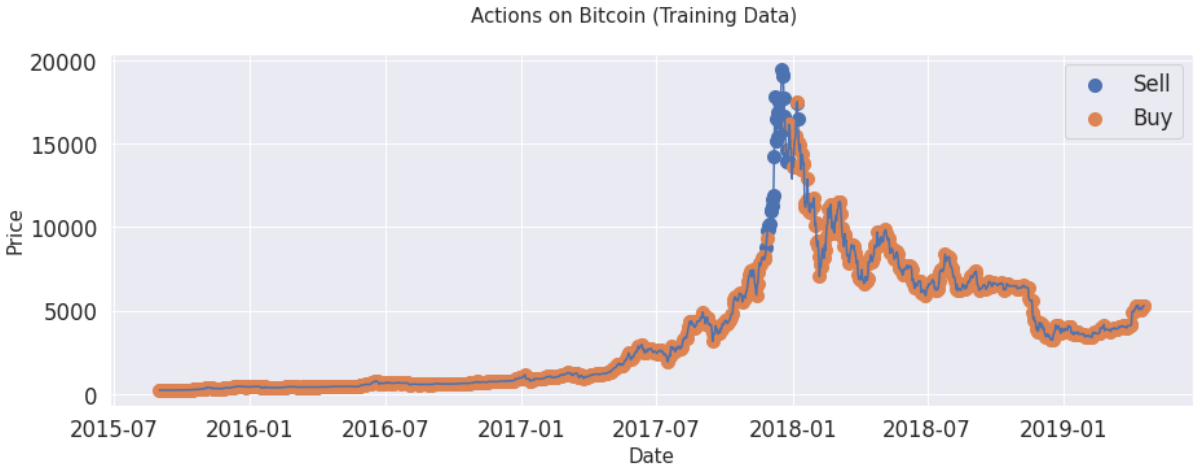

Figure 12: Actions on Bitcoin (Testing Data) in the 100th epoch

6.4.2 Testing Result

In this testing part, we also tested the testing data with 100 epochs to test whether the model trained previously could work on the new pricing environment. After the testing, the rewards are still excellent. With $100,000 investment in the beginning, the final profolio value is with an average of $405551.16 and a maximum of $475200.33, which the average reward is 4.06 times of the orginial values. Looking on the distribution of testing data rewards, most of the epochs could reach the rewards above $400,000. This proves that the model trained before is workable in the new data with a relatively stable performance.

Average Rewards	Min Rewards	Max Rewards	Average APY
405551.16	236409.73	475200.33	192.29%

It is needed to emphasize that although the average APY in testing data is not as high as in training data, the BTC price movement in testing data is not as extreme as in the training data which limit the reward amount in the testing data.

