Problem Chosen	2022 MCM/ICM Summary Sheet	Team Control Number
C		2205961

DTSL-PPO: A Well-Tailored Deep Reinforcement Learning Model to Highly Profit in Futures Trading Market

Summary

Market traders are keen to buy and sell volatile assets with the goal of maximizing their total returns. How volatile assets can be profitable depends on a trader's strategy of whether the assets in their portfolio should be bought, held or sold. Therefore, the selection of trading strategy for two typical volatile assets (gold and bitcoin) in the investment process is of great importance.

In the first part, we establish **Reinforcement Learning (RL) Model** over five years of transactions. Specially, with the goal of maximizing the final profit in five years, we need to make the best current decision each day based on current and prior information. This dynamic decision-making characteristic inspires us to adopt **Markov Decision Model** to characterize the investor's action sequence, and further establish **RL model** to attain a dynamically-approaching solution to this problem. RL is able to interact with market environments in time sequence during the training process to maximize the set **reward function**, so as to achieve the most suitable strategy for futures trading market.

In the second part, we carry out the strategy optimization of our proposed model. First of all, the optimality of our framework based on RL is rigorously been proved in mathematical theory. Then, considering model unpredictability and action continuity, we employ **Proximal Policy Optimization (PPO)** among a number of state of the art DRL algorithms, which combines **gradient descent idea** with **neural network** and improves upon the basic **Actor-Critic (AC) framework**. It is verified that this selected algorithm has a considerable good effect on combined futures trading, with the reward rate of up to 2286.173% and the Sharpe Ratio of up to 1.02.

In the third part, we perform three progressive optimizations on our model and propose **Dynamic T-Period Sliding Window Deep Reinforcement Learning Algorithm PPO Combining Short-Term and Long-Term Strategies (DTSL-PPO) Model** that is well-designed for combined futures trading. Firstly, **risk aversion coefficient** is introduced in combination with modern asset portfolio theory to enhance the update strategy of PPO algorithm. Then we propose **Dynamic T-Period Sliding Window Model (DT-PPO)**, which solves the problems of slow start-up in the early stage and focuses on short-term trends while ignoring long-term trends in original PPO algorithm. However, DT-PPO Model concentrates too much on long-term trends in the late trading period and responds to short-term trends slowly. Hence, based on economic principle, a **Short-Term and Long-Term combination framework (SL)** is further proposed. Moreover, we polish the combination strategy of long-term and short-term trends, and finally propose **DTSL-PPO Model**. With the well-tailored enhancements above, the reward rate reaches 4065.38% and the Sharpe Ratio reaches 1.14.

In the fourth part, we analyze the sensitivity of model. By testing different transaction cost combinations through **control variable method**, we discover that the strategy of our model has different sensitivity to changes in gold and bitcoin commission, but the cumulative trend of assets within five years remains unchanged.

Keywords: DRL; Futures Market; Strategy Gradient; PPO; DTSL-PPO

Contents

1 Introduction

1.1 Problem Background

Market traders are keen to buy and sell volatile assets with the goal of maximizing their total returns. How volatile assets can be profitable depends on a trader's choice of whether the assets in their portfolio should be bought, held or sold. Therefore, the choice of trading strategy in the investment process is of great importance.

Two typical volatile assets are gold and bitcoin. Gold has the characteristics of strong value preservation and light tax burden, while Bitcoin has the features of global circulation and convenient transaction. In order to rationally plan the investment of gold and bitcoin, we need to formulate trading strategies based on actual pricing sequence.

1.2 Restatement of the Problem

Through in-depth analysis and research on the background of problem, combined with the specific constraints given, the restatement of the problem can be expressed as follows:

(1) Develop a model that gives the best daily trading strategy based only on the pricing data for the day.

(2) Provide evidence to convince that the model provides the best strategy.

(3) Determine the sensitivity of the strategy to transaction costs.

(4) Communicate trading strategy, model and result to traders in the form of memos.

1.3 Literature Review

A number of researchers have previously contributed to trading strategy of volatile assets.

- Research on price fluctuation based on traditional economic model

Fabio Fornari et al. used the Auto Regressive Conditional Heteroskedasticity (ARCH) model to predict the stock markets of six countries, and concluded that the ARCH model was superior to the traditional Auto Regressive Moving Average (ARMA) model [1].

Wang Bo et al. used the ARMA-GARCH model to analyze the predictive ability of model for the daily rate of return when the sequence residuals obeyed different distributions, and came to the conclusion that the results obtained by the ARMA-GARCH model with t distribution are the closest to the real price fluctuations [2].

- Research on price fluctuation based on deep learning algorithm

Avraam Tsantekidis, Nikolaos Passalis et al. used a deep learning method based on convolutional neural networks. For high-frequency data, the study proved that convolutional neural networks are more suitable for such tasks than multi-layer neural networks and support vector machines [3].

Wu Wei et al. believed that neural networks can better fit nonlinear and noisy financial data [4].

- Research on price fluctuation based on reinforcement learning algorithm

Moody et al. applied the reinforcement learning algorithm to a single stock and asset portfolio to propose an RRL model [5].

J. W. Lee proposed the Q-trader stock trading system based on Q-Learning to complete the buying and selling of stocks [6].

- Research on price fluctuation based on deep reinforcement learning algorithm

Yue Deng et al. applied deep reinforcement learning to the representation and learning of financial signals, and verified the robustness of the method in the stock market and commodity futures market [7].

Qi Yue et al. adopted the deep deterministic policy gradient algorithm in deep reinforcement learning, which helps to avoid the overfitting problem [8].

In conclusion, we discover that current research exists papers considering reinforcement learning applied to futures market. However, when modeling, it is generally believed that each futures must be sold in each transaction, or purchased without leaving balance. Considering actual scenario and risk of investment, this is not realistic.

1.4 Our Work

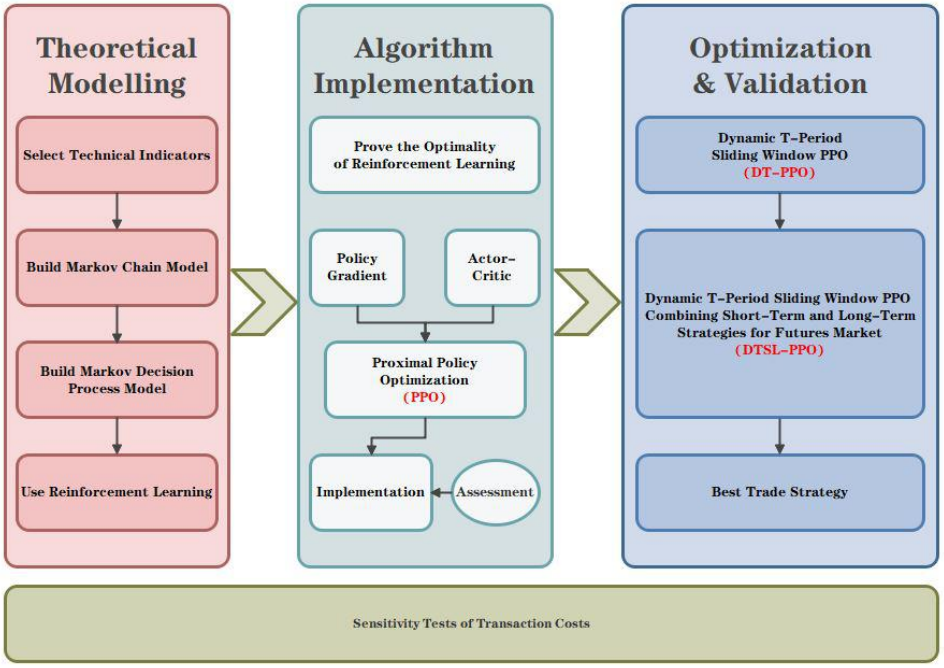


Figure 1: Flow Chart of Our Work

2 Assumptions and Explanations

Considering that practical problems always contain many complex factors, first of all, we need to make reasonable assumptions to simplify the model, and each hypothesis is closely followed by its corresponding explanation:

1. **The price on trading day is unchanged.**

➢ Due to the given data, we believe that the price of trading day will not change.So that after obtaining the price of the day, we can use it as the closing price to make a decision to trade. At the same time, there is no need to make multiple transactions in one day.

2. **Before trading, we know that the volatility of gold is small and the volatility of bitcoin is large, but we aren't aware of any fluctuation trend and range.**

➢ The different properties of gold and bitcoin should be taken as prior knowledge. Gold has always been regarded as a stable investment with little volatility. While Bitcoin, as an emerging business, has huge unknowns and high risks.

Additional assumptions are made to simplify analysis for individual sections. These assumptions will be discussed at the appropriate locations.

3 Notations

Some important mathematical notations used in this paper are listed in Table 1.

Table 1: Notations used in this paper

Symbol	Description
pg_t	gold price (USD per troy ounce)
pb_t	bitcoin price (USD per bitcoin)
C_t	balance (USD)

G_t	gold quantity (troy ounce)
B_t	bitcoin quantity (bitcoin)
X_t	state of each trading day
S_t	state of each trading day after mapping
A_t	trading action on each day
R	cumulative reward in trading process
π	strategy of action execution

Note: There are some variables that are not listed here and will be discussed in detail in each section.

4 Model Establishment

4.1 Background Analysis

The different properties of gold and bitcoin should be taken as prior knowledge. Gold has always been regarded as a stable investment with little volatility [9]. While Bitcoin, as an emerging business, has huge unknowns and high risks [9].

