Problem Chosen	2022 MCM/ICM Summary Sheet	Team Control Number
C		2207494

Transcation Assistant: A Decision Model based on Improved Deep Q-learning Network

Summary

Bitcoin has become a rising star in the investment market, setting off a wave of virtual currency investment boom. At the same time, in recent years, the traditional investment product gold has maintained a relatively stable appreciation trend, whose attraction to investors has not decreased. Therefore, in such a diversified investment environment, investors need a more sophisticated investment strategy to grab greater interests.

In this paper, we first try to use **Deep Q-learning Network**(DQN) which is a fusion product of reinforcement learning and deep learning. The DQN model takes state observations as input and then makes a choice based on the state. After making a choice, the model need to get an incentive value, which is also called reward, so that it can make a better choice in the next similar situation through continuous learning. Considering the limitations of applying the model directly to market transactions, various modifications have been made to most parts of the DQN to enable it to perform better in market transactions.

For state observations, we first conduct random sampling in the data of price trend, and each time extract a continuous series to train a **Gated Recurrent Neural Network**(GRU). After training, the network is able to predict price trend over the next few days based on previous price trend. Then the real trend in the previous week and the trend predicted by GRU in the next three days are used as the observations of DQN state. As for the incentive value of DQN after making a certain decision, we apply **Grid Trading Algorithm** whose core idea is buy low and sell high. Meanwhile, in the process of training DQN, we adopted the training algorithm of ε**-greedy**, so that the model could keep a certain degree of randomness without over-fitting, so as to be more flexible in dealing with various situations. For the optimization of relevant parameters, we use **Adam** optimization algorithm to make it more fast and smoothly close to the optimum point.

After training, the model can make superior decisions according to different price trend. Based on the real market price data provided by the question, the maximum profit our model can make is **$1.4** million with the initial $1,000.

In the sensitivity analysis of the model to transaction costs, we fix other parameters and try to change the transaction rate alpha within a certain range, and then find that the change of α has a great impact on the model. As α increases, the maximum profit that can be made tends to decrease, which is in line with expectation. In addition, we also analyze the stability of the model to price fluctuation and initial money.

Finally, we did a self-evaluation of the model and wrote a summary of our work in a memo.

Keywords: Bitcoin and Gold Transcation; Deep Q-learning Network; Gated Recurrent Unit; Grid Trading Algorithm; Reinforcement Learning; Adam optimization;

Contents

1 Introduction

1.1 Problem Background

As bitcoin has been growing rapidly in financial markets, an increasing number of businessmen have begun investing in bitcoin, which fluctuates sharply in price. Meanwhile, gold, a traditional investment with a relatively stable price, remains a popular choice for investors. Traders tend to determine their current behavior based on the past price of the investment. How to combine the pasttrends of gold and bitcoin to obtain the maximum benefits, especially when traders invest in a variety of assets, has become an important issue.

1.2 Restatement of the Problem

Given two data sets for gold and bitcoin, we are asked to provide the trader with a model of how to operate his assets and money to maximize his portfolio. The data sets contain changes in the two assets from 2016 to 2021. We should only use these data to solve the following problem:

- Design an investment strategy that determines the merchant's investment behavior based on past price movements, prove the optimality of the strategy and find out how much the merchant's $1000 will worth in five years according to this strategy.
- Explore the influence of transaction loss on the model.
- Write a memo to the merchant demonstrating our investment strategy, model, and results.

1.3 Our Work

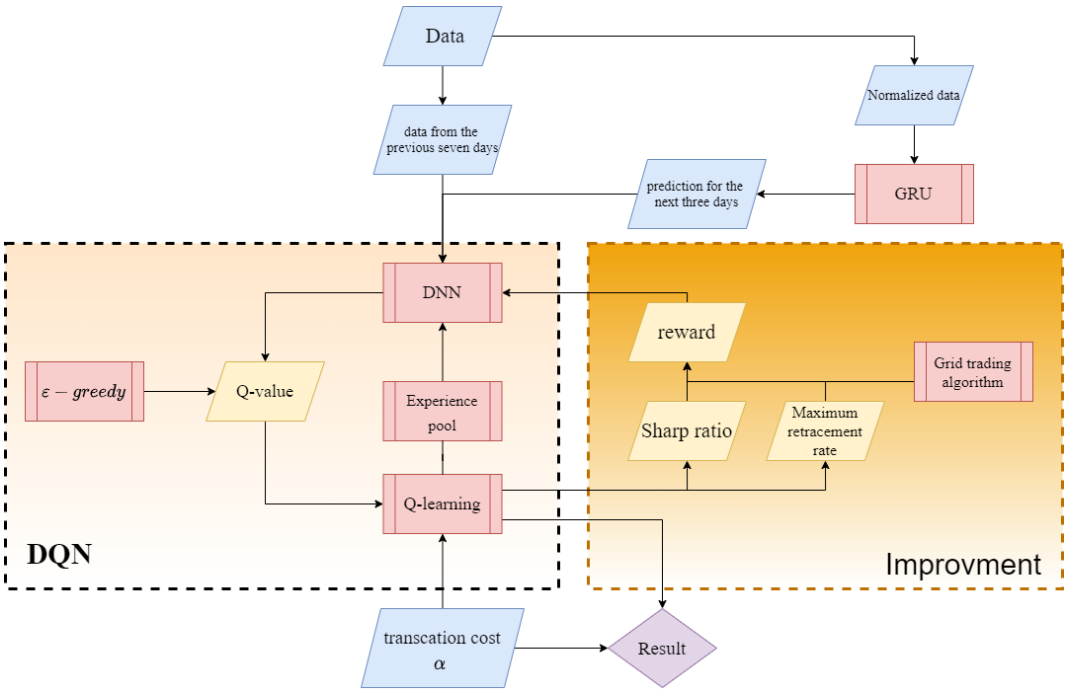


Figure 1: Our work

After analyzing the given data, we set out to explore the maximum profit. Initially, Deep Q-Learning Network (DQN) is used to learn the experienced data (especially the data in the last 7 days) so as to earn more profits in subsequent transactions, especially in the last seven days, which is provided to Gated Recurrent Unit(GRU) to analysis the trend.

Later, we found that the reward mechanism in the DQN can also be improved by the principles of economics, so that the model can be quantified, timed and traded according to the price fluctuations, which has achieved good results.

2 Assumptions and Justifications

In order to simplify the model, we make the following assumptions and justify their use:

- **The commission for each transaction (purchase or sale) costs α% of the amount traded. Assume α_{gold} = 1% and $\alpha_{bitcoin}$ = 2%. There is no cost to hold an asset.**

  Although in actual transactions, the charge ratio of commission fee will change with the change of transaction quantity and price, but in general, the change is not big, so in the small transaction in this paper, it is reasonable to regard the charge ratio as constant.

- **Our transaction will not affect market prices.**

  In this problem, we have only $1000 is as initial capital. Even if our money reaches $100,000 in the end, it won't stir a ripple in the huge financial market. The impact of our asset operation on the whole market is almost negligible. Therefore, it is reasonable to assume that the price is not affected by our transcations.

- **When trading is allowed, gold and bitcoin's trading order is arbitrary. And in order to have more capital when he need to purchase, we assume that sells are executed first.**

  We believe that the trader had plenty of time each day to trade and could chose his own trading order, while trading markets allow traders to trade in any order.

3 Notations

Table 1: Nomenclature

Symbol	Description
s_t	The state at time t
a_t	The action at time t
r_t	The reward at time t
Q	Q-value in Q-learning
$price_t$	price at time t
θ	the parameter of net in DQN
θ^-	the parameter of target-net in DQN
σ	sigmoid function
tanh	tanh function

4 Deep Q-learning Network

During a trader's investment, he gets a new price data item every day, namely our model will get a new training data item every day. Hence we are excepted to build a model that can constantly improve itself as the environment changes. And considering the complexity of price trend, our model rely on a neural network to deal with the infinite states of price trend.