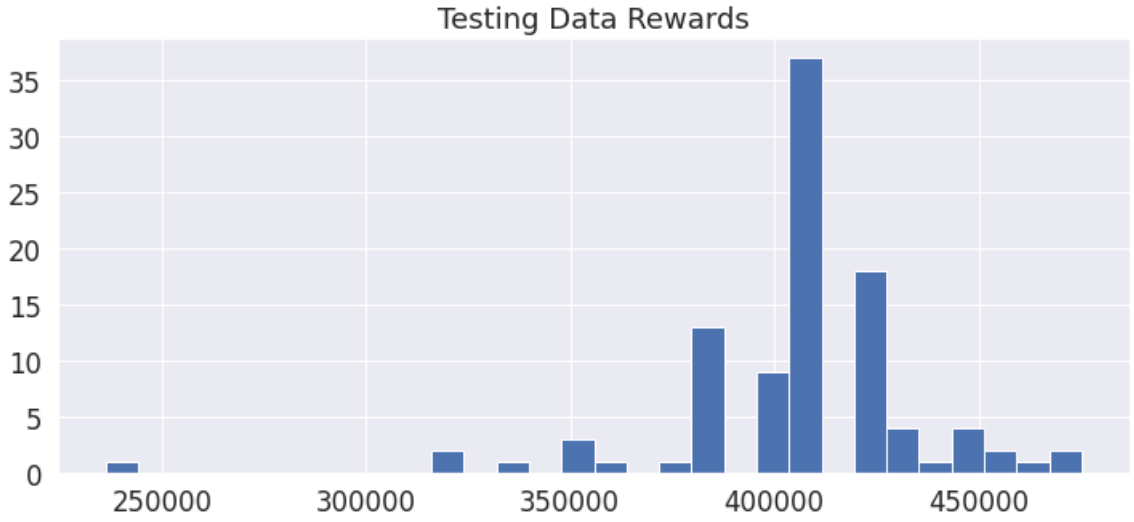

Figure 13: Distibution of Testing Data Rewards within 100 epochs

Looking on the actions taken in the testing data, the actions is more optimal than in the training data. In the following figure, the agent is able to buy at almost the lowest price and sell at almost the highest price in every short interval in order to maximize the profits. This proves the model trained is both workable in a complete new pricing environment. This encourages the futher development of trading bot in real life.

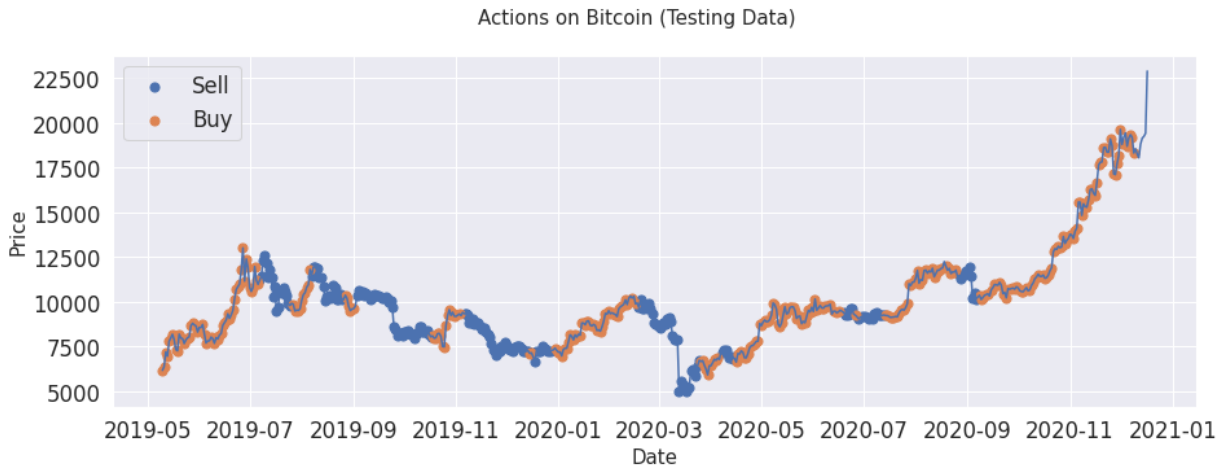

Figure 14: Actions on Bitcoin (Testing Data)

6.4.3 Outperforming in Coronavirus Period

If we try to focus the actions taken during December 2019 to June 2020, there is a drastic decrease during March 2020, which is also known as Coronavirus Crash. At that period, the BTC price follows the stock market movement which resulted in a 39% decrease in Black Thursay (12 March 2020). It reached the lowest price level within 7 years. This made lots of traders being fear and strongly sold the BTC. However, the price keep going up until the present (Decemeber 2020) which caused them regreted selling BTC too quick.

At the same time, the reinforcement learning model has outperformed at that time. In the beginning of the Coronavirus outbreak (mid Feb 2020), the model has already sold all the BTC at that time which is the highest reached before the crash. After that, it keeps indicating "sell" actions until late March 2020. When the model bought BTC again at late March 2020, the price trend started to increase and reached to the history record of $23,000 USD in 17 December 2020.