4.2 Model Analysis

Our goal is to establish a trading strategy for the two ever-changing futures prices that will ultimately make us profitable on unlimited investments in the five-year market. A common idea is to make predictions about future prices based on past prices, and then implement strategies based on the predictions. However, the investment market fluctuates greatly and is significantly influenced by social factors. Fama[10] proposed that past prices cannot predict future prices considering market efficiency theory and random walk theory. Therefore, we should ensure that our model can avoid such "prediction irrationality". At the same time, from common sense perspective, futures trading is also a continuous process that requires constantly evolving strategies, getting feedback from market, and trying to optimize trading strategies over time.

Based on this thinking, we adopt the **reinforcement learning (RL) model** that is more suitable for futures arbitrage. Reinforcement learning is a machine learning algorithm that "learns while playing". We can create an **agent** for gold and bitcoin futures arbitrage. In order to achieve the goal of making a profit in five years, the agent can learn from the market's feedback after making each trading day's decisions, and then adjust its strategy. Therefore, reinforcement learning methods can fully understand all underlying market laws. For the self-adjusting market price, it has excellent ability of learning and adapting.

On the other hand, **deep reinforcement learning (DRL)** has many unique advantages over current existing solutions.

- In contrast to Modern Portfolio Theory's strategy based solely on inventory returns, reinforcement learning (RL) is able to incorporate other relevant factors, for example, certain technical indicators such as Moving Average Convergence Divergence (*MACD*) and Relative Strength Index (*RSI*), and has satisfactory control over trend.
- In contrast to supervised machine learning, RL does not require large labeled training datasets. This is a significant advantage because today's exponentially growing data make labeling large datasets very time-consuming and laborious.
- In contrast to ML regression/classification models that predict the likelihood of future outcomes, as mentioned above, this "prediction" is unreasonable, and the core algorithm of reinforcement learning can avoid the unreasonableness caused by this predictive behavior.

4.3 Selection of Economic Indicators

Aiming to describe changes in market prices and current states effectively, we use the following economic indicators:

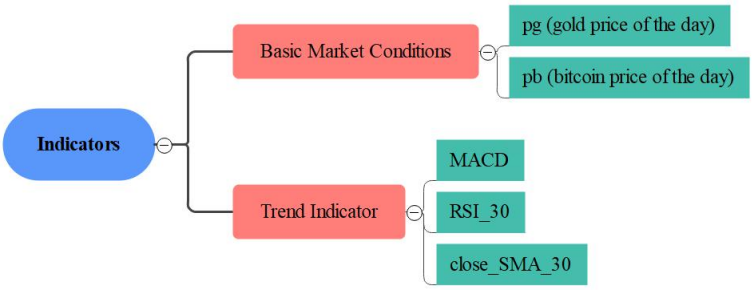


Figure 2: Economic Indicators

➢ **Price**

In order to facilitate the establishment of the model, we first conduct a preliminary analysis of the pricing data in LBMA-GOLD.csv and BCHAIN-MKPRU.csv. By natural logarithmic transformation of the original data, the visualization curves are shown in the figures below.

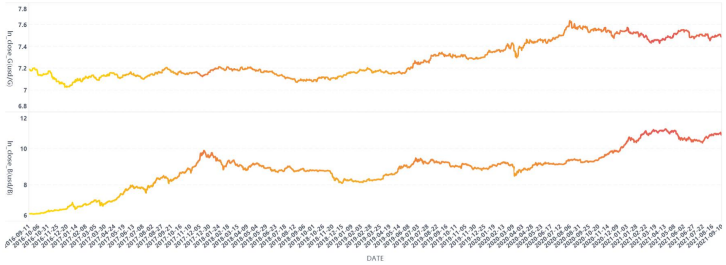


Figure 3: Visualization Curve of Daily Prices by Natural Logarithmic Transformation

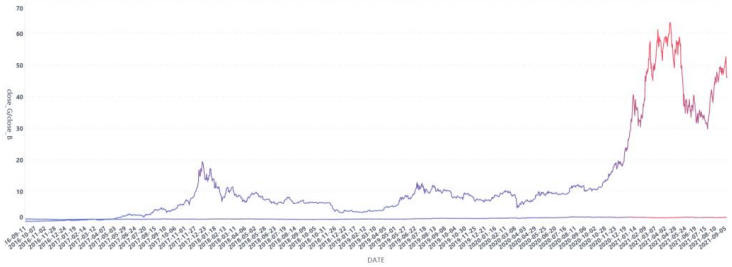


Figure 4: Visualization Curve of Daily Prices

By comparing and analyzing the visualization curves, we find that Gold price volatility is relatively small while Bitcoin price volatility is relatively large. At the same time, both show a certain upward trend. The Bitcoin price has three small peaks on 01/24/18, 09/16/19 and 05/08/21 respectively. By searching for information, this reflects the sharp expansion of Bitcoin market.

But here we only analyze the overall trend and grasp the role of five indicators. We do not know the future information when executing the action of the day. The same is true for the following indicator analysis of MACD and RSI.

➢ **MACD**

Moving Average Convergence Divergence (MACD) is an indicator to determine price momentum and short-term trend, and it is a trend-following indicator. The MACD indicator is made up of three components: MACD line, Signal line, MACD histogram.

MACD line equals the 12-day EMA minus the 26-day EMA.

Signal line equals a 9-day EMA of the MACD.

MACD histogram equals the MACD minus the MACD Signal line.

Where Exponential Moving Average (EMA) is expressed in the formula below.

$$EMA_i(T) = \frac{T-1}{T+1} EMA_{i-1}(T) + \frac{2}{T+1} C_i \quad (1)$$

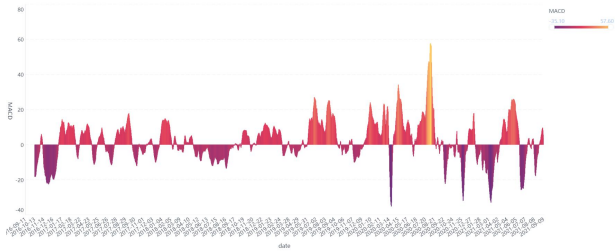


Figure 5: MACD of Gold

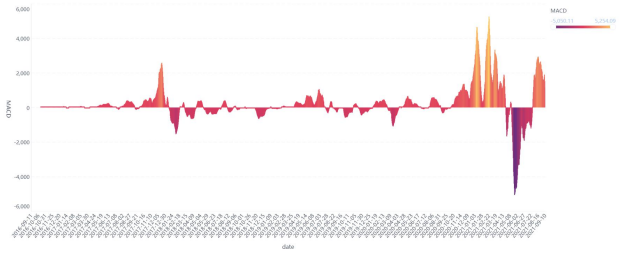


Figure 6: MACD of Bitcoin

➢ **RSI_30**

The relative strength index (RSI) represents the size of recent gains and losses, during a specified time period, and measures the speed of these price movements. It is therefore primarily used to identify potential overbought and oversold situations for a particular currency pair.

The indicator is calculated the following way:

$$RSI = 100 - [100/(1+RS)] \quad (2)$$

where,

$$RS = SAG / SAL \quad (3)$$

$$\text{Average Gain: } AG = \sum_{i=1}^{t} G_i \quad (4)$$

$$\text{Average Loss: } AL = \sum_{i=1}^{t} L_i \quad (5)$$

To create a Smoothed Average Gain, the following calculation is performed:

$$\text{Smoothed Average Gain: } SAG = [(AG_p \times 13 + G_c)] \quad (6)$$

$$\text{Smoothed Average Loss: } SAL = [(AL_p \times 13 + L_c)] \quad (7)$$

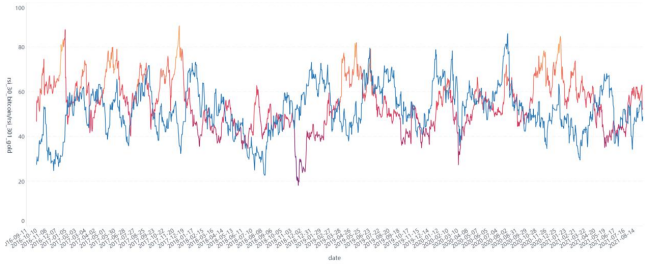


Figure 7: RSI_30 of Gold and Bitcoin

➢ **close_SMA_30**

EMA is generally considered more suitable for short-term trading. SMA lags slowly and tends to smooth out price action over time, making it a good trend indicator for long-term trade evaluations with less volatility.

$$SMA_i(T) = \frac{T+1-M}{T+1} SMA_{i-1}(T) + \frac{M}{T+1} C_i \quad (8)$$

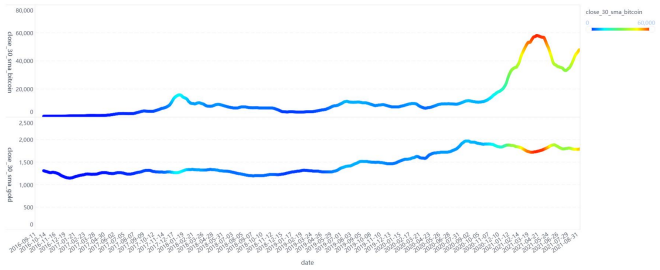


Figure 8: close_SMA_30 of Gold and Bitcoin

4.4 Establishment of Markov Chain Model

Based on the establishment of above economic indicators and the high volatility of futures market, we adopt the Markov Decision Model to model the daily trading strategies. Markov Decision Model is an extension of **Markov Chain Model**. Its biggest feature is that in time series, the decision at the current moment is only related to the information of the previous *n* moments.