4.1 Reinforcement Learning

Reinforcement learning(RL) is mostly used in scenarios that require interaction with the environment, that is to say, the algorithm can select a corresponding action according to a certain policy when given the state of a certain environment. And the environment will change after the execution of this action. Then, the current state $\mathrm{s_t}$ will convert to a new state $\mathrm{s_{t+1}}$ and the agent will get a reward, according to which the agent will adjust its policy. Thus, after all the steps are executed, in other words, when the process reaches the terminal state, the sum of the reward obtained is expected to be largest.The flow chart of RL is as follows:

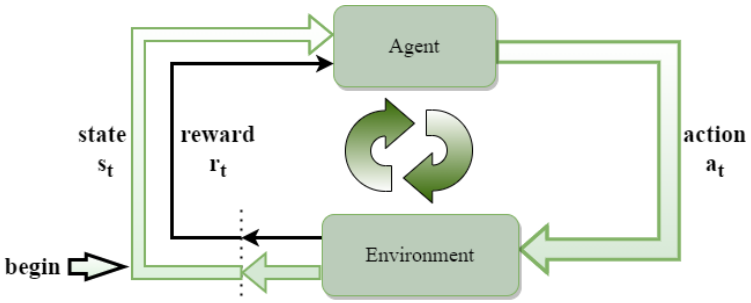


Figure 2: Reinforcement learning flow chart

In order to apply reinforcement learning, we assume that the state is the magnitude of the price uctuation over the past 7 days.

$$state = \left\{ \frac{price_{t-i} - price_{t-i-1}}{price_{t-i-1}} | i = 0, 1, 2, \cdots, 6 \right\}. \tag{1}$$

And the action is to buy, sell or hold. Everyday we give a reward to yesterday's action based on yesterday's and today's prices.

$$reward = \begin{cases} r, & a = buy \\ -r, & a = sell \\ 0.1r, & a = hold \end{cases} \quad (\mathrm{r} = \frac{price_t - price_{t-1}}{price_{t-1}}, \text{a is yesterday}'\text{s action}) \tag{2}$$

As for how to choose the better behavior, we use the Q-learning algorithm to train the agent, which will be elaborated below.

4.2 Q-learning

Q-learning is a way of choosing a strategy, whose core idea is: $Q(s, a)$ equals the sum of future rewards in state s after taking action a. If you know which action will maximize the sum of future rewards, then it makes sense to choose that action.So the point is how to get the value of Q function. One way is to list all the possible $Q(s_t, a_t)$ and then pick the largest one. But when the combination of state and action is infinite, taking this problem for example, we cannot choose the optimal action in this way. So we use deep neural networks which is elaborated as follows.

4.3 Deep Neural Networks

Deep Q-learning Network(its abbreviation is DQN) is a combination of deep learning and Q-learning, whose owchart is as follows:

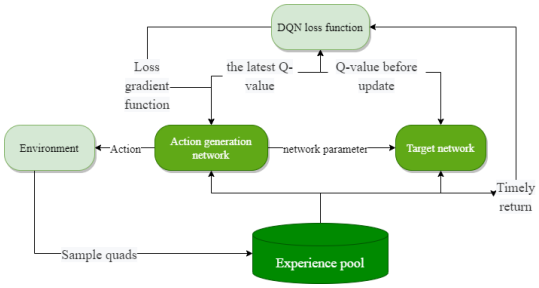


Figure 3: DQN flow chart

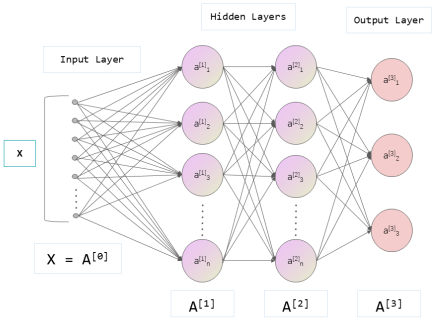


Figure 4: Structer figure of our DNN

In Fig.3, we use a DNN(deep neural network) as the agent, and take the current state as its input. The outputs of the DNN are predictions of Q-values after making dierent actions in current state. The structer of DNN is shown in Fig.4.

In order to choose the overall optimal strategy in each state we have to update Q-values, using the following formula.

$$Q^{new}(s_t, a_t) \leftarrow (1 - \beta) \cdot Q(s_t, a_t) + \alpha \cdot (r + \gamma \cdot max_a Q(s_{t+1}, a_t)) \tag{3}$$

$\boldsymbol{\beta}$ stands for the magnitude of the new Q-value's impact on the past Q-value.**r** is the reward received after taking the action in state s. $\boldsymbol{\gamma}$ is used to reduce the impact of the new Q-value. Besides, the range of both β and γ is between 0 and 1. What's more, we can choose the behavior with the largest Q-value to execute which is mentioned before. But in real problem, this is problematic.For example, assuming all the initial Q-values are 0, then after the states adopt a random action (such as a_1), and get the reward for the rst time; if the reward is positive, then when the state s are encountered later, the program will directly adopt the action a_1.

However, there are many kinds of actions that have never been tried, and perhaps taking other actions will get a larger Q-value. In order to solve this problem, we need to set a threshold ε to maintain a certain degree of randomness, that is, before making a decision, model generates

a random number, if this number is smaller than ε, then the model randomly selects an action, otherwise it selects an action that can make the Q-value largest under the currently known conditions. This threshold ε is often set to a large value at the beginning, and it will slowly decay to the minimum value we set as the program continues to iterate. It has the advantage that allowing the program to iterate through all the S-A pairs to determine exactly which action is optimal to choose in a given state, which is called exploitation, and this algorithm is called ε-greedy. The nal question is how we can train DNN to make better predictions, which we'll list below.

4.4 Train the DNN

We have mentioned that it is up to DNN to decide which action we choose, so we need to train DNN to perform satisfying. And this is obviously a supervised learning issue. In deep learning, in order to train a supervised learning model, we have to determine training set, labels and loss function. Firstly, we can see that the output of DNN is the Q-value. However, there is no way to know the optimal Q-value , so the only thing we can use is reward. For a spspecific state or action, the reward obtained is certain and constant.

Therefore, we consider starting from reward, and transform the comparison between the predicted Q-value and the real Q-value into the issue of making the model t the reward in essence. Now, we give the loss function. The problem has now shifted to requiring a training set that provides a set of quads (s, a, r, s') where s' is the next state after s executes a. This is the **experience replay** we are talking about. Since after each action our environment moves to the next state and the agent earns a reward, we can either get one of these quads after each action, or put the quad directly into the experience pool.

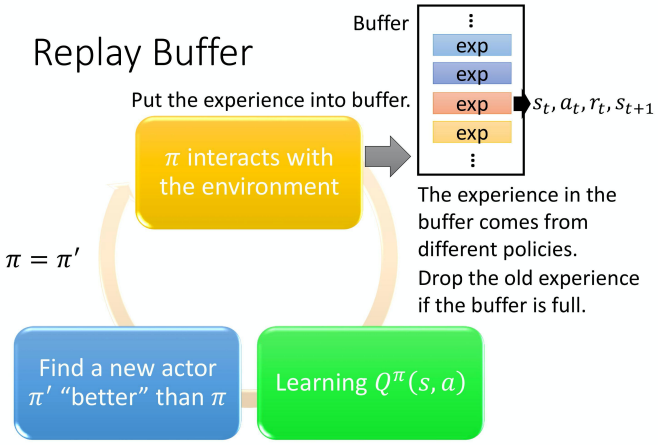


Figure 5: Structer figure of our DNN

Then in training, we can randomly select a small number of quads from the experience pool as a batch. The training pseudocode is shown below.

