2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)

A DDPG Algorithm for Portfolio Management

Fang Lin*, Meiqing Wang
College of Mathematics and Computer Science
Fuzhou University
Fuzhou, China
email: fzu_lf@163.com; mqwang@fzu.edu.cn

Rong Liu, Qianying Hong
College of Mathematics and Computer Science
Fuzhou University
Fuzhou, China
email: liu_r@fzu.edu.cn; qianying.hong@qq.com

***Abstract*—The purpose of portfolio management is to select a variety of financial products to form a portfolio and then manage these portfolios to achieve the purpose of diversifying risk and improving efficiency. In this paper, the Deep Deterministic Policy Gradient (DDPG) algorithm with neural networks is used, new states, actions and reward functions are proposed. The empirical analysis shows that this paper's method performs better than the method of investing with Q-learning algorithm, equally-weighted method, investing all funds in risk-free assets, or investing all funds in stocks.**

Keywords: deep reinforcement learning; DDPG algorithm; portfolio management

I. INTRODUCTION

A portfolio is a collection of stocks, bonds, financial derivatives, etc. held by investors or financial institutions. The purpose of portfolio management is to diversify the investment risk and obtain maximum benefits. The mean-variance model proposed by Markowitz [10] is the cornerstone of modern portfolio theory. De Miguel et al. [1] argued that most of the strategies based sample performed worse than the equally-weighted portfolio approach (Funds invested in risky assets and risk-free assets are equally weighted) in out of sample. For this reason, many researchers put forward different improvement schemes. For example, Kourtis [4] proposed a general method to improve the stability of the mean-variance model and the Sharpe ratio.

With the development of machine learning, deep learning (DL) and reinforcement learning (RL) combine to deep reinforcement learning (DRL). In 2015, DeepMind researchers published a paper [9] in the journal Nature. In the paper, the authors proposed the Deep Q-Network method (DQN), which successfully achieved or exceeded the human level in playing Atari games. DQN is a kind of the DRL method. Since then, the DRL method has quickly become the focus of the artificial intelligence community, and widely used in various fields: video games [9], board games [11-12], complex mechanical operation [8], etc.

The successful research of deep reinforcement learning in these fields has attracted a large number of financial researchers. They can't help but ask the question: Can these technologies be applied to the field of financial investment? Jiang et al. [2] proposed a financial-model-free reinforcement learning framework to the portfolio management problem. Li et al. [7] studied the stock market investment strategy of the deep reinforcement learning model, and verified the validity of the model through empirical data. Jin and Hamza [3] used deep Q learning in managing the portfolio of two stocks. Zhu etc. [13] used the Q-learning algorithm in which the relative change of the opening and closing price is used to form the state space and the state value function iterative method based on the Monte Carlo algorithm is proposed. The method has a better performance than the equally-weighted method, however, the Q-table constructed by the Q-learning method is discrete and limited and cannot solve the problem of too high dimension of state spaces or action spaces.

In this paper, the deep deterministic policy gradient algorithm with neural network is used to deal with high-dimensional problems. In this method, the continuous state space and the continuous action space are used to replace the discrete state space and the discrete action space in Ref [13]. And the reward function is improved so that it is easier to increase the action range of high returns and reduce the range of poor returns. Empirical analysis show that, this paper's method performs better than the Q-learning method in Ref [13], the method of equally-weighted, the method of investing all funds in bonds and the method investing all funds in stock .

II. REINFORCEMENT LEARNING FRAMEWORK

A. Reinforcement Learning Decision Process

Reinforcement learning is used to solve decision problems. That is, an agent learns by "trial and error", performing actions in the environment for maximum cumulative returns. The main decision-making process is as follows: at each moment t , the agent receives the state from the environment, then adjusts the strategy according to the state, executes an action and feeds back to the environment. The environment gives the agent a reward and transfers to the next state according to the action performed by the agent. After such a series of interactions between the agent and the environment, an interaction sequence is obtained, that is:

$$\tau = \langle s_0, a_0, r_1, s_1 \cdots, s_{T-1}, a_{T-1}, r_T, s_T \rangle \quad (1)$$

B. Policy

The policy of reinforcement learning can be abstracted as a state-to-action mapping, denoted as: $\pi : S \to A$, which can be classified as the deterministic policy and the random policy.