For each trading day, it is necessary to make a decision on that day to choose to buy or sell gold and bitcoin based on past information and its own state. Therefore, we establish the basic market and decision-making as the state of each trading day:

$$X_t = [pg_t \quad pb_t \quad C_t \quad G_t \quad B_t \quad MACD_t \quad RSI\_30_t \quad close\_SMA\_30_t] \tag{9}$$

According to the establishment of previous economic indicators, it can be considered that the decision of the day is related to the state of previous 26 days. Thus we establish a **26-order Markov chain**.

$$p\{X_n = j \mid X_{n-1} = i, X_{n-2} = k, \cdots, X_0 = m\} = p\{X_n = j \mid X_{n-1} = i, X_{n-2} = k, \cdots, X_{n-26} = l\} \tag{10}$$

In order to facilitate the understanding and analysis, we establish the mapping of the symbol sequence of length 26 to the state *S*, and obtain the **1-order Markov chain**:

$$(X_1 \cdots X_{26}) \to S_j \tag{11}$$

According to the transformation relationship, there is a state transition probability:

$$p(X_{27} \mid X_1 \cdots X_{26}) = p(S_{j+1} \mid S_j) \tag{12}$$

Where $(X_2 \cdots X_{m+1}) \to S_{j+1}, (X_1 \cdots X_m) \to S_j$.

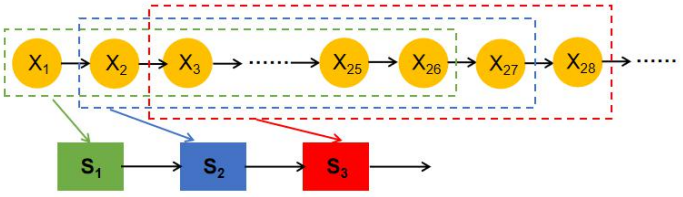


Figure 9: Diagram of Markov chain mapping

4.5 Establishment of Markov Decision Process

Based on the above-mentioned 1-order Markov Chain Model, we further establish **Markov Decision Process (MDP)**. For a decision, the "action" performed and the "reward" obtained are most important.

The transition between states is dependent on "action", that is, the amount of gold and bitcoin chosen to sell or buy on that day. It can be described as:

$$A_t = [m_g \quad m_b] \tag{13}$$

Where $m_g \in [-a, b]$ indicates the amount of gold bought or sold (troy ounce), $m_b \in [-c, d]$ indicates the amount of bitcoin bought or sold (bitcoin), $a = G_t$, $b = \frac{C_t}{pg_t}$, $c = B_t$, $d = \frac{B_t}{pb_t}$.

The probability that the process moves into its new state S' is influenced by the chosen action. Specifically, it is given by the state transition function $P_A(S,S')$. Thus, the next state S' depends on the current state S and the decision maker's action A. But given S and A, it is conditionally independent of all previous states and actions.

And every time a state change occurs, an evaluation can be made for this change. We define state value function as $V(t)$. $V(t)$ describes the asset of the day, which consists of the value of balance and the value of futures held.

$$V(t) = C_t + DA_t \tag{14}$$

$$DA_t = G_t \times pg_t + B_t \times pb_t \tag{15}$$

It is easy to understand that the difference of state value function is the profit or loss of the trading day relative to the previous one:

$$\begin{cases} V(t) > V(t-1), profit \\ V(t) = V(t-1), break\_even \\ V(t) < V(t-1), loss \end{cases} \tag{16}$$

We define this difference as the reward:

$$r(t) = V(t) - V(t-1) \tag{17}$$

Further, the cumulative reward for performing a complete Markov sequence is:

$$R = \sum_{t=0}^{N} r(t) \tag{18}$$

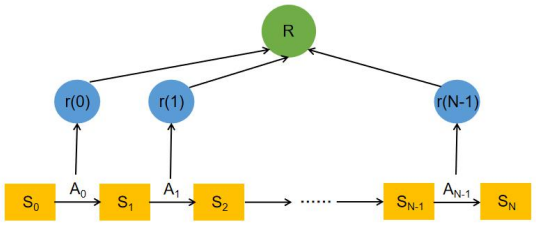


Figure 10: Diagram of Markov Sequence Reward

Obviously, the reward depends on multiple factors. Profitability depends not only on the state of the day and previous day, but also on the actions performed on previous day.

$$S \times A \times S' \Rightarrow R \tag{20}$$

$$r(t) = R(S_t, A_t, S_{t+1}) \tag{21}$$

4.6 Establishment of Basic Reinforcement Learning Model

Summarizing the above model, we obtain a clear quadruple $E =< X, A, P, R >$. Based on this quadruple, we further model using the idea of reinforcement learning as follows.

Firstly, the **interaction relationship model** between agent and environment is established. The agent is the person who makes the investment and is the main body of the action. The actions performed act on market (environment), and cause changes in state and reward. The most important point at this time is that the agent needs to learn this "response", adjusts its subsequent actions according to the reward obtained, and continuously strengthens its cognition of the environment through this interaction.

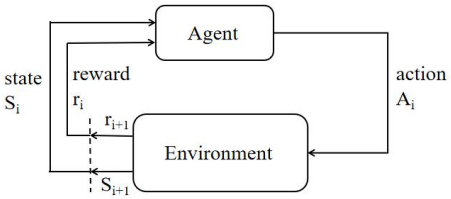


Figure 11: Interaction Relationship Model between Agent and Environment

After a series of interactive learning, the agent can learn a rule that determines actions according to the state, which is defined as "strategy".

$$A_t = \pi(S_t) \tag{22}$$

According to the randomness of futures market, the strategy should have a certain randomness, which is expressed as:

$$A_t \sim \pi(\cdot \mid S_t) \tag{23}$$

In other words, reinforcement learning is an advisor in the process of arbitraging gold and bitcoin. It learns a trading strategy by continuously learning the relationship between each day's trading decisions and the next day's profit or loss, which is the ultimate goal of reinforcement learning.

In order to gain the maximum ultimate profit at the end of five-year period, the cumulative reward for all actions needs to be maximized. For futures market, the environment is very complex and random, and it is not feasible to directly calculate the cumulative rewards for all subsequent trading days. At the same time, for purpose of making the exploration phase more efficient, **decay factor γ** is defined for the cumulative reward, and the goal of action on each trading day is to maximize the γ discount reward:

$$R_\gamma(t) = \sum_{i=0}^{\infty} \gamma^i r(i), \gamma \in (0,1) \tag{24}$$

If a series of interactions are performed with this goal, we define the entire sequence of states and actions as **motion trajectory τ**.

$$\tau = (S_0, A_0, S_1, A_1, \ldots) \tag{25}$$

Considering that the state and action of each trading day will change with strategy, according to the probability distribution of strategy and state transition, the probability distribution of state and action of the next T trading days is:

$$P(S_0, A_0 \mid \pi) = P(S_0)\pi(A_0 \mid S_0) \tag{26}$$

$$\begin{aligned} P(S_0, A_0, S_1, A_1 \mid \pi) &= P(S_0, A_0 \mid \pi)P(S_1 \mid S_0, A_0)\pi(A_1 \mid S_1) \\ &= P(S_0)\pi(A_0 \mid S_0)P(S_1 \mid S_0, A_0)\pi(A_1 \mid S_1) \end{aligned} \tag{27}$$

$$\cdots$$

$$P(\tau \mid \pi) = P(S_0)\prod_{i=0}^{T-1} \pi(A_i \mid S_i)P(S_{i+1} \mid S_i, A_i) \tag{28}$$

At this time, we calculate the expected return as:

$$\underset{\tau \sim \pi}{E}\left[R_\gamma(\tau)\right] = \int_\tau P(\tau \mid \pi) \times R_\gamma(\tau) \tag{29}$$

The goal of reinforcement learning is to find the optimal strategy π^* on every trading day to maximize the expected return of each step.

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \underset{\tau\sim\pi}{E}\left[R_\gamma(\tau)\right] \tag{30}$$

5 Model Implementation and Optimization

5.1 Overall Analysis

In this section, we first prove the "optimality" of reinforcement learning strategies. There may be multiple optimal strategies for reinforcement learning tasks, but the ultimate goal is the same. It can be proved mathematically that if the strategy can satisfy the **optimal Bellman equation**, then the optimal strategy has been found.

For the application scenario of futures trading market, we cannot predict the future situation.So we must adopt model-free algorithm and wait for the feedback from real world step by step before taking the next action. Meanwhile, since our action space is continuous, we choose **PPO algorithm** that performs well in most applications and implement it on the basis of **Open AI**.

5.2 Optimal Strategy

Based on analysis of MDP decision-making and reinforcement learning, on each trading day, when the state is *S*, the optimal action should be selected to maximize the expected return. If it is considered that subsequent strategy remains unchanged, we define the expected return from state *S* as the **same-strategy value function** (formula 31). If it is considered that subsequent actions should be performed according to the optimal strategy at each step, we define the expected return from state *S* as the **optimal value function** (formula 32).

$$V^\pi(S) = \underset{\tau\sim\pi}{E}\left[R_\gamma(\tau) \mid S_0 = S\right] \tag{31}$$

$$V^*(S) = \max_{\pi} \underset{\tau\sim\pi}{E}\left[R_\gamma(\tau) \mid S_0 = S\right] \tag{32}$$

Due to the Markov property of MDP, the state-value function has a very simple recursive form.

$$\begin{aligned} V^\pi(S) &= \underset{\tau\sim\pi}{E}\left[R_\gamma(\tau) \mid S_0 = S\right] \\ &= \underset{A\sim\pi(\cdot|S)}{E}\left[\sum_{i=0}^{\infty}\gamma^i r(i) \mid S_0 = S\right] \\ &= \underset{\substack{A\sim\pi(\cdot|S)\\ S'\sim P(\cdot|S,A)}}{E}\left[r(S,A,S') + \gamma\sum_{i=0}^{\infty}\gamma^i r(i) \mid S_0 = S'\right] \\ &= \underset{\substack{A\sim\pi(\cdot|S)\\ S'\sim P(\cdot|S,A)}}{E}\left[r(S,A,S') + \gamma V^\pi(S')\right] \end{aligned} \tag{33}$$

This equation also corresponds to our previous analysis. That is, after obtaining the next day's reward according to the strategy on that day, the expected return on current trading day is equal to multiplying the expected return starting from the new state on next trading day by decay factor γ.