Algorithm 1 Q-learing algorithm

Initialize reply memory D to capacity N
Initialize action-value function Q with random weights
for episode = 1, M **do**
  Initialize sequence $s_1 = x_1$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
  for t = 1, T **do**
    With probability ε select a random action a_t
    otherwise select $a_t = max_a Q * (\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
    Set

$$y_j = \begin{cases} r_j, & \text{for terminal } \phi_{j+1} \\ r_j + \gamma max_{a'} Q(\phi_{j+1}, a'; \theta), & \text{non-terminal } \phi_{j+1} \end{cases}$$

    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$
  end for
end for

What's more, in order to prevent over-fitting, we design another DNN which has the same structure as the previous DNN, but with different parameters. We call that network target network, and its parameters are denoted by θ_{i-} . as is shown in Fig.6. In each iteration, we update $_i$ instead of θ_{i-} , and specify that θ_{i-} = θ_i after a certain number of steps.

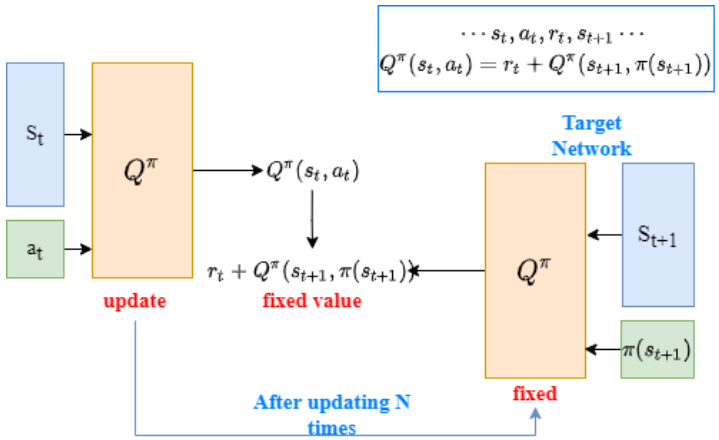


Figure 6: structer figure of target network

Now our loss function is:

$$Loss = (r + \gamma max_{a'} Q(s', a'; \theta') - Q(s, a; \theta))^2 \tag{4}$$

The pseudocode of the training is as follows:

Algorithm 2 deep Q-learning with experience replay.

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with rangom weights θ
Initialize target action-value function $\widehat{Q}$ with weights $\theta^- = \theta$
for episode = 1, M **do**
  Initialize sequence $s_1 = x_1$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  for t = 1, T **do**
    With probability ε select a random action a_t
    otherwise select $a_t = argmax_a Q(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transition $(\phi_j, a_j, r_j, \phi_{t+1})$ from $\mathcal{D}$
    Set

$$y_j = \begin{cases} r_j, & \text{if episode terminates at step j+1} \\ r_j + \gamma max_{a'} \widehat{Q}(\phi_{j+1}, a'; \theta^-), & \text{otherwise} \end{cases}$$

    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to the network parameters θ
    Every C steps reset $\widehat{Q} = Q$
  end for
end for

5 How to set reward

In the text above,we have used formula (2) to calculate the reward. But in this way, rewards cannot be quantified,which means they cannot be influenced by how much money is made or lost. Therefore, the method of Sharpe Ratio and maximum retracement rate in economic principles are used to quantify the reward.

In this section, we will first introduce the grid trading algorithm to explain how we perform quantitative operations, and then introduce the two ratios, which will be used in the updated methods to make the reward more reasonable.

5.1 Grid trading algorithm

Grid transactions are derived from foreign exchange transactions. Because foreign exchange prices are usually in a range of fluctuations. Traders often buy low and sell high to boost returns. But there needs to be a standard for what price to buy and sell. In order to prevent traders' subjective factors from influencing the transaction, the bid price and offer price are calculated in advance with a quantitative model, forming a grid table, which is operated strictly in accordance with the grid table, and finally evolved into a grid trading method.

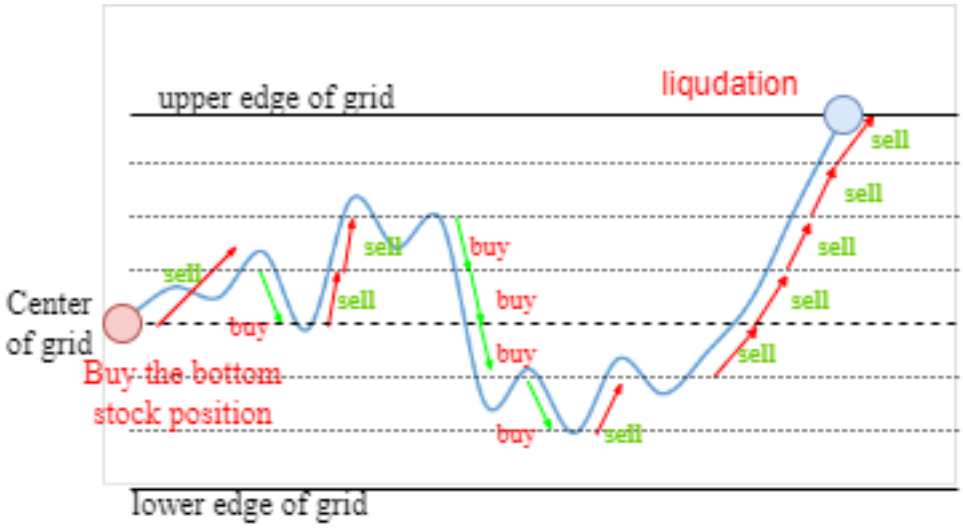


Figure 7: Grid trading diagram

Therefore, when the stock market is faced with price fluctuation, we can set up the price fluctuation center, determine the upper and lower orbits of the grid, and divide the upper and lower orbits into several grids. Buy bottom positions at the center of price movements, sell your positions one by one as prices rise, and buy positions one by one as prices fall.

In order to better reflect the profit advantages of grid algorithm, an example is given below, where the data is shown in Table 2:

Table 2: Grid trading algorithms simulate trading records

Operation	Present price	Shares held	Stock market price	Cash in the account	Total assets
Buy	1.7	1000	1700	8300	10000
Buy	1.5	3000	4500	5300	9800
Buy	1.3	5000	6500	2700	9200
Hold	1.1	7000	7700	500	8200
Sell	1.3	5000	6500	3100	9600
Sell	1.5	3000	4500	6100	10600
Sell	1.7	1000	1700	9500	11200

Table 2 is the simulated trading record of a certain stock in a certain period according to the grid trading algorithm. Where, the leftmost operation column represents the corresponding operation on the stock at the current price according to the grid trading algorithm; When the stock price uctuates between $1.1 and $1.7, the number of grids is 3 and the interval between grids is $0.2, that is, the length of the price interval of each grid is $0.2, and the number of stocks bought or sold each time is 2000 shares.

When the stock price drops from $1.7 to $1.1, add positions and buy stocks in batches, each time buying 2000 shares at the trading price within the grid range; When the stock price starts to

recover, gradually sell the position, selling 2000 shares at each trading price in the grid until you return to the initial position. Finally, the prot of each grid is $400, and the total prot of the grid trading algorithm is $1200. Although the above simulated transaction records omit the calculation of transaction fees, the prot margin is still considerable.