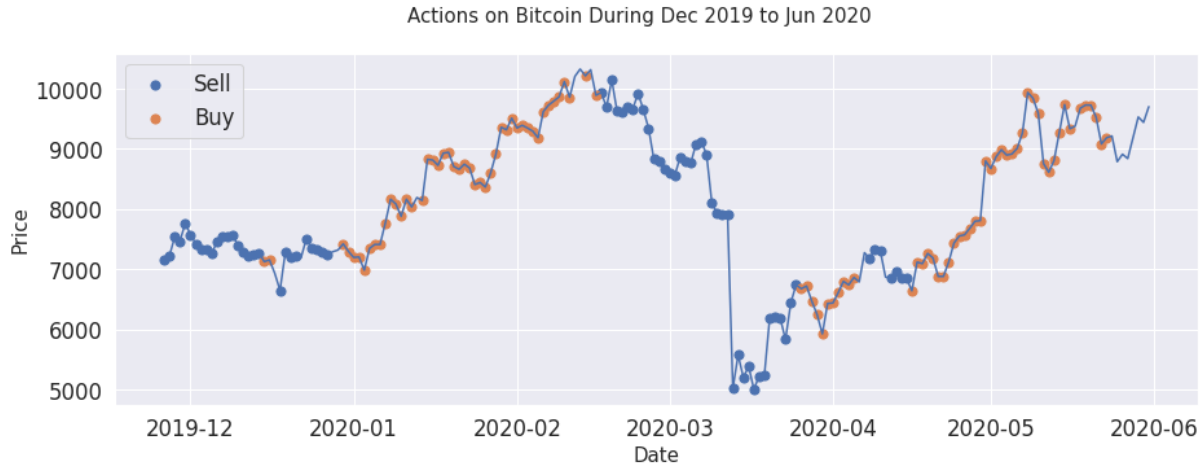

Figure 15: Actions on Bitcoin During Dec 2019 to Jun 2020

Therefore, it can be shown that the model has performed well even facing in the sudden fall movement. The results is encouraging which is out of the expectation. It also shows the importance of including technical indicators in the training to understand the future movement of the price.

7 Future Extension

In the current progress, we achieved to make excellent net profits on cryptocurrency trading, with the condition of trading multiple coins at the same time. However, efforts are still needed to adopt this approach into the real life.

7.1 Strategy Selections

It is observable that the maximizing the reward is the only aim of the agent performed. However, sometimes there might be other considerations and human preferences that could change the policy decision. For example, when it enters bull markets or bear markets, agents can decide when is the right time to all in or sell most to adjust the risk level. Aggressive or preservative modes could be added in the future to tailor-made different people's investment preferences.

7.2 Exchange API Integration

The current experiment result is tested by the history data in a simulation environment. To be more applicable in real life further, exchange API could be integrated with the actions defined in the reinforcement learning policies. On the other hand, commission fee is also one of the potential issues as it might limit the trading frequencies in short-term/day trading strategy and affect the rewards of the agent.

7.3 Sentiment Analysis

In this project, technical indicators are the only consideration of determining the action. As sometimes sentiment in the social media will affect the price trend. Therefore, it is possible to implement sentiment analysis by searching posts or contents on Twitter, forum, and so on. After that, natural language processing is used to identify the level of implication on what trading actions should be taken.

8 Conclusion

The implementation of using reinforcement learning in this project is an exciting start which successfully earn huge profits and obey "buy low, sell high" principle in both training data and testing data. It even prevents trendmenous loss when Bitcoin fell drastically during March 2020. It can be conclued that the project is a huge success in the initial stage.

This project lays out the foundation of applying reinforcement learning on automated

cryptocurrency trading. Once price data with a more shorter time interval and the integration with exchnage API is available, it is possible to apply on the real market world and reduce the time spent on investment and generate passive income with ease.