The deterministic policy defines a unique action to be performed in each state, namely,

$$a = \pi(s) \quad (2)$$

For the random policy, a state may be mapped into several actions, the policy function is described as the probability of the agent taking the action a in the state s :

$$\pi(a \mid s) = P(a_t = a \mid s_t = s) \quad (3)$$

A trajectory is generated when an agent performs a series of actions according to policy π at time t under the current state s_t .

$$\tau = \langle s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \cdots \rangle \quad (4)$$

The cumulative return represents the sum of the rewards received by the agent according to the trajectory which is defined as:

2473-3636/20/$31.00 ©2020 IEEE
DOI 10.1109/DCABES50732.2020.00065

$$G_t = r_t + \gamma r_{t+1} + \cdots = \sum_{k=1}^{\infty} \gamma^k r_{t+k} \quad (5)$$

Where, $\gamma \in [0,1]$ is the discount factor. It's means that the further away from the current state, the smaller the impact.

C. Value Function

The state value function represents the expectation of the cumulative return at state $s \in S$ which is defined as,

$$V_\pi(s) = E_\pi[G_t \mid S_t = s] = E_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid S_t = s]$$
$$= E_\pi[r_{t+1} + \gamma V_\pi(s_{t+1}) \mid S_t = s] \quad (6)$$

Where E is expectation.

The state-action value function $Q_\pi(s,a)$ represents the expectation of the cumulative return in the current state $s \in S$ executing the action $a \in A$ which is defined as,

$$Q_\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a] = r^a_{ss'} + \gamma \sum_{s' \in S} P_{ss'} V_\pi(s') \quad (7)$$

Where, s' represents the next state. $r_{ss'} = E[r_{t+1} \mid S_t = s, A_t = a]$ represents the reward for executing an action in a state. $P_{ss'} = E_\pi[S_{t+1} = s' \mid S_t = s, A_t = a]$ is the state transition probability.

This means that the state value function is the expectation of all state-action value functions based on policy π. It can be defined as,

$$V_\pi(s) = \sum_{a \in A} \pi(a \mid s) Q_\pi(s,a) \quad (8)$$

D. Deep Reinforcement Learning

The traditional reinforcement learning algorithm, such as Q-learning algorithm which cannot maintain a Q-table with too high dimension, can't solve the problem of too high dimension of state spaces or action spaces. To solve this problem, deep reinforcement learning uses value function approximation instead of Q-table.

Let $\hat{V}(s,\theta)$ represent the approximate state value function where θ is a parameter:

$$\hat{V}(s,\theta) \approx V_\pi(s) \quad (9)$$

Let $\hat{Q}(s,a,\theta)$ represent the approximate state-action value function described by parameters θ:

$$\hat{Q}(s,a,\theta) \approx Q_\pi(s,a) \quad (10)$$

There are many methods to approximate the value function, such as the linear representation, the decision tree, the nearest neighbor method, the neural network and so on. In this paper, the neural network is used to approximate the value function.

III. DYNAMIC PORTFOLIO BASED ON DEEP REINFORCEMENT LEARNING

The dynamic portfolio problem maximizes returns by constantly changing the proportion of funds invested in financial assets. Deep reinforcement learning builds the learning framework through neural network and adjusts the parameters of the network to get the optimal investment weight, so as to achieve the purpose of dynamic portfolio optimization.

A. Problem Description

Consider the simplest portfolio problem. It is assumed that the investor will invest a risky asset such as a stock S, and a risk-free asset f. The weight of funds invested in stock is ω_s and the weight of funds invested in risk-free assets is ω_f. The rate of return of the stock is r_s and the rate of return of the risk-free asset is r_f. The purpose of investors is to dynamically adjust the proportion of funds to maximize the return at the terminal time T. Namely,

$$r_t = \omega_{st} \cdot r_{st} + \omega_{ft} \cdot r_{ft}, \; R_T = \max_{\omega_{st} \in [0,1]} \{\prod_{t=1}^{T} r_0(1+r_t)\} \quad (11)$$

Where, r_0 is the initial investment funds. ω_{st} is the weight of funds invested in the stock at time t, ω_{ft} is the weight of funds invested in the risk-free asset at time t, and $\omega_{st} + \omega_{ft} = 1$.

B. Q-Learning Method

1) State

In previous studies, researchers usually normalized the closing price of stocks as state inputs. Ref [13] used the daily relative change of stock price to form the state space. In which a state consists of two parts. The first part is the daily relative change of the closing price of the stock, the second part is the daily relative change of the opening price of the stock, and the information of three trading days is used,

$$s_t = [\tilde{r}_{Sc}(t-3), \tilde{r}_{So}(t-2), \tilde{r}_{Sc}(t-2), \tilde{r}_{So}(t-1), \tilde{r}_{Sc}(t-1), \tilde{r}_{So}(t)], t \geq 3 \quad (12)$$