5.2 Sharpe Ratio and Maximum Retracement Rate

In the design of immediate reward function, the deep reinforcement learning model designed in this paper adds the inuence of risk and return factors, and uses Sharpe ratio and maximum retracement rate. The Calculation formula of Sharpe ratio is:

$$sharp\_ratio = \frac{E(R_t) - R_f}{R_f} \tag{5}$$

Rf stands for risk-free rate of return, and the short-term possible rate of return of gold or Bitcoin is used in this paper. $E(R_t)$ represents the expected return of the investment strategy, and $S(R_t)$ represents the standard deviation of return of the strategy. As can be seen from the above formula, Sharpe ratio takes returns and risks into account comprehensively, indicating the excess returns that can be obtained by an investment strategy if one unit of risk is assumed, which is a common risk-return factor in the eld of quantitative trading. The calculation formula of the maximum withdrawal rate is as follows:

$$maxRecall = max\frac{netWorth_i - netWorth_j}{netWorth_i} \tag{6}$$

Where, $\mathrm{netWorth_i}$ represents the total assets of the account at time i, and j represents some time after time i. According to Formula (5), the maximum withdrawal rate can be used to represent the anti-risk ability of the investment strategy, and its practical signicance can be understood as the maximum loss that the investment strategy may bring.

When the growth rate of the total assets of the account is increasing or the decline rate is decreasing, and when the total assets are increasing, reward is positive. Reward is a negative value when speed decreases or decreasing speed increases. For liquidation, reward refers to the dierence between the total market value of the asset after liquidation and the initial capital of the account, namely, the nal prot of the account. Finally, the calculated reward value is summed up with the rst-order dierence of the maximum withdrawal rate of total assets and sharpe ratio of the account at the current time as the nal reward value. At the same time, in order to reduce the phenomenon of "breaking the net" due to the bursting of positions in grid trading and the phenomenon of short positions due to the premature stopping of prots, a large penalty value is also set for these two situations in the calculation of reward.

6 Gated Recurrent Unit Network

When DQN decides its action, if it can predict the price trend in the next few days and take it as a part of the state, it will be of great benefit to DQN's behavior decision. Therefore, we establish the GRU model to predict the future price trend. In the previous work, we simply take the daily price fluctuation in the last week as the state observation in Q-learning, which is obviously not

accurate. in order to make a better choice, we should also consider subsequent price fluctuation. However,in reality, we can not accurately know the future trend, so we need to use the popular GRU neural network to predict. In this section, we will begin with a brief introduction to the fundamental principles of GRU and the reasons for choosing GRU. After that, we show how to train a GRU neural network and apply it to predict the price trend. Finally, we will show how it can be used for state observation in Q-learning.

6.1 Gated Recurrent Unit

In the training process of Recurrent Neutral Network (RNN), when the size of sequence is really large or small, the gradient of RNN is more likely to decay or explode. Although clipping gradient can deal with gradient explosion, it cannot solve the problem of gradient attenuation. For this reason, it is difficult for recurrent neural networks to capture the dependency relation with large time step distance in time series in practice. To overcome this problem, long short-term memory (LSTM) network is obtained by improving RNN. LSTM adds input gates, forgetting gates and output gates to the hidden layer of RNN, as well as memory cells with the same shape as the hidden state, so that RNN has long-term memory function, so as to record additional information. LSTM network structure is shown in Fig.8.

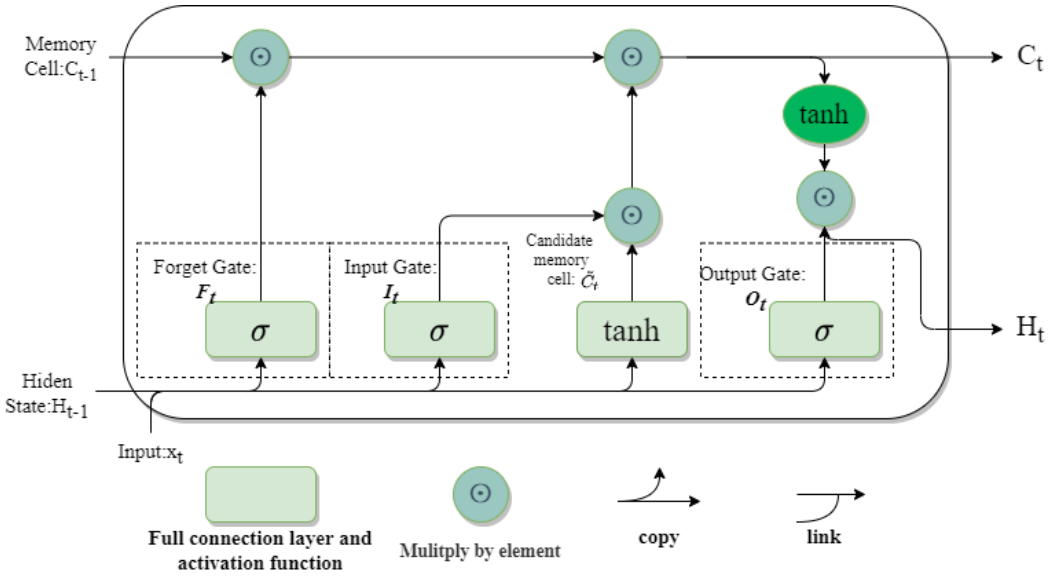


Figure 8: Long and short-term memory neural network structure diagram

In Fig.8, the three gates in dotted box from left to right are forgetting gate, input gate and output gate. The forgetting gate determines what information will be forgotten and what will be retained; the input gate inputs the retained information; the output gate outputs information. C_{t-1} represents the information of memory cells at the last moment; H_{t-1} represents the hidden state at the last moment; X_t represents the input at present; F_t stands for the output of forgetting gate; I_t stands for the output of input gate; O_t represents the output of output gate. $\tilde{C}_t$ represents the current candidate memory cell information. C_t represents the current memory cell information, H_t represents the current hidden state. The relevant equations are shown below

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i)$$
$$F_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f))$$
$$O_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o)$$
$$\tilde{C}_t = \tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c)$$
$$C_t = F_t \odot C_{t-1} + I_t \odot \tilde{C}_t$$
$$H_t = O_t \odot \tanh(C_t)$$

We can see that the structure of LSTM is complex and the training will take a lot of time. Therefore, GRU is proposed to simplify LSTM. GRU only contains reset gates and update gates, and its structure is shown in Fig.9:

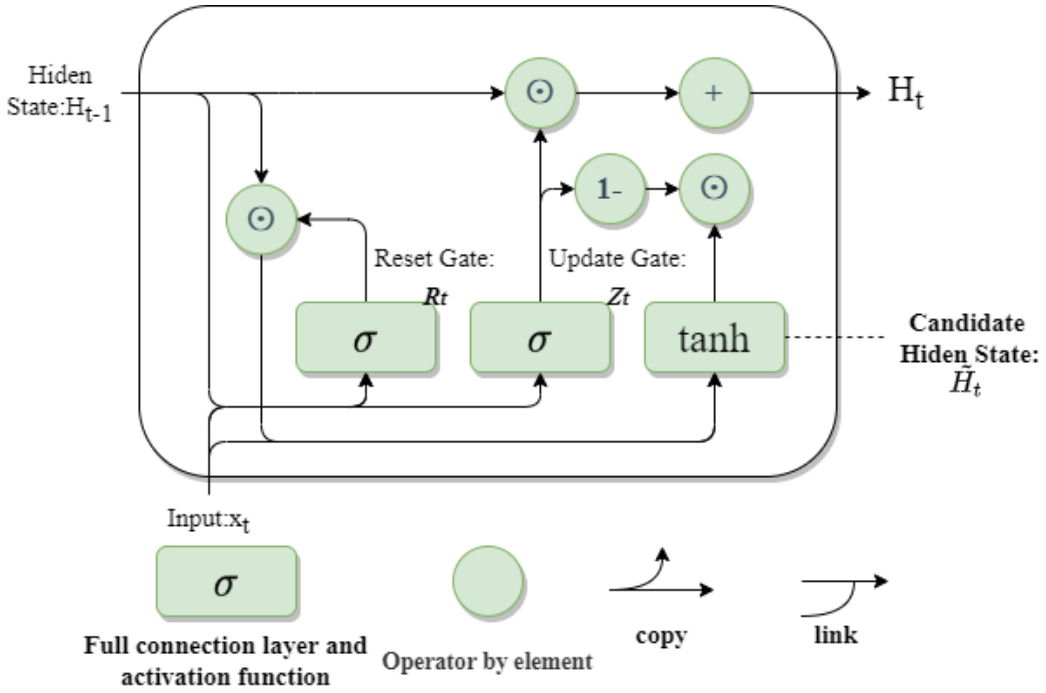


Figure 9: Structure diagram of gated loop unit neural network

In Fig.9, R_t represents the output of the reset gates ; Z_t represents output of the update gates , and $\tilde{H}_t$ represents the candidate hiden state of the current time. The relevant equations are shown below:

$$R_t = \sigma(X_t W_{xr} + H_{t-1} W_{hr} + b_r)$$
$$Z_t = \sigma(X_t W_{xz} + H_{t-1} W_{hz} + b_z)$$
$$\tilde{H}_t = \tanh(X_t W_{xh} + (R_t \odot H_{t-1}) W_{hh} + b_h)$$
$$H_t = Z_t \odot H_{t-1} + (1 - Z_t) \odot \tilde{H}_t$$