That is to say, the expected return can be divided into two parts. One part is the reward of next day after today's action is executed, and the other part is the expected return that can be obtained by continuing to use certain strategy after executing the action. We define the latter as the state-action function: the expected return after executing action *A* when the state is *S* on each trading day. Also according to the update of strategy, it is defined as the **same-strategy state-action function** (formula 34) and the **optimal state-action function** (formula 35).

$$Q^{\pi}(S,A) = \underset{\tau\sim\pi}{E}\left[R_{\gamma}(\tau) \mid S_0 = S, A_0 = A\right] \tag{34}$$

$$Q^{*}(S,A) = \max_{\pi} \underset{\tau\sim\pi}{E}\left[R_{\gamma}(\tau) \mid S_0 = S, A_0 = A\right] \tag{35}$$

The same as the proof of equation 33, the conversion relationship between the state-value function and the state-action function can also be obtained:

$$Q^{\pi}(S,A) = \underset{S'\sim P(\cdot|S,A)}{E}\left[r(S,A,S') + \gamma V^{\pi}(S')\right] \tag{36}$$

It also has Markov property and satisfies the equation 37:

$$Q^{\pi}(S,A) = \underset{S'\sim P(\cdot|S,A)}{E}\left[r(S,A,S') + \gamma Q^{\pi}(S')\right] \tag{37}$$

We obtain such an optimization idea by analyzing the relationship between the state-value function and the state-action function. We change the action selected by strategy to the current optimal action. When each step is performed, it is not to evaluate the expected return of the day, but to evaluate the expected return after executing a certain action, which can optimize the action of each step and find optimal strategy. According to this idea, the above equation is modified: the sum of actions is changed to optimization.

$$Q^{*}(S,A) = \underset{S'\sim P(\cdot|S,A)}{E}\left[r(S,A,S') + \gamma \max_{A'} Q^{*}(S',A')\right] \tag{38}$$

This equation is the optimal Bellman equation, and then we prove that through this equation, the optimal strategy can be found. We make the corresponding strategy after the action is changed as π'. The condition for changing action is: assuming that when each subsequent step is executed with strategy π, for the current action, it can be found that if action A' calculated by strategy π' is executed, the expected benefit will be greater than it obtained with strategy π. This condition can be expressed as below.

$$\underset{A'\sim\pi'(S)}{Q}{}^{\pi}(S,A') \geq V^{\pi}(S) \tag{39}$$

Combined with Bellman equation, we continue to derive the formula:

$$\begin{aligned} V^{\pi}(S) &\leq \underset{A'\sim\pi'(S)}{Q}{}^{\pi}(S,A')V^{\pi}(S) \\ &= \underset{\substack{S'\sim P(\cdot|S,A)\\ A'\sim\pi'(S)}}{E}\left[r(S,A',S') + \gamma V^{\pi}(S')\right] \\ &\leq \underset{\substack{S'\sim P(\cdot|S,A)\\ A'\sim\pi'(S)}}{E}\left[r(S,A',S') + \gamma \underset{A''\sim\pi'(S')}{Q}{}^{\pi}(S',A'')\right] \\ &= V^{\pi'}(S) \end{aligned} \tag{40}$$

The first row to the second row and the third row to the fourth row use the conversion relationship of formula 37, and the second row to the third row use the inequality relationship of formula 39. After two iterations and inequality relation, we prove the following conclusion. If the conditions of formula 39 are satisfied, it can be considered that it is scientific to change strategy π to π' on the day, and the value function is monotonically increasing for each improvement of strategy.

So for the current strategy π', it can be safely improved to:

$$\pi' = \underset{A}{\operatorname{argmax}}\, Q^{\pi}(S,A) \tag{41}$$

Until π and π' are consistent (the optimal Bellman equation is satisfied), the strategy must be the optimal strategy.

5.3 Strategy Gradient Algorithm

It can be seen from section 4.6 that the optimization objective of optimal strategy is:

$$\pi^* = \arg\max_{\pi} \underset{\tau\sim\pi}{E}\left[R_\gamma(\tau)\right] \tag{42}$$

The update of strategy is actually to update its parameters, so the strategy is written as π_θ below. And $\underset{\tau\sim\pi}{E}\left[R_\gamma(\tau)\right]$ above is naturally related to the parameters, so it is rewritten as R_θ.

Based on each trading day, we ought to find optimal strategy to maximize the expected return. An intuitive idea is to try to find all possible trajectories τ, calculate $R_\gamma(\tau)$ of each trajectory to compare, and then update the strategy in reverse. We treat "find all" as sampling, that is, expand the sampling strategy according to current π_θ, collect a batch of trajectories, and the distribution of trajectories is as formula 28.

In the update of strategy, in order to maximize the target value R_θ, we use the idea of **gradient descent** to find the direction in which R_θ changes the most on θ.

$$\begin{aligned}\nabla R_\theta &= \int_\tau R_\gamma(\tau)P(\tau\mid\pi_\theta)\frac{\nabla P(\tau\mid\pi_\theta)}{P(\tau\mid\pi_\theta)}\\ &= \int_\tau R_\gamma(\tau)P(\tau\mid\pi_\theta)\nabla\log P(\tau\mid\pi_\theta)\\ &= \underset{\tau\sim P_\theta(\tau)}{E}\left[R_\gamma(\tau)\nabla\log P(\tau\mid\pi_\theta)\right]\end{aligned} \tag{43}$$

Combined with formula 28, we can get the gradient update formula:

$$\begin{aligned}\nabla R_\theta &= \underset{\tau\sim P_\theta(\tau)}{E}\left[R_\gamma(\tau)\nabla\log P(\tau\mid\pi_\theta)\right]\\ &= \underset{\tau\sim P_\theta(\tau)}{E}\left[R_\gamma(\tau)\nabla\log P(\tau\mid\pi_\theta)\right]\\ &= \underset{\tau\sim P_\theta(\tau)}{E}\left[R_\gamma(\tau)\nabla\log P(S_0)\prod_{\mathrm{i}=0}^{\mathrm{T}-1}\pi_\theta(A_i\mid S_i)P(S_{\mathrm{i}+1}\mid S_{\mathrm{i}},A_{\mathrm{i}})\right]\\ &= \underset{(S_t,\mathrm{A}_{\mathrm{t}})\sim P_\theta(\tau)}{E}\left[R_\gamma(\tau)\nabla\log\pi_\theta(A_t\mid S_t)\right]\end{aligned} \tag{44}$$

When updating strategy, since our goal is to find the highest point, not the lowest point, we use its idea to perform gradient ascent in reverse, and the update formula of neural network is:

$$\theta_{\mathrm{new}} \leftarrow \theta_{\mathrm{old}} + \eta\nabla R_{\theta_{\mathrm{old}}} \tag{45}$$

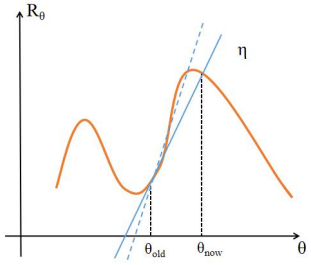


Figure 12: Diagram of Gradient Ascent

5.4 Actor-Critic (AC) Framework Combined with Deep Learning

Looking back at the reinforcement learning process of the interaction between agent and environment, on each trading day, two core tasks that agent needs to perform are executing actions and evaluating strategies. For complex MDP problems, both state space and action space are huge. In the face of problems with high state and action space dimensions, the use of **neural network** is the most effective method. It has the most efficient function approximation ability and can effectively extract data features from the original data. The network is constructed for the two cores as follows:

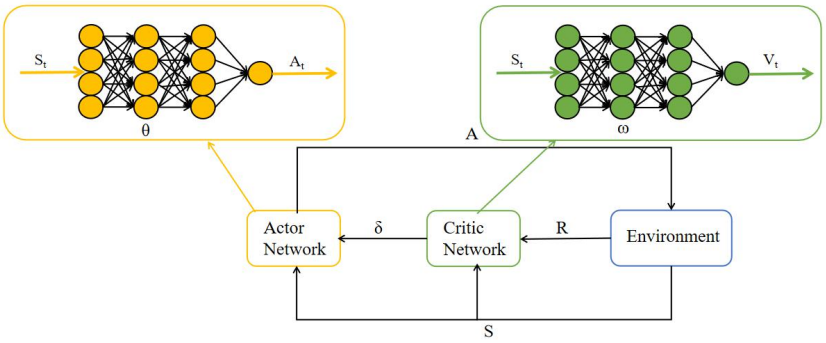


Figure 13: Diagram of Neural Network

Agents are controlled by Actor Network, interact with environment, and collect states, actions, and rewards. Actor network is like an actor that performs performances (actions A_t) based on the state S_t. Critic Network is like a judge, which estimates the output error based on S_t, S_{t+1}, and r_t to evaluate the difference between the reward obtained by action of Actor Network (actor) and the expected reward. Actor Network (actor) relies on this evaluation to judge how good or bad its actions are, thereby improving its own performance (parameter θ).