The value of each component is computed as follows ,

$$\forall t, \tilde{r}_{Sc}(t) = \begin{cases} -1 & r_{Sc}(t-1) \leq med1 \\ 1 & r_{Sc}(t-1) > med1 \end{cases}$$
$$\tilde{r}_{So}(t) = \begin{cases} -1 & r_{So}(t) \leq med2 \\ 1 & r_{So}(t) > med2 \end{cases} \quad (13)$$

Where,

$$r_{Sc}(t) = \frac{pc_t - pc_{t-1}}{pc_{t-1}}, \; r_{So}(t) = \frac{po_t - pc_{t-1}}{pc_{t-1}} \quad (14)$$

$r_{Sc}(t)$ represents the daily relative change of the closing price pc_t of stock S at time t compared to the closing price pc_{t-1} of time $t-1$. $r_{So}(t)$ represents the daily relative change of the opening price po_t of stock S at time t compared to the closing price pc_{t-1} of $t-1$. $med1$ represents the median of $r_{Sc}(t)$ in the training set; $med2$ is the median of $r_{So}(t)$ in the training set.

2) Action

Action a_t is defined as the proportion of funds invested in the stock market, which is discretized into $[0, 0.2, 0.4, 0.6, 0.8, 1]$.

3) Reward Function

The reward function is defined as follows,

$$r_t = a_t \cdot r_{Sc}(t+1) + (1 - a_t) \cdot r_f \quad (15)$$

C. State, Action and Reward Function in DDPG

The DDPG algorithm is composed of four networks, the actor current network, actor target network, critic current network and critic target network. And the DDPG algorithm is a deterministic policy. This means that our actor network no longer outputs the probability of each action, but a concrete action, which is more conducive to our learning in the continuous action space. Due to the characteristics of DDPG, the state, action and reward function are proposed as follows.

1) State

Because Q-learning algorithm is based on Q-table to update value function, it can only deal with Q-table with finite dimensional states and actions. The state space defined in Ref [13] is easy to make the state limited. That is to say, when using the Eqn.(12) to describe a state , it is

possible that due to the limitations of observation or modeling, the two different states in the real environment will have the same feature after modeling, which leads to the insufficient ability of the method to deal with problems. In this paper, the state is defined as follows,

$$s_t = [r_{Sc}(t-3), r_{So}(t-2), r_{Sc}(t-2), r_{So}(t-1), r_{Sc}(t-1), r_{So}(t)], t \geq 3 \quad (16)$$

2) Action

In DDPG algorithm, we can deal with the problem that the action is a continuous value. In this paper, the action a_t is defined as the proportion of funds invested in the stock market. and the continuous interval of the action is supposed as $a_t \in [0,1]$. In order to avoid the action too small to buy enough stocks, set $a_t = 0$ when $a_t < 0.01$.

3) Reward Function

Since the setting of the reward function will affect the selection of actions, the actions are selected in the actor network according to the feedback of critic network. When the reward increases, the range of action behavior increases, which makes it easier to be selected. This paper chooses 2 and 5 to enlarge (shrink) the reward function. The reward function is modified as follows,

$$r_t = a_t \cdot r_{Sc}(t+1) + (1-a_t) \cdot r_f,$$

$$r_t = \begin{cases} 5 \cdot r_t & if \quad |r_t| > 0.05 \\ 2 \cdot r_t & else \end{cases} \quad (17)$$

IV. Algorithm Implementation

A. Data Preprocessing

Ref[13] selected 2% annual return bonds, Jinfeng Wine and CITIC Securities stocks for empirical analysis. The training period is from September 1, 2006 to August 31, 2010, and the test period is from September 1, 2010 to August 29, 2014. In order to compare with their methods, in this paper the same stocks and data are used. The data is based on the closing and opening prices of stocks for each trading day downloaded by the Tushare financial community.

B. DDPG Algorithm

The following assumptions are made for the model.

1. Every transaction can be calculated at the closing price.
2. The funds invested in the transaction are small and will not affect the stock market.
3. Transaction costs are not considered.
4. All the funds invested in the stock market can be converted into the stock.