As you can see from the formula above, the reset gate controls how the hidden state of the previous time squence flows into the candidate hidden state of the current time squence. The hidden state of the previous time step may contain all the historical information of the time series up to the previous time squence. Thus, a reset gate can be used to discard historical information irrelevant to the prediction. And the update gate controls how the hidden state should be updated by the candidate hidden state that contains the current time squence information. In general, the reset gate helps capture short-term dependencies in time series while the update gates help capture long-term dependencies in time series.

GRU has a simpler internal structure, fewer parameters, faster convergence rate and higher efficiency. Therefore, GRU is selected in this task.

6.2 Apply GRU to state observation

As we mentioned earlier, in DQN, we simply use daily price fluctuations in the previous week as state observation in Q-learning, which is obviously not accurate. And we should also consider subsequent price movements in order to make a better choice. But in reality, we can not accurately know the future trend, so we use GRU neural network to predict and its structure is shown in Fig.10.

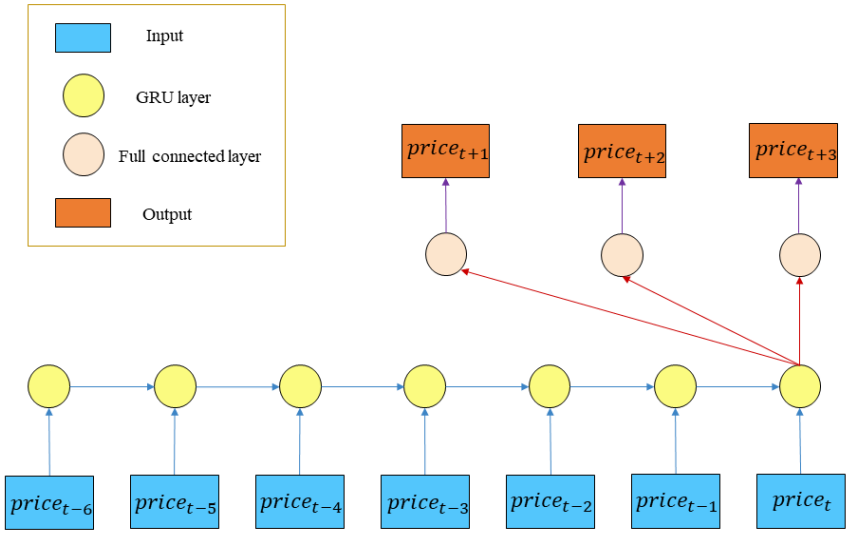


Figure 10: Structure figure of GRU network

First, we build a training set. Each input data in the training set is the price of each day in the previous week, while the target is the price in the next three days. We measure the loss by mean square error, whose formula is as follows:

$$MSE = \frac{\sum_{i=1}^{n} (Prediction_i - Truth_i)^2}{N} \tag{7}$$

And we use Adam algorithm to optimize the parameters of GRU. The pseudocode is as follows:

Algorithm 3 Adam algorithm

Input:γ**(lr),**β_1**,**β_2**(betas),**θ_0**(params),**$f(\theta)$**(objective),**λ**(weight decay),amsgrad**

Initialize $m_0 \leftarrow 0$(first moment),$v_0 \leftarrow 0$ (second monent), $\widehat{v_0}^{max} \leftarrow 0$

for t = 1 to $\cdots$ **do**

$g_t \leftarrow \bigtriangledown_\theta f_t(\theta_{t-1})$

if $\lambda \neq 0$ **then**

$g_t \leftarrow g_t + \lambda\theta_{t-1}$

end if

$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1)g_t$

$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2){g_t}^2$

$\widehat{m_t} \leftarrow m_t/(1-{\beta_1}^t)$

$\widehat{v_t} \leftarrow v_t/(1-{\beta_2}^t)$

if amsgrad **then**

$\widehat{v_t}^{max} \leftarrow max(\widehat{v_t}^{max}, \widehat{v_t})$

$\theta_t \leftarrow \theta_{t-1} - \gamma\widehat{m_t}/(\sqrt{\widehat{v_t}^{max}} + \epsilon)$

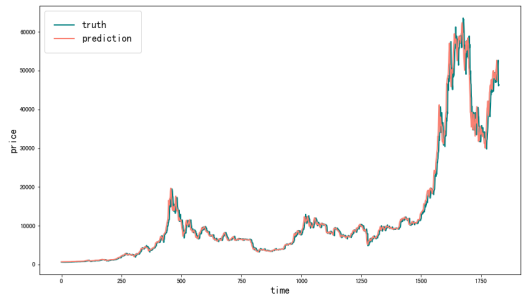
else$\theta_t \leftarrow \theta_{t-1} - \gamma\widehat{m_t}/(\sqrt{\widehat{v_t}} + \epsilon)$

end if

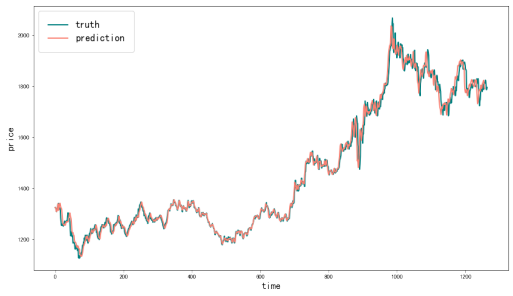
end for

return θ_t

After 2,000 rounds of training, we use the GRU to predict the price trend from 9/11/2016 to 9/10/2021. Each time we use the prices in previous weeks as inputs and predict the prices in next three days until the last day. The results of GRU are shown as follows:



Trend of bitcoin



Trend of gold

Figure 11: Prediction of GRU and true prices

We can see that the GRU performs extremely well on predicting the future trend based on the previous trend .That's to say, although we cannot know the future trend, we can take the prediction of GRU as a reference. Now, we can modify the formula (1):

7 Comprehensive strategy and model

So far, we've introduced how to apply economical principles to get rewards, how to apply GRU to get state, and how to train DNN. And the whole process can be seen in Fig.12. Then, in this section, we will talk about the training process and result of the comprehensive model.