In the reinforcement learning process, firstly, according to the state S_t of the day and certain strategy, the action A_t to be performed on the day can be obtained. Then we can acquire the expected return $V^{\pi_\theta}(S_t, A_t)$ under the current strategy. The return can be divided into two parts. One part is the reward r_{t+1} of next day after today's action is executed, and the other part is the expected return $Q^{\pi_\theta}(S_t, A_t)$ that can be obtained by continuing to use certain strategy after executing the action. $V^{\pi_\theta}(S_t, A_t)$ describes the expected return when the action is not performed, and $Q^{\pi_\theta}(S_t, A_t)$ describes the expected return after the action is performed. We define the difference as the **advantage function**.

$$\hat{A}_t = Q^{\pi_\theta}(S_t, A_t) - V^{\pi_\theta}(S_t) \tag{46}$$

According to formula 36, there is:

$$\hat{A}_t = \underset{S_{t+1} \sim P(\cdot|S_t, A_t)}{E} \left[r(S_t, A_t, S_{t+1}) + \gamma V^{\pi_\theta}(S_{t+1}) \right] - V^{\pi_\theta}(S_t) \tag{47}$$

We can approximate the conditional expectation above by **Monte Carlo**, substituting the observed S_t, S_{t+1}, r_t with $V^{\pi_\theta}(S_t; w)$ computed by the value function, and define it as temporal difference (TD):

$$TD\delta = r_t + \gamma V^{\pi_\theta}(S_{t+1}; \omega) - V^{\pi_\theta}(S_t; \omega) \tag{48}$$

TD describes the difference between model estimation and true observation.

Therefore, Critic Network as a judge should ensure that the difference is as small as possible, and use this as the **loss function** to train Critic Network:

$$L(\omega) \overset{\Delta}{=} \frac{1}{2}\left[Q^{\pi_\theta}(S_t, A_t) - V^{\pi_\theta}(S_t; w)\right]^2 \tag{49}$$

In order to achieve minimization, Critic Network is updated according to the gradient descent idea:

$$\omega_{new} \leftarrow \omega_{old} - \alpha L(\omega) \tag{50}$$

For the update of Actor Network, as analyzed in Section 5.3, the goal is to maximize R_θ. Combined with formula 31, we can find another form of formula 44, and it can be discovered that the update of this strategy will indeed be based on the judges' scores.

$$\begin{aligned} \nabla R_\theta &= \underset{(S_t, \mathrm{A}_t)\sim P_\theta(\tau)}{E}\left[R_\gamma(\tau)\nabla \log \pi_\theta(A_t \mid S_t)\right] \\ &= \underset{\tau\sim\pi_\theta}{E}\left[R_\gamma(\tau)\right]\nabla \log \pi_\theta(A_t \mid S_t) \\ &= V^{\pi_\theta}(S_t)\nabla \log \pi_\theta(A_t \mid S_t) \end{aligned} \tag{51}$$

5.5 Proximal Policy Optimization (PPO)

We adopt PPO algorithm to optimize the update strategy as follows.

We assume that the original strategy is π_θ, and it is updated to $\pi_{\theta'}$ after the above gradient method. However, this method makes the sampling strategy change after one update, so the trajectories previously collected by strategy π_θ cannot be used.

Assuming that original strategy is sampled using $p(x)$, the expectation of sample function is:

$$E_{x\sim p}\left[f(x)\right] = \int f(x)p(x)dx \tag{52}$$

After that, it is modified to use $q(x)$ for sampling, then the expectation is $E_{x\sim q}\left[f(x)\right]$. Further, it is easy to derive the following connected equation:

$$E_{x\sim p}\left[f(x)\right] = \int f(x)p(x)dx = \int f(x)q(x)\frac{p(x)}{q(x)}dx = E_{x\sim q}\left[\frac{p(x)}{q(x)}f(x)\right] \tag{53}$$

Therefore, when using $\pi_{\theta'}$ for sampling and gradient updating, in order to make the previous sampling still available, a **correction term** can also be similarly introduced to improve the gradient update formula:

$$\nabla R_\theta = \underset{(S_t, \mathrm{A}_t)\sim P_{\theta'}(\tau)}{E}\left[\frac{\pi_\theta(A_t \mid S_t)}{\pi_{\theta'}(A_t \mid S_t)}R_\gamma(\tau)\nabla \log \pi_\theta(A_t \mid S_t)\right] \tag{54}$$

According to $f(x)\nabla \log f(x) = \nabla f(x)$, we can continue to modify the gradient update formula:

$$\nabla R_\theta = \underset{(S_t, \mathrm{A}_t)\sim P_{\theta'}(\tau)}{E}\left[\frac{\nabla \pi_\theta(A_t \mid S_t)}{\pi_{\theta'}(A_t \mid S_t)}R_\gamma(\tau)\right] \tag{55}$$

Observing this formula, it can be discovered that the gradient update is actually looking for the maximum direction of the expected return under the new strategy on θ:

$$R_{\theta'} = \underset{(S_t, \mathrm{A}_t)\sim P_{\theta'}(\tau)}{E}\left[\frac{\pi_\theta(A_t \mid S_t)}{\pi_{\theta'}(A_t \mid S_t)}R_\gamma(\tau)\right] \tag{56}$$

At the same time, it is worth noting that we should ensure that the sampling distribution of $\pi_{\theta'}$ and π_θ cannot be too far apart. Aiming to remind the gradient update to pay attention to this, **KL distance** (define the distance between two strategies) is introduced as a regular term. When KL distance is too large, R_θ will decrease accordingly:

$$R_{\theta'} = \underset{(S_t, A_t)\sim P_{\theta'}(\tau)}{E}\left[\frac{\pi_\theta(A_t \mid S_t)}{\pi_{\theta'}(A_t \mid S_t)} R_\gamma(\tau)\right] - \beta KL(\pi_\theta, \pi_{\theta'}) \tag{57}$$

5.6 Model Implementation

At the starting point of reinforcement learning, random strategies are often used to explore. But for futures trading, this initial "random" risk is very large. So we decide that the first three months from 2016/09/11 will not actually invest, but the strategy will still be trained on the data of these three months. Then we follow the initial strategy to "learn while trading". Aiming at making the model to learn recent trends, we use three months as a "unit" to adjust the model's strategy for the next month, creating a **sliding window model**, as shown in the following figure.

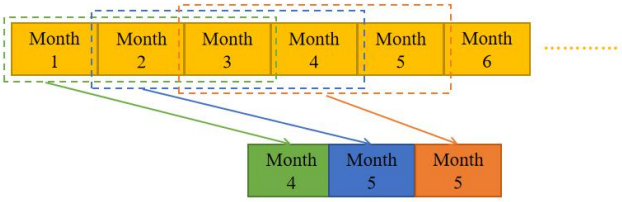


Figure 14: Diagram of Sliding Window Model

It is worth noting that although three-month data are used to adjust the next month's strategy each time, this does not mean that other previous data have been thrown away. Based on the core of reinforcement learning, the parameters of neural network are updated every three months of training. So all previous experience is stored in these parameters.

5.7 Model Evaluation

The three indicators of **cumulative reward**, **annualized rate of return** and **Sharpe ratio** are to quantify the trading decision-making income of the model, so as to achieve the function of evaluating the performance of our model. The definitions and formulas of the three are as follows.

- **Cumulative Reward**

Cumulative reward refers to the accumulated reward after each transaction in the trading cycle, and its calculation formula is expressed below:

$$R = V_T - C_0 \tag{58}$$

Where V_T refers to capital price on the last day, C_0 refers to investor's principal.

The formula for calculating the cumulative reward rate R_R is:

$$R_R = R / C_0 \tag{59}$$

Cumulative reward is an intuitive indicator to measure the performance of trading model. The greater the cumulative reward is, the stronger the model's investment decision-making ability is and the higher the model's ability to obtain high investment returns is.

- **Annualized Rate of Return**

The transactions conducted in this article are high-frequency. The data set is not in units of years, and the actual duration is shorter in natural days. Both the cumulative reward and the cumulative reward rate mainly reflect the investment results during the collection period. For investors who spend several years in practice, the measurement range is somewhat narrow. So

we also use the annualized rate of return to measure the investment ability of our model. The calculation formula is as follows:

$$R_a = R_R * \frac{365}{n} \tag{60}$$

Where R_R is the cumulative reward rate over n days. In order to simplify calculation, we do not consider the impact of compound interest.

- **Sharpe Ratio**

Sharpe Ratio (*SR*), also known as Sharpe Index, combines the return and risk of a portfolio to measure how much excess return brings when an investor bears per unit of total risk, helping investors build a portfolio that minimizes risk. Its calculation formula is as follows.

$$SR = \frac{E(R_p) - R_f}{\sigma_p} \tag{61}$$

Where $E(R_p)$ is the expected reward rate of the portfolio, R_f is the risk-free rate of reward, and σ_p is the volatility of the portfolio (the risk of the portfolio).

After determining the evaluation strategy, we test sliding window model and get the following results. R_a has reached 54.929%, which is already a very satisfactory strategic effect. But it is clear that this strategy still has major flaws, and we will perform three progressive optimizations next.