DDPG Algorithm

Randomly initialize critic network $Q(s,a|w)$ and actor $\mu(s|\theta)$ with weights w, θ .
Initialize target network Q', μ' with weights w', θ'.
Initialize replay buffer Rand a random process δ .
for episode=1, M do
  Initialize state s_0.
  for d=1,T do
    Select action $a_t = \pi_\theta(s_t) + \delta$ according the current policy
    Execute action a_t and receive reward r_t and observe new next state s_{t+1}.
    Store transition (s_t, a_t, r_t, s_{t+1}) in R.
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R.
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \pi_{\theta'}(s_{i+1}) + \delta; w')$.
    Update critic by minimizing the loss:

$$L = \frac{1}{N} \sum_i (y_i - Q(s_i, a_i; w))^2$$

    Update the actor policy using the sample policy gradient:

$$\nabla_\theta J \approx \frac{1}{N} \sum_i [\nabla_a Q(s_i, a_i; w)|_{s=s_i, a=\pi_\theta(s)} \nabla_\theta \pi_\theta(s)|_{s=s_i}]$$

    Update the target networks:

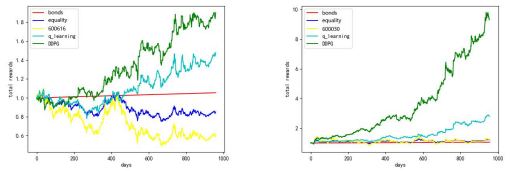
$$w' \leftarrow \tau w + (1-\tau) w', \quad \theta' \leftarrow \tau \theta + (1-\tau) \theta'$$

  end for
end for

V. Empirical Analysis

De Miguel et al. [1] argued that most of the strategies based sample performed worse than the equally-weighted portfolio approach (investing in risky assets and risk-free assets with equal funds weights) in out of sample. Therefore, this paper method (DDPG) compare with the method of equally-weighted, Q-learning method, the method of investing all funds in bonds and the method of investing all funds in stock .

Assuming that the initial investment funds is 1, the following figure shows the total return obtained by using different investment methods for the combination of bonds and Jinfeng Wine (stock code: 600616) and the combination of bonds and CITIC Securities (stock code: 600030).



(a) bonds and Jinfeng Wine (b) bonds and CITIC Securities

Figure 1. Relative total returns during the test

The indicators of different investment methods during the test period are shown in the table 1.

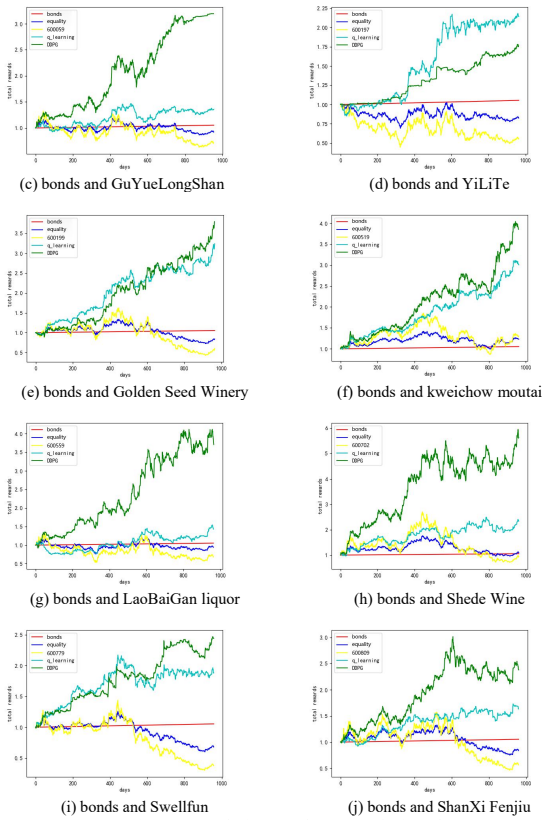
TABLE I. INDICATORS OF VARIOUS INVESTMENT METHODS

	Annual return	**Annual volatility**	**Sharpe ratio**
bonds	2%		
600616	-11.56%	34.29%	-39.55%
Equality	-3.5%	17.28%	-31.83%
Q-learning	7.59%	8.85%	63.16%
DDPG	17.05%	20.63%	72.96%
600030	4.42%	35.29%	6.86%
Equality	5.09%	17.89%	17.27%
Q-learning	23.02%	8.29%	253.56%
DDPG	74.63%	28.56%	254.30%

It can be seen from the above chart that Jinfeng Wine and CITIC Securities have a downward trend in the whole test period. Even so, the overall return of Q-learning method and DDPG method in this paper are higher than that of bonds and equally-weighted method, but DDPG method is better than Q-learning method in both annual return and Sharpe ratio.

A. The Universality of Model

In order to illustrate the universality of the model, this paper additionally selects 8 wine stocks of Shanghai stock A for comparison. The sample selection of training set and test set is the same as the above experiment. The results are shown in Figure 2 and the various indicators of each investment method are calculated as shown in Table 2.