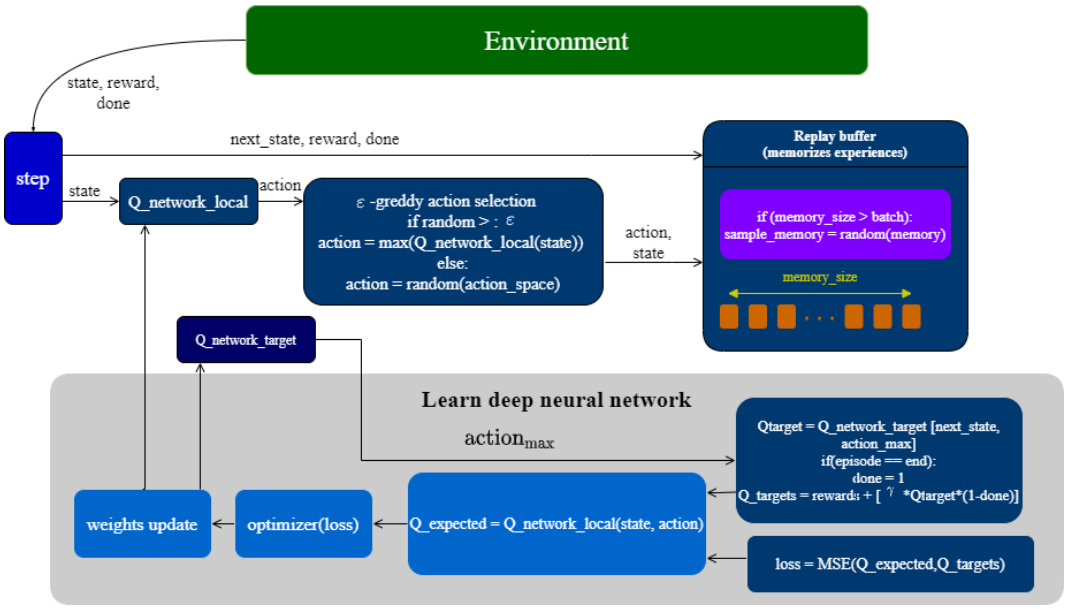


Figure 12: Structure figure of GRU

7.1 Training process

We use the Adam algorithm, which has been mentioned before, to optimize the parameters of DQN. And the training process is shown below:

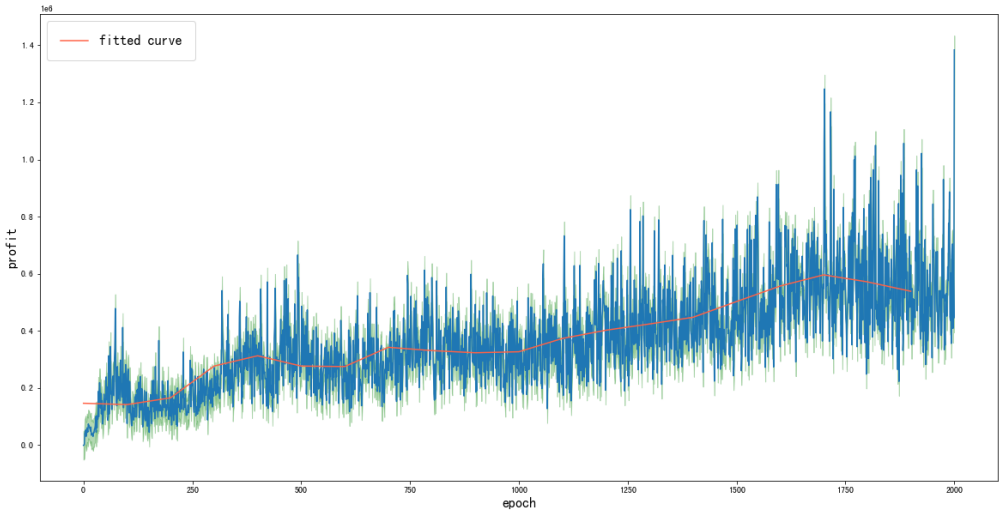


Figure 13: Training curve

From the Fig.13, we can see that the profit rises gradually with increasing epochs of training. And our maximum profit can reach **$1.4 million** with the initial $1000.

Now, we use the trained model to conclude the transaction. The comparison of account growth rates with price growth rates is shown below and we can find that our accounts have multiplied **1,400 times**.

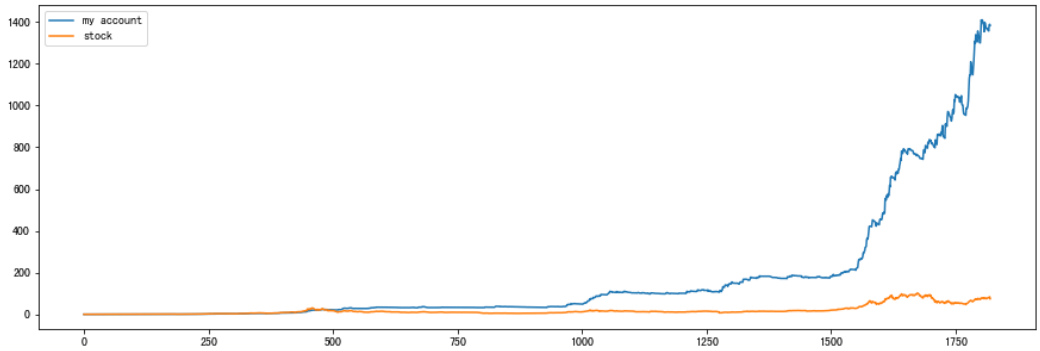


Figure 14: Training curve

7.2 Result

The daily decisions made by the model are shown in Fig.15. where we can see that our model makes more selling actions when the price starts to go down and makes more buying actions when the price starts to rise which is well in line with our expectations.

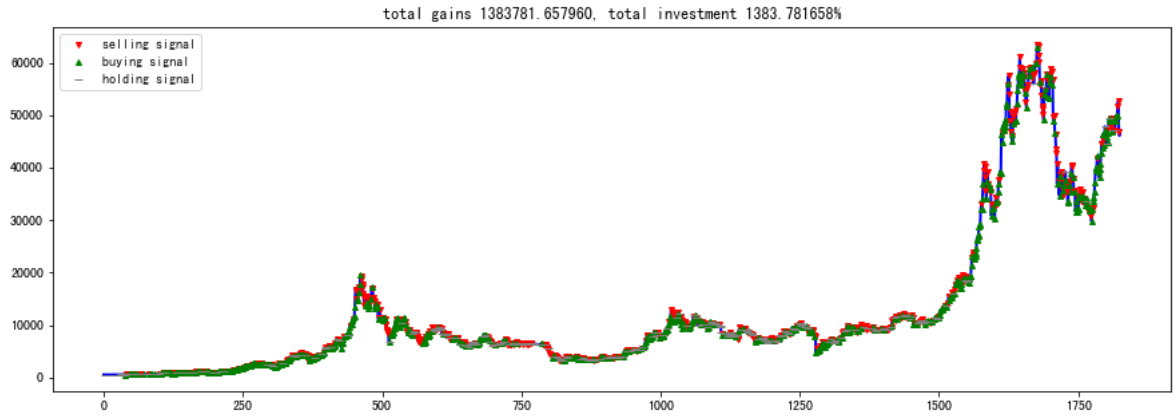
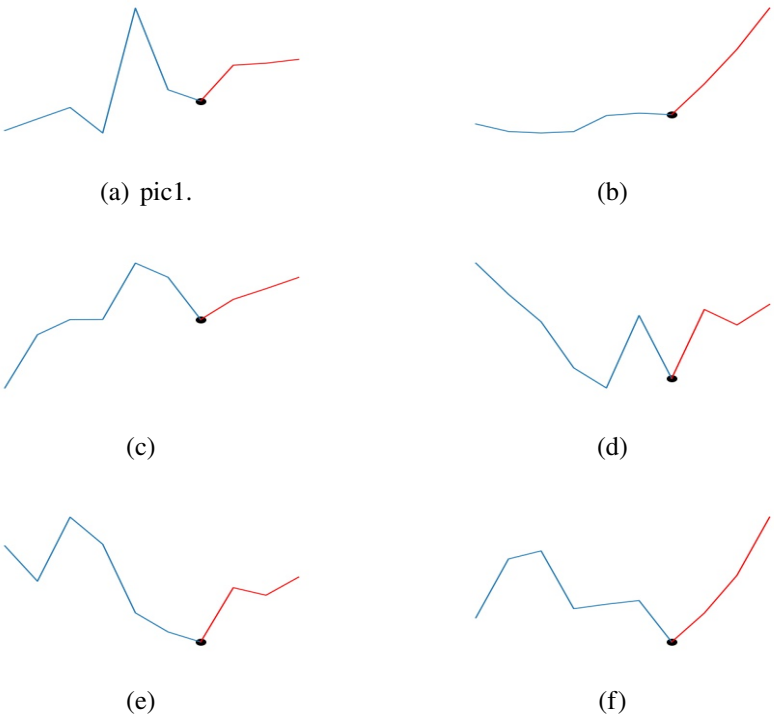


Figure 15: Training curve

Now, we make some trend charts from random sequential segments of the price data, where the blue line is the real price trend, the red line is the trend predicted by GRU , and the black dot indicates the current time.