Table 2: Sliding Window Model Performance

Total Return Rate *R*	Annualized Rate of Return *Ra*	Sharpe Ratio *SR*
2286.173%	54.929%	1.02

6 Improved Deep Reinforcement Learning

6.1 Reasonable Allocation Strategy of Two Assets

In previous PPO algorithm, the goal is only maximizing returns, while ignoring potential risks. However, risks should also be controlled when choosing investment strategies. The proportion of investment in two assets largely determines the risk level of our investment, so reasonable combination of investment in two assets should be made.

We define daily portfolio weight as ϖ , return as a , and return covariance matrix as $\sum$, which is actually determined by actions taken each day.

According to modern asset portfolio theory, **risk aversion coefficient** λ is introduced to measure the maximum acceptable risk. We add risk as a **regular term** to the objective function, and the daily objective is:

$$\max_{\varpi} \quad \varpi' a - \frac{\lambda}{2} \varpi' \Sigma \varpi \tag{62}$$

Where $\lambda = \frac{CR_\gamma(\tau) - A}{D}$, $C = 1'\Sigma^{-1}1, A = 1'\Sigma^{-1}a, B = a'\Sigma^{-1}a, D = BC - A^2$.

After the daily objective is modified, the corresponding PPO strategy update formula is revised as:

$$R_{\theta'} = \underset{(S_t, A_t) \sim P_{\theta'}(\tau)}{E} \left[\frac{\pi_\theta(A_t \mid S_t)}{\pi_{\theta'}(A_t \mid S_t)} \left[R_\gamma(\tau) - \frac{\lambda}{2} \varpi_t' \sum_t \varpi_t \right] \right] - \beta KL(\pi_\theta, \pi_{\theta'}) \tag{63}$$

Then we acquire the best weight according to strategy.

$$\varpi_t \sim \pi_{\theta'}(\cdot \mid S_t) \tag{64}$$

However, risk aversion coefficient is not a parameter that is intuitively the easiest to obtain. We define the maximum tolerable risk $\sigma_{\max}$. The daily objective is to maximize returns while keeping risks within maximum risk tolerance.

$$\begin{aligned} &\max_{\varpi} \quad \varpi' a \\ &s.t. \quad \varpi' \Sigma \varpi \le {\sigma_{\max}}^2 \end{aligned} \tag{65}$$

The relationship between $\sigma_{\max}$ and λ is derived as follows:

$$\lambda = \left(\sqrt{\frac{{\sigma_{\max}}^2 C - 1}{D}} \right)^{-1} \tag{66}$$

6.2 Dynamic T-Period Sliding Window Policy (DT-PPO)

For the implementation and result analysis of PPO algorithm in section 5.6, we summarize its main problems. The first point is that the start-up is too slow in the early stage, and it takes three months to learn before the transaction can be officially launched. The second point is that the sliding window slides in units of "three months", focusing entirely on the learning of short-term trends within three months and ignoring long-term trends in annual quantities, which makes it easy to miss the actual peak to some extent.

On account of this analysis, we build the following **Dynamic T-Period Sliding Window Model (DT-PPO)**. Under the condition of ensuring no blind exploration, the model shortens the start-up interval to two months, and gradually expands the period size of the sliding window, which can mainly learn long-term trends for strategy distribution in the later stage.

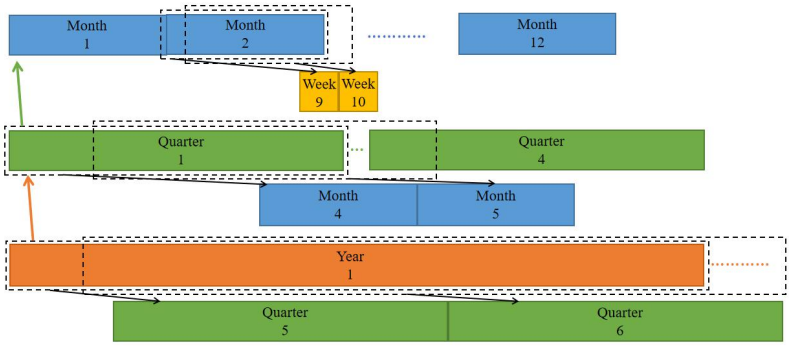


Figure 15: Dynamic Period Policy Framework

The framework of this strategy is shown in the figure above. First, in order to obtain economic indicators such as *MACD*, *RSI*, and *SMA*, we only conduct data collection in the first month of 2016/09/11. After that, when one month's data is collected, we use the data for training, and also do not trade in that month. Further, we apply the resulting strategy to transactions in one week after the end of the month, and adopt "one month" as the T1-period of sliding window. When one quarter's data is collected, we apply the resulting strategy to transactions in one month after the end of the quarter, and adopt "one quarter" as the T2-period of sliding window. When one year's data is collected, we apply the resulting strategy to transactions in one quarter after the end of the year, and adopt "one year" as the T3-period of sliding window.

6.3 Long-term and Short-term Balance Based on DT-PPO (DTSL-PPO)

In section 4.2, the investment strategy is revised with the risk aversion coefficient, but one problem of DT-PPO is that it pays more attention to the long-term trend in the later stage, which leads to the inability to respond quickly to the short-term trend. Therefore, we further propose a **balancing strategy** for long-term and short-term trends.

Short-term strategic asset allocation takes long-term strategic asset allocation plan as the "anchor". Therefore, the monthly asset allocation strategy should be based on the annual asset allocation strategy plan, and try to incorporate long-term and short-term asset allocation into the **unified framework SL** for research. The framework is shown as follows.

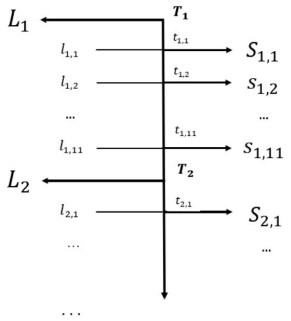


Figure 16: Framework of DTSL-PPO

T_i represents the starting point of the *i* year, which is not natural year, but transaction cycle. L_i represents the annual strategic asset allocation plan in the *i* year, and $l_{i,j}$ represents the actual asset allocation weight in the *j* month of *i* year. Taking the long-term annual strategy as the "anchor", we revise the short-term monthly asset allocation plan. That is, the short-term strategic asset allocation plan $S_{i,j}$ for the *j* month of *i* year is obtained.

We then amend the long-term and short-term strategic asset allocations based on risk aversion coefficient. First, we define an **investment benchmark** ϖ_b, which is an annual benchmark with a long changing period. Under this benchmark, the quantitative long-term risk aversion coefficient is:

$$\lambda_L = \left(\sqrt{\frac{\varpi_b' \Sigma \varpi_b C - 1}{D}} \right)^{-1} \tag{67}$$

Using λ_L to modify the maximization objective of formula 62, we obtain the annual strategic asset allocation plan L_i of the i-th year, that is, to make a weighted deployment ϖ_L for the current year. As previously analyzed, short-term strategies should be attached to long-term strategies. Hence, based on the existing annual strategic plan L_i, when analyzing short-term strategies, ϖ_L should be used as benchmark. The quantitative short-term risk aversion coefficient is:

$$\lambda_S = \left(\sqrt{\frac{\varpi_L' \Sigma \varpi_L C - 1}{D}} \right)^{-1} \tag{68}$$

Using λ_S to modify the maximization objective of formula 62, we obtain the monthly strategic asset allocation plan $U_{i,j}$ of the *j* month of *i* year.

Based on the above three-step progressive improvement, DTSL-PPO strategy is adopted to test in the scenario. The results are shown in table below.

Table 3: DTSL-PPO Strategy Results

Total Return Rate *R*	Annualized Rate of Return *Ra*	Sharpe Ratio *SR*	Annual Volatility Rate
4065.38%	70.365%	1.14	65.849%

The improved strategy makes annualized interest rate reaching 70.365%, which is an excellent level. In addition, average Sharpe Ratio reaches 1.14. When the market develops well, volatility Sharpe Ratio can even reach 6.00, which proves that the return of investment is greater than the risk.

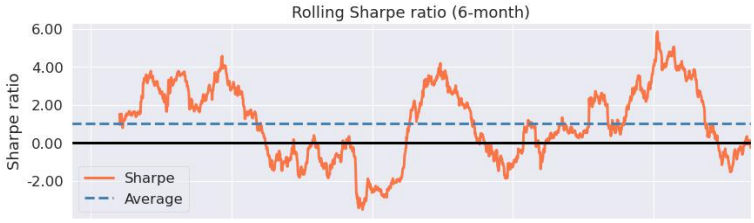


Figure 17: Rolling Sharpe Ratio Curve

According to the training of DTSL-PPO, we get the investment strategy for five years as follows: 18(a) is gold trading timing figure, 18(b) is bitcoin trading timing figure, and 18(c) is allocation strategy of daily trading amount of gold and bitcoin.