(c) bonds and GuYueLongShan (d) bonds and YiLiTe

(e) bonds and Golden Seed Winery (f) bonds and kweichow moutai

(g) bonds and LaoBaiGan liquor (h) bonds and Shede Wine

(i) bonds and Swellfun (j) bonds and ShanXi Fenjiu

Figure 2. Relative total returns during the test

TABLE II. INDICATORS OF VARIOUS INVESTMENT METHODS

	Annual return	Annual volatility	Sharpe ratio
bonds	2%		
600059	-5.47%	34.74%	-21.50%
Equality	-2.08%	17.43%	-23.38%
Q-learning	9.54%	19.85%	37.98%
DDPG	33.64%	22.44%	140.99%
600197	-11.34%	22.27%	-59.90%
Equality	-4.68%	19.07%	-35.02%
Q-learning	20.03%	22.28%	80.92%
DDPG	15.15%	9.21%	142.78%
600199	-13.16%	39.49%	-38.39%
Equality	-4.26%	19.84%	-31.54%
Q-learning	28.80%	21.57%	124.25%
DDPG	39.63%	25.67%	146.59%
600519	5.01%	29.72%	10.13%
Equality	5.06%	14.97%	20.42%
Q-learning	34.73%	17.04%	192.08%
DDPG	40.12%	21.79%	174.94%
600559	-8.41%	42.72%	-24.37%
Equality	-0.78%	21.64%	-12.85%
Q-learning	9.25%	25.17%	28.80%
DDPG	38.77%	30.00%	122.58%
600702	-4.25%	44.46%	-14.06%
Equality	2.18%	22.51%	0.80%
Q-learning	19.58%	23.27%	75.55%
DDPG	54.03%	33.40%	155.78%
600779	-21.46%	35.52%	-66.05%
Equality	-9.21%	17.96%	-62.42%
Q-learning	14.06%	20.28%	59.47%
DDPG	25.00%	16.34%	140.78%
600809	-11.08%	39.17%	-33.39%
Equality	-4.41%	19.66%	-32.60%
Q-learning	11.45%	18.97%	49.82%
DDPG	24.23%	23.92%	92.93%

As can be seen from Figure 2 and Table 2, for the stock 600197, the average annual return of the DDPG method is less than the Q-learning method, but the annual volatility is lower than the Q-learning method and the Sharpe ratio is higher than the Q-learning method. For stock 600519, the annual return of the DDPG method is higher than the Q-learning method, but the Sharpe ratio is lower than the Q-learning method. In general, DDPG method performs better than the methods of investing with Q-learning algorithm, equally-weighted method, investing all funds in risk-free assets, or investing all funds in stocks.

VI. CONCLUSION

In this paper, DDPG algorithm is used to study simple portfolio problem and to solve the problem of limited state defined by Q-learning algorithm. In this paper, the new states, actions and reward functions are proposed. Empirical analysis show that, the proposed method performs better than the Q-learning method in Ref [13], the method of equally-weighted, the method of investing all funds in bonds and the method investing all funds in stock. And shows high returns and high Sharp ratio when the stock price is down.

REFERENCES

[1] DeMiguel, V., L. Garlappi, and R. Uppal, 2009, Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy, Review of Financial Studies 22, 1915 – 1953.

[2] Jiang Z , Xu D , Liang J . A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem[J]. Papers, 2017.

[3] Jin, Olivier, and Hamza El-Saawy. "Portfolio management using reinforcement learning." Stanford University (2016).

[4] Kourtis A. A Stability Approach to Mean-Variance Optimization[J]. The Financial Review (Statesboro), 2015, 50(3):301-330.

[5] Kanwar, Nitin. Deep Reinforcement Learning-Based Portfolio Management. Diss. 2019.

[6] Ledoit, O., and M. Wolf, 2004, Honey, I shrunk the sample covariance matrix: Problems in mean-variance optimization, Journal of Portfolio Management 30, 110 – 119.

[7] Li Y , Nee M , Chang V . An Empirical Research on the Investment Strategy of Stock Market based on Deep Reinforcement Learning model[C]// 4th International Conference on Complexity, Future Information Systems and Risk (COMPLEXIS 2019). 2019.

[8] Levine, Sergey, et al. "End-to-end training of deep visuomotor policies." The Journal of Machine Learning Research 17.1 (2016): 1334-1373.

[9] Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." Nature 518.7540 (2015): 529-533.

[10] Markowitz, H., Portfolio selection [J]. The Journal of Finance, 1952, 7(1):77–91.

[11] Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." nature 529.7587 (2016): 484.

[12] Silver, David, et al. "Mastering the game of go without human knowledge." Nature 550.7676 (2017): 354-359.

[13] ZHU Kun, LIU Rong, WANG Meiqing. Reinforcement learning state and value function selection for portfolio optimization[J]. Journal of FZU (Natural Science),2020,48(02):146-15.