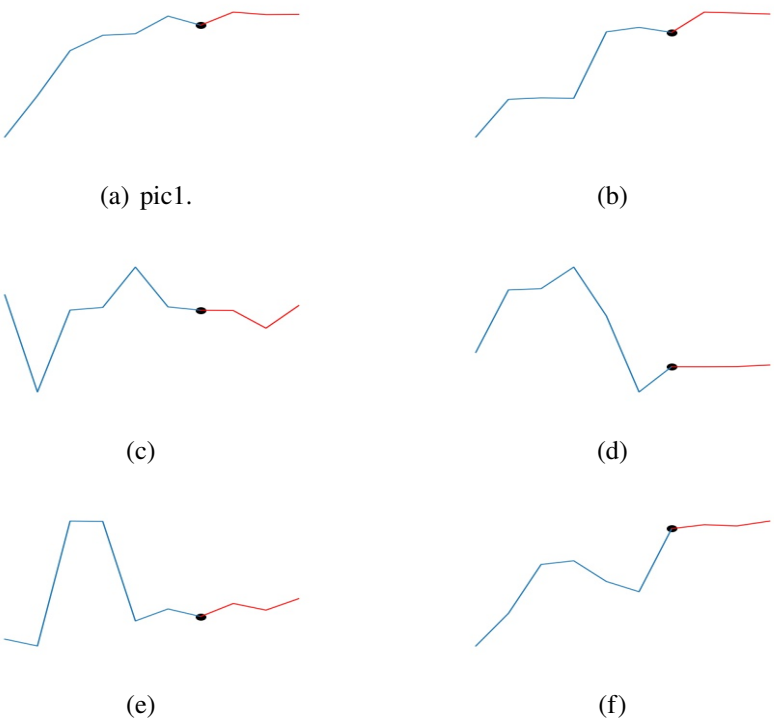


(a) pic1. (b)

(c) (d)

(e) (f)

Figure 16: Price trends suitable for buying

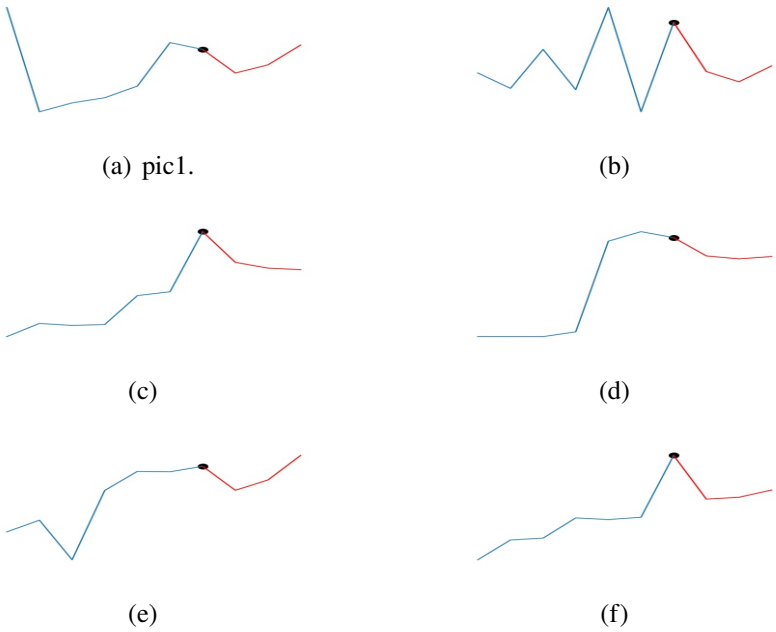


(a) pic1. (b)

(c) (d)

(e) (f)

Figure 17: Price trends suitable for holding



(a) pic1. (b)

(c) (d)

(e) (f)

Figure 18: Price trends suitable for selling

As we can see from the table, When price is predicted to rise, the model makes the decision to buy; when price is predicted to fall, the model makes the decision to sell; when no significant price fluctuation is predicted, the model will make the decision to hold. This is in line with our expectations.

8 Analysis of Sensitivity and Stability

8.1 Sensitivity

In order to test the sensitivity of our model, we select several parameters for analysis and select two important parameters: α and learning rate. $\alpha\%$ is the transaction rate, and the result caused by the change of α can reflect the reaction degree of the model to the fee. The learning rate reflects the learning degree of the model to the experienced data. By changing alpha and learning rate respectively, the following two figures can be obtained:

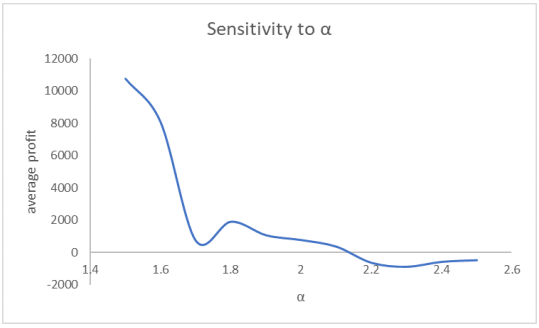


Figure 19: sensitivity to alpha

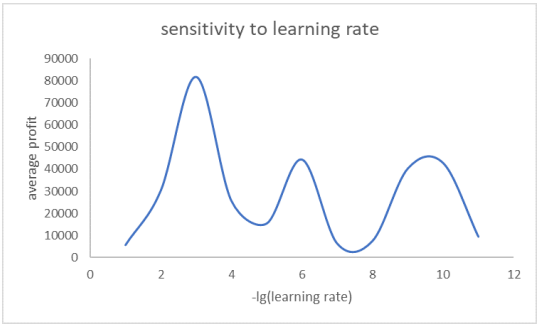


Figure 20: sensitivity to learning rate

As can be seen from Fig.19, the profit fluctuates and declines with the increase of α, showing the model has good sensitivity. In addition, this is a good reflection of the actual transaction, the higher the fee, the less money can be made with the same initial capital; At the same time, when α continues to increase, the profit will turn negative, indicating that multiple transactions will lose money instead of making money when the commission fee is too high. As can be seen from Fig.20, the model is very sensitive to learning rate and its response fluctuates when the learning rate changes.

8.2 Stability

Since prices in financial markets fluctuate over time, we further test the stability of our model. Using the given data, the random fluctuation range is set to change from 1% to 5%.Similarly, our capital may also change, so we also conduct random fluctuation analysis for the initial amount. Then, we get the following two graphs.