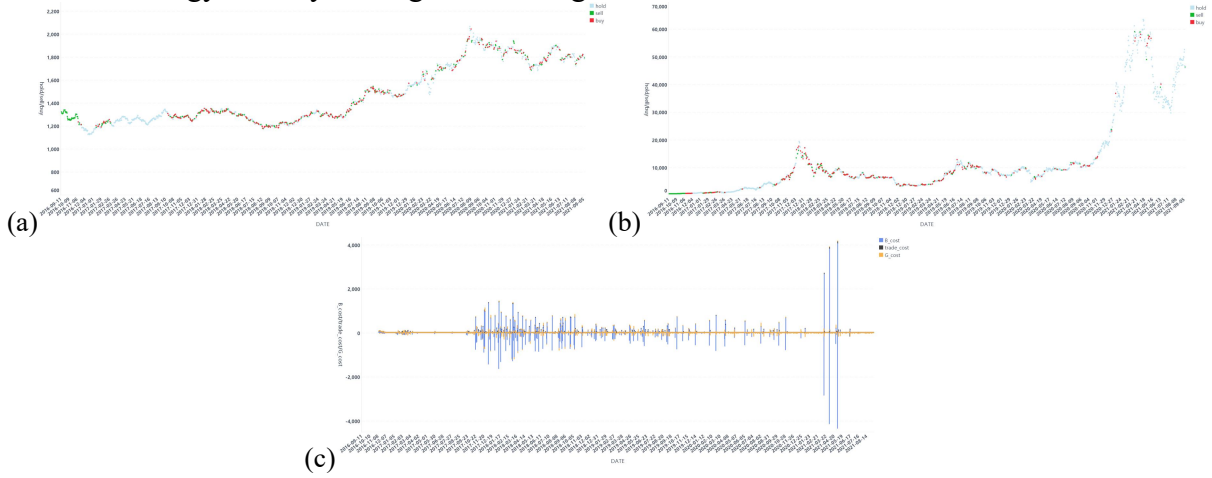


Figure 18: Diagram of Trading Strategy

When using this strategy, the changes in total assets are shown in figure 19(a). It can be seen that although there is a high return after five years, the assets have been in a low state for a long time from 2017/12 to 2018/12. and the peak of asset development is mainly in 2017/12 and 2021/04. According to the statistics in figure 19(b), it can be seen that annual interest rate in 2017 was the largest. In 2017, the first capital accumulation is carried out, and then in 2020, it seize the trend and quickly gain a huge income.

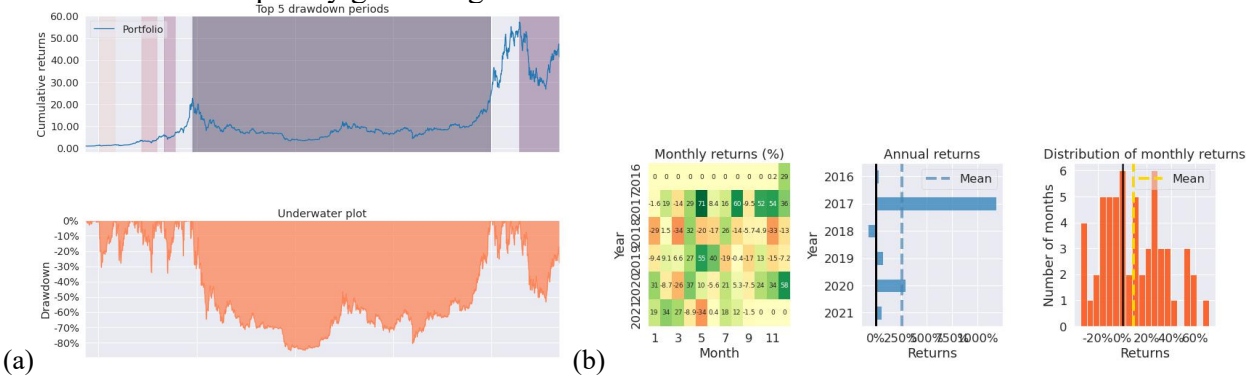


Figure 19: Changes in Total Assets

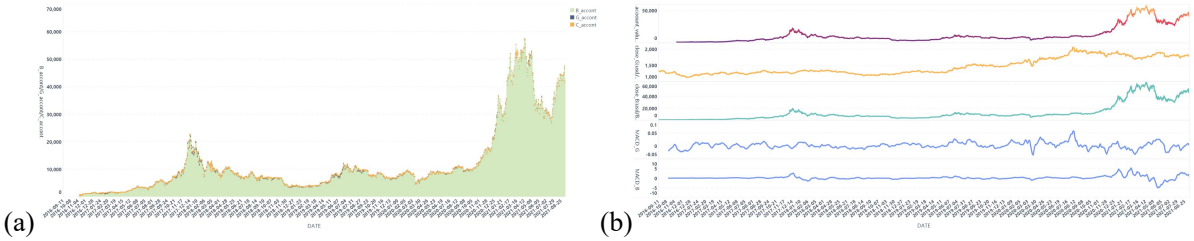
By observing the composition of daily assets (figure 20a) and the relationship between daily assets and prices and trends (figure 20b), it can be explained that the most important reason why

this algorithmic strategy can make returns is to quickly analyze the trend and capture the two small peaks of bitcoin's price rise. We discover that during the five-year period of trading, the bitcoin value accounted for the vast majority of assets, which is consistent with our previous analysis of gold and bitcoin volatility.

In the early stage, the gold price is high while the bitcoin price is low and trending up, so the strategy inclines to buy more bitcoin. This allows our strategy to capture the small peak of bitcoin price in 2017/12, earning a profit of 20,000*USD*.

In the mid-stage, the market is relatively sluggish, and the strategy tends to be conservative after updating, which makes the assets change little.

In the later stage, the prices of bitcoin and gold have already had a very large gap, and bitcoin has risen very fast. In the case of owning large risk tolerance, our algorithm will try to use bitcoin as the main benefit origin as much as possible. This allows our strategy to further capture the small peak of 2021/05, selling asseta at the top and completing the second major profit.



(a) (b)

Figure 20: Composition and Relationship of Daily Assets

7 Sensitivity Analysis

The hyperparameter transaction costs of gold and bitcoin (commission for each purchase and sale) may have an impact on our model. Through the idea of **controlling variables**, we explore the impact of changes in transaction costs on our model strategy. Considering the characteristics of real market, the commission will not exceed 3%, so the tests are carried out with gradients of 0.5%, 1% and 2% respectively. The results are as follows:

Table 4 shows the profit and Sharpe Ratio for the four parameter combinations. Figure 21 (in the same order as Table 4) shows the changes in assets under four strategies.

Table 4: Profit and Sharpe Ratio for the Four Parameter Combinations

Commission of Gold	Commission of Bitcoin	Total Return Rate	Annualized Rate of Return	Annual Volatility Rate	Sharpe Ratio	Annualized Rate of Change in Return
1%	2%	2286.173%	54.929%	62.201%	1.02	×
1%	1%	2616.789%	57.728%	62.961%	1.04	5.096%
2%	2%	9095.005%	86.631%	53.309%	1.44	54.715%
0.5%	2%	3046.907%	60.96%	60.078%	1.10	10.979%

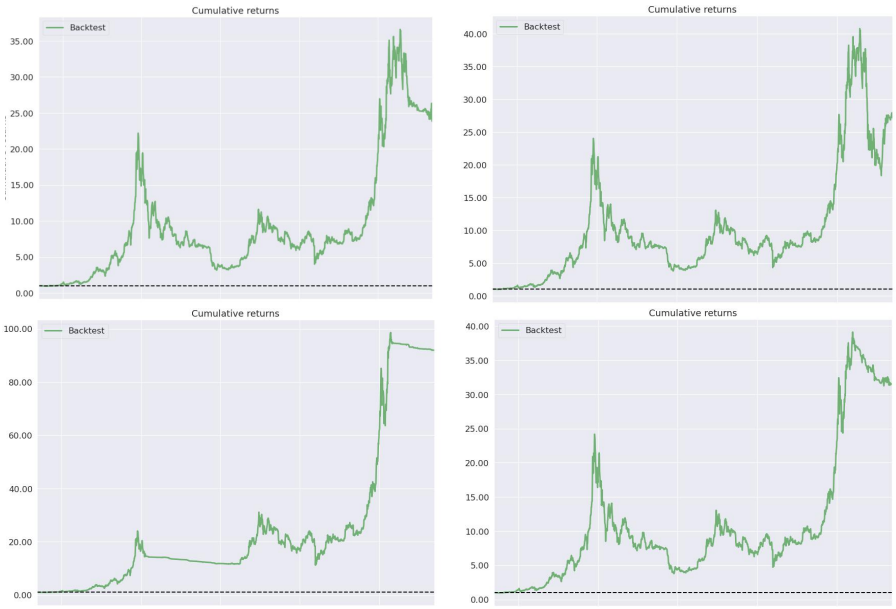


Figure 21: Changes in Assets under Four Strategies

Combining above table and figure, it can be noticed that although final annualized rate of return is different under different given commission combinations, the trends of different commissions are basically the same by observing changes in assets. This is because our model still cares most about price movements.

The combination of commissions will affect the proportion of gold and bitcoin in our allocation. As analyzed in section 6.3, the capital accumulation in the past five years is mainly due to price change of bitcoin. So it can be seen that when bitcoin commission is reduced (to 1%), the strategy will learn the advantages of bitcoin faster and thus have a better strategy for the five-year situation. But this change is not obvious.

On the contrary, gold commission will have a greater impact on strategy, and it is not a monotonous relationship. When gold commission increases (to 2%), the improvement of strategy is mainly reflected in decay period. This enables our strategy to smoothly survive the market downturn in 2017/12-2018/12 and 2021/05, so that capital will not lose too much, which in turn greatly affects the final income. When gold commission reduces (to 0.5%), the improvement of strategy is mainly reflected in the early stage. This allows our strategy to accumulate more capital in the early stage, thus catching up with the first peak of price volatility. However, because it is the early stage, the price is lower, so the improvement effect is not large.

