Neurocomputing 402 (2020) 171–182

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Portfolio trading system of digital currencies: A deep reinforcement learning with multidimensional attention gating mechanism

Liguo Weng[a], Xudong Sun[a], Min Xia[a,*], Jia Liu[a], Yiqing Xu[b]

[a] Jiangsu Key Laboratory of Big Data Analysis Technology, Nanjing University of Information Science and Technology, Nanjing, 210044, China
[b] College of Information Science and Technology, Nanjing Forestry University, Nanjing, 210037, China

ARTICLE INFO

Article history:
Received 23 August 2019
Revised 15 February 2020
Accepted 1 April 2020
Available online 11 April 2020

Communicated by Dr. Peter. W Vamplew

Keywords:
Portfolio
Deep-reinforcement learning
Reinforcement learning
Attention gating mechanism

ABSTRACT

As a hot topic in the financial engineering, the portfolio optimization aims to increase investors' wealth. In this paper, a portfolio management system based on deep-reinforcement learning is proposed. In contrast to inflexible traditional methods, the proposed system achieves a better trading strategy through Reinforcement learning. The reward signal of Reinforcement learning is updated by action weights from Deep learning networks. Low price, high price and close price constitute the inputs, but the importance of these three features is quite different. Traditional methods and the classical CNN can't deal with these three features separately, but in our method, a designed depth convolution is proposed to deal with these three features separately. In a virtual currency market, the price rise only occurs in a flash. Traditional methods and CNN networks can't accurately judge the critical time. In order to solve this problem, a three-dimensional attention gating network is proposed and it gives higher weights on rising moments and assets. Under different market conditions, the proposed system achieves more substantial returns and greatly improves the Sharpe ratios. The short-term risk index of the proposed system is lower than those of the traditional algorithms. Simulation results show that the traditional algorithms (including Best, CRP, PAMR, CWMR and CNN) are unable to perform as well as our approach.

1. Introduction

In financial engineering, portfolio optimization is a practical task. As an effective approach to investment, it restrains an investor's reckless investment behaviors to a certain extent. The purpose of portfolio optimization is to allocate funds in a group of assests in order to maximize returns. The research on portfolio can be divided into two categories: Mean–Variance-based approaches and Capital-based approaches [1]. The original Mean–Variance model which aimed to minimize the risk (variance) for the desired return was a single-objective model [2]. When a portfolio was constructed by the original Mean–Variance model, calculating the mean and the covariance of assets was involved in the process. If the sample size was too small, the mean and covariance of these samples would be highly uncertain. However, if the sample size was too large, computing the mean and covariance would consume a lot of time. Another disadvantage of the original Mean–Variance model was that when the final return was either too high or too low, the variance of the model would be increased, as a result, investors might abandon the strategy with the highest return. This action was against the original intention of portfolio optimization. Because of the difficulties that the Mean–Variance portfolio optimization encountered in the real world, multi-objective approaches were proposed. It may not be possible to use multi-objective models to optimize all objective functions. Therefore, it is necessary to give priority to some objectives over others by using methods such as weighted sum method or Pareto-based approaches. The weighted sum approach is the most popular approach for Mean–Variance portfolio optimization. In the weighted sum methods, a set of objectives were combined into a single objective. However, in spite of its simplicity, there was difficulty in obtaining Pareto optimal solutions by this approach because multi-objective optimization problems had a non-convex Pareto-optimal front [3]. Therefore, the main disadvantage of weighted sum approach is that it is unable to generate all Pareto-optimal solutions with non-convex trade-off surfaces. Pareto-based approaches are able to handle large search spaces and multiple alternative trade-offs. However, there is no single criterion to assess the quality of a trade-off front and the quality measurements are difficult to define [3] The Capital-Growth-based methods paid more attention to solve multi-stage or sequential portfolio problems. Its purpose was to maximize the expected growth

* Corresponding author.
E-mail address: xiamin@nuist.edu.cn (M. Xia).

https://doi.org/10.1016/j.neucom.2020.04.004
0925-2312/

rate or logarithmic return of the portfolio. Capital-Growth-based methods still had some limitations, which made it difficult for them to achieve good results in some special market situations. Methods like Passive Aggressive Mean Reversion (PAMR) [4] and Confidence weighted mean reversion (CWMR) [5] relied on the assumption that mean reversion existed in a portfolio pool, that is, buying worse performed stocks was profitable. Moreover, referring to the PAMR, Li et al. only justified that the algorithm worked with relatively low transaction costs, but the performance of the PAMR may decline when transaction costs increase.

Digital currency has become an innovative investment option and a credible investment vehicle. The remarkable features of digital currency are its decentralization and openness. Digital currency is a decentralized, peer-to-peer network that allows for ownership transferring without a central regulating party. Decentralization allows the digital currency to have better security and faster settlement. Compared to other financial markets, the digital currency market is more conducive in learning the market's own behaviors. Openness means that this market is more accessible and that data from this market is richer. Exchanges in digital currency market are carried out on the Internet, and hence the market is open all day long. These non-stop markets are ideal for our system to learn in shorter time-frames. However, most existing algorithms to digital currency market were limited to the study of its prices using Bayesian neural networks [6], long short-term memory neural networks [7] and so on. These studies just predicted the price of a digital currency, but did not further implement portfolio management. In addition, the above studies only focused on a single digital currency instead of multiple assets. Alessandretti et al. built investment portfolios based on the predictions [8]. However, authors stressed that they did not consider any transaction fees which apparently could affect the profits in real markets. Besides, the algorithm proposed by Alessandretti first predicted the price of digital currencies and then built portfolios based on the predictions. The goal of portfolio is to maximize the investment returns and therefore it is different from prediction of asset prices. The performance of this work highly depends on prediction accuracies, but it is difficult for a network to predict prices in the digital currency market.

The Deep-reinforcement learning (DRL) has gradually emerged in applications of video games, chess games and traffic management since 2015 [9]. This is mainly because the DRL can learn independently according to the rules provided. The application of reinforcement learning(RL) in portfolio management began in 2001, Dirk presented a kernel-based reinforcement learning method to overcome the disadvantage of instability in reinforcement learning. They focused on learning in an average-cost framework and on a practical application to the optimal portfolio choice problem [10]. Jiang et al. conducted portfolio using a combination of convolutional neural networks(CNN) and reinforcement learning in [11]. Saud et al. proposed a recurrent reinforcement learning method with an adjusted objective function [12]. In 2019, Saud et al. extended their recurrent reinforcement portfolio with a combination of the recurrent reinforcement learning and a particle swarm algorithm with Calmar ratio [13]. In recent years, the Deep-reinforcement learning(DRL) has paved another way for portfolio management