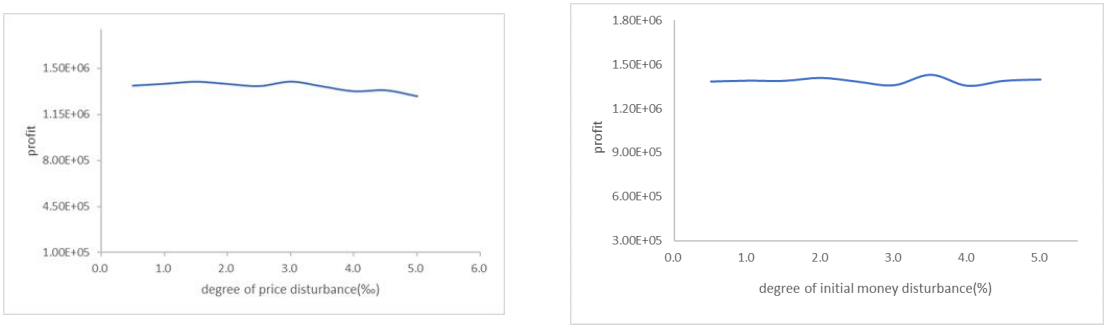


Figure 21: Profit fluctuation

In the two figures above, the vertical axis is the profits of our model under different fluctuation conditions, and the horizontal axis stands for the magnitude of fluctuation. As can be seen from the figure, when the price fluctuates in range of 5‰ in the same day and the principal fluctuates in range of 5%, the profit of the model is good, so it shows that our model has good stability.

9 Model Evaluation and Further Discussion

9.1 Strength

- According to the complex market price trend, the model automaticaly analysis of the transcation strategy, and obtain a satisfying profit.
- By analyzing and learning data, our model can take advantage of price fluctuations, even when the market is depressed.
- Our model provides variable parameters that can be adjusted manually or automatically during training process.
- In the face of a variety of investment products, model can automatically configure the purchase ratio and sale situation according to trends of each asset's price.
- The model incorporates economic principles.

9.2 Weakness and Possible Improvement

- The response to the change of learning rate fluctuates greatly. The profit change under the combined effect of learning rate and alpha is shown below. In the future, we may improve the parameters to make the curve more smooth.

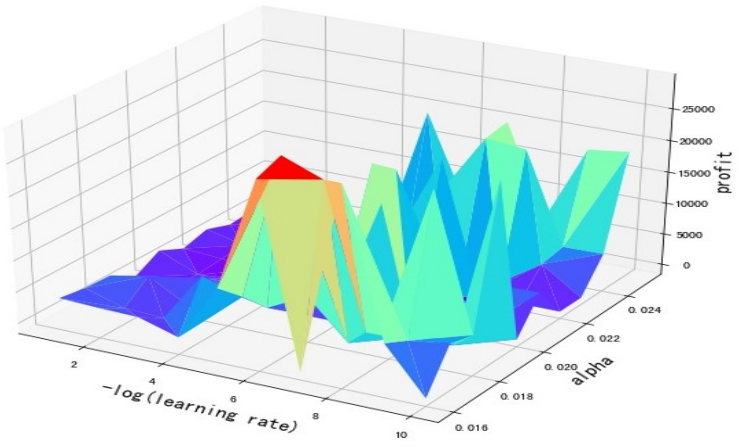


Figure 22: Profit surface plot relative to LR and α

- We can add users interfaces for different prople's preferences, then the trading strategy will not be only planned according to the price and profit.
- Model parameters can be scripted, so that previous learning can be reused in other predictions.

Memorandum

To: The investment trader

From: MCM Team 2207494

Subject: Strategy to Invest in Gold and Bitcoin

Date: February 22, 2022

Hello! Are you still worried about not knowing how to operate against frequent changes in assets price? Now, we propose a solution and get a model for your distress. and it's our pleasure to introduce the content of our model, which includes the problems you may meet in the process of usage. Then, we will divide it into three aspects: strategy, model and results.

Strategy

- By predicting the data, we can make quantitative and timing trading, reduce trading risks and increase our possible profits;
- In consideration of fees, try to avoid frequent trading when the price fluctuation is not obvious;
- Taking gold and Bitcoin into consideration, investment benefits will be gained according to price fluctuations;
- Paying attention to price peaks and valleys can effectively mitigate the impact on profits when the economy is depressed.

Model

- Good consistency. For gold and bitcoin, good profits can be obtained by using the same model;
- Expandable to increase other features, fast training speed, adaptability of the model to different situations, high sensitivity;
- Growability. The data used at present is relatively small, so the current trained model does not reach the optimum.

The model works well for small transactions, but if you want to use it for large financial transactions, you need to adjust the parameters of the model, or you need to provide some samples - our model can adjust internal parameters automatically.

Results

- After 5 years of stable trading, the model can obtain a relatively stable income of more than $300,000 by using $1,000;
- Prots can still be made under more extreme circumstances, but may not be very high.

Because actual gold and bitcoin prices can also change over the course of a day, the model can also be used to predict the purchase of stocks or other assets over a short period of time.

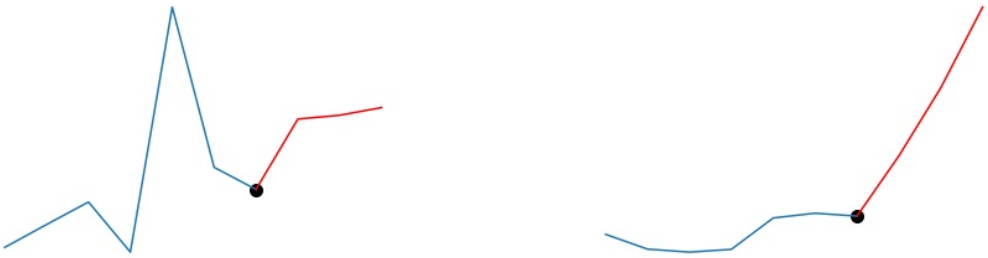


Figure 23: Price trend suitable for buying

Fig.23 shows the conditions suitable for buying assets, and we recommend that you actively buy bitcoin when you see the price action as shown above.

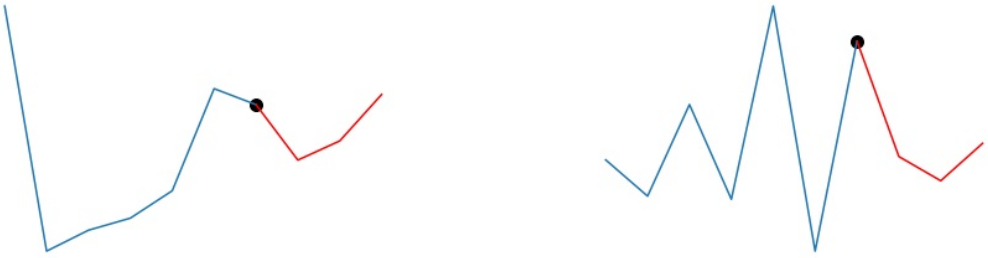


Figure 24: Price trend suitable for selling

Fig.24 shows the conditions suitable for selling assets, and we recommend that you actively sell bitcoin when you see the price action as shown above.

References

[1] Eleanor P. Jones, Chris Conyers, Victoria Tomkies, Nigel Semmence, David Fouracre, Maureen Wakefield, and Kirsty Stainton. Managing incursions of Vespa velutina nigrithorax in the UK: an emerging threat to apiculture. Scientific Reports, 10(19553), 2020.

[2] Lin Weimin. Simulation Effect analysis of Grid Trading Method for Stock Index Products of Fund Companies [D]. Guangdong University of Foreign Studies,2020.

[3] Hochreiter, S., chmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[4] Chung, J., Gulcehre, C., Cho, K., Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg & Demis Hassabis (2015). Human-level control through deep reinforcement learning. Nature, 518, 529-533

[6] Zhang Maojun, Rao Huacheng, Nan Jiangxia, Wang Guodong. Research on Quantitative Trading Timing Strategy Based on Decision Tree [J]. Systems Engineering,,:1-17.

[7] Shen Xuemei, The application of grid trading strategy in financial product investment[J], Research on industrial Innovation,2021,(19):105-107.

[8] Xu Hai-yan, WU Hao, LI Dong, Chen Lei, DENG Si-jing. Fault Line Selection of Distribution Network based on Gated Loop Unit Neural Network [J]. Journal of Electric Power Systems and Automation,:1-10.