8 Strengths and Weaknesses

8.1 Strengths

- Based on market efficiency theory and random walk theory, the price of futures is actually unpredictable, and the use of reinforcement learning framework avoids the irrational behavior of future price predictions.
- Compared to other algorithms that have to relearn every action, deep reinforcement learning reuses experience. Therefore, in every transaction, our model can make full use of past experience to make decisions, and each step is currently optimal.
- The core of reinforcement learning lies in the interaction between agent and environment, which is in line with the interaction between market and people in futures trading, so our model can describe futures trading more accurately.

- The improved DTSL-PPO algorithm can reasonably set the combination of futures configuration through risk control. The trading process can start quickly, and balance the long-term trend and short-term trend.
- Reinforcement learning can theoretically prove that when the sequence is large enough, the optimal strategy can be trained. Hence, our model can be applied to a larger trading system and has a good development space.

8.2 Weaknesses

- The calculation of algorithm is time-consuming.
- For daily purchases of stocks, the output of sliding window in practical applications should be daily actions. However, due to time constraints, the minimum output of sliding window in this paper's implementation strategy is one-week actions. According to the strategy, the final action trajectory and final reward are obtained. If the time cost is not considered, our model can find a better action trajectory.

9 Conclusion

According to the analysis of trading characteristics of futures market, we establish the DTSL-PPO algorithm based on deep reinforcement learning for trading problems of gold and bitcoin. The strategy obtained by our model using this algorithm can make each transaction the best choice for the moment, which is in accordance with the characteristics of futures trading behavior. What's more, it has the advantages of quick start, reasonable allocation ,and long-term and short-term trend balance, and has good performance in current data.

At the same time, for the trading problems of more futures portfolios, or the trading problems with more difficulty to predict fluctuation trends, or even the trading problems of large companies rather than individual units, the DTSL-PPO algorithm can solve with three major tools -- reasonable configuration combination settings, powerful neural network computing capabilities, and "interaction" abilities of reinforcement learning.

Therefore, the DTSL-PPO algorithm has a very broad development space.

References

[1] Fornari F, Mele A. SIGN-AND VOLATILITY-SWITCHING ARCH MODELS: THEORY AND APPLICATIONS TO INTERNATIONAL STOCK MARKETS[J]. Journal of Applied Econometrics, 1997, 12(1):49-65.

[2] Wang Bo. Empirical Analysis of Shanghai Stock Exchange Index Based on ARMA-GARCH Model[J]. Science, Technology and Engineering, 2012, 12(5):1219-1221.

[3] Tsantekidis A, Passalis N, Tefas A, et al. Forecasting Stock Prices from the Limit Order Book Using Convolutional Neural Networks[C]// IEEE Conference on Business Informatics. IEEE, 2017.

[4] Wu Wei, Chen Weiqiang, Liu Bo. Prediction of stock market fluctuations by BP neural network[J]. Journal of Dalian University of Technology, 2001, 41(1):8-18.

[5] Moody J, Saffell M. Learning to trade via direct reinforcement. IEEE Trans. on Neural Networks, 2001, 12(4):875-889.

[6] J. W. Lee, E. Hong, and J. Park, "A q-learning based approach to design of intelligent stock trading agents," in Engineering Management Conference, 2004 IEEE International,:1289-1292.

[7] Y. Deng, F. Bao, Y. Kong, Z. Ren, Q. Dai, Deep Direct Reinforcement Learning for Financial Signal Representation and Trading[J]. IEEE Transactions on Neural Networks and Learning Systems, 2017, 3,:653-664.

[8] Qi Yue, Huang Shuohua. Portfolio Management Based on Deep Reinforcement Learning DDPG Algorithm[J]. Computer and Modernization, 2018(05):93-99.

[9] Ye Wuyi, Sun Liping, Miao Boqi. Research on Dynamic Cointegration of Gold and Bitcoin: Based on Semiparametric MIDAS Quantile Regression Model[J]. Systems Science and Mathematics, 2020, 40(07): 1270-1285.
[10] Fama, Eugene F. "Random walks in stock market prices." Financial analysts journal 51.1 (1965):75-80.

MEMORANDUM

TO: All traders
FROM: Team #2205961
DATE: February 21, 2022
SUBJECT: A Deep-Reinforcement-Learning-Based Model for Trading Strategies

I. INTRODUCTION

Trading volatile assets has always been a hot and controversial topic of finance. Investors make effort to learn about the market, balance risks and benefits and find the best trading strategy in order to maximize their total return. Hitherto, several practical methods for market analysis have been wide in use. However, making right decisions at right moments all the time is nearly impossible due to the unpredictability of the market. Data scientists and financial experts are still exploring the mechanism behind.

Given the issue above, this memorandum is intended to inform you of a new deep-reinforcement-learning-based model for trading strategies originated from our team. This memorandum first discusses some common methods for strategy generation. The memorandum then shows the theoretical fundaments of our model. Finally, the memorandum shows the training process as well as the performance of our model.

II. METHODS FOR STRATEGY GENERATION

- **Fundamental Analysis**

Fundamental analysis is based on the traditional economics theory. It takes insight into the value of enterprises. By comparing the evaluation with stock prices, professionals give trading strategy accordingly. However, this method is often too stubborn and rigid to fit the flexible market, and it neglects many details in data.

- **Technical Analysis by Human Experts**

Technical analysis is based on the traditional securities theory. It mainly aims to predict the future prices given the history of prices and technical indicators. Usually, the suggestions are given by professionals. However, computers perform better than human in aspects such as comprehending hidden logic and learning from data.

- **Technical Analysis by Machines**

Recent development of machine learning and deep learning allows computers to fully utilize data and understand the mechanism behind without sticking to human paradigms. There are two mainstream ideas of deploying deep learning: The first idea is about using deep-learning-based models to predict future prices, then plan for a strategy in a traditional way. The second idea is about periodically training deep-reinforcement-learning-based model to interact with the past market environment in order to let the machine learn the rule behind the market. In our opinion, the second way is better because the market is already proved to be highly unpredictable by Fama [10], whereas some potential patterns may still be unveiled and can be discovered by machines.

III. THEORETICAL FUNDAMENTS

The process of the market updating daily prices of securities accords with Markov Decision Process. It means that every present decision is only based on the history of previous n timesteps.

It features temporality of states (prices and technical indicators, as well as their order) and dynamic optimization (observations and decisions in turns).
Reinforcement learning is based on Markov Decision Process theory and it allows machine agent to interact with past environment thus learn the potentials. At the beginning, the agent will be acting randomly so it will receive either positive or negative feedback. Then, according to the feedback, the agent renews its policy in order to maximize the total reward from the environment, which means that the probability distribution for its future actions will change to perform better. Plus, the policy function is presented by a deep-learning-based neural network. (That is how 'Deep-reinforcement-learning' comes.) Iterating in this way, the agent soon learns the best strategy to maximize cumulative reward, which is often designed as the total return.
An example of Markov Decision Process in reinforcement learning is shown in **Fig. 1**.

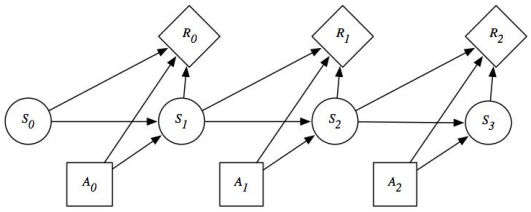


Fig.1 A simple example of Markov Decision Process in reinforcement learning

IV. DTSL-PPO: An Optimal Futures Market Trading Strategy

Among many state-of-art deep-reinforcement-learning-based models, we chose the more stable PPO referring to existing experiences. Considering of the famous trade-off of exploration and exploitation in reinforcement learning, as well as shortening the no-trades-data-collecting period (enough data must be gathered for training before the agent starts to handle trades), we came up with a unique training strategy called 'dynamic T-period sliding window', which allows the model to start shortly and maximize short-term reward, then iterate later when more data is collected thus it plans long term. We then use an ensemble of agents trained respectively from short and long (S-L) data window in order to make our model aware of recent fluctuations while being conscious of long-term trend. We finally propose Dynamic T-Period Sliding Window Deep Reinforcement Learning Algorithm PPO Combining Short-Term and Long-Term Strategies (DTSL-PPO).
After iterative updates and operations, our model has a satisfying performance. The account value raise from **$1,000.00** on Sep/11/2016 to **$416538.80** on Sep/10/2021. The reward rate reaches **4065.38%** and Sharpe Ratio reaches **1.14**, and the trades generated by the agent is timely and reasonable. Furthermore, we tested the sensitivity of our model towards the fluctuation of transaction fee and the result was acceptable.
Some of the performance are shown in **Fig. 2**.

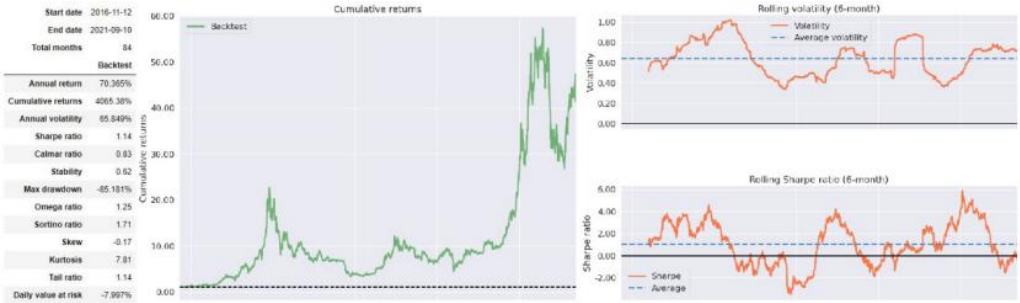


Fig.2 Performance of our DTSL-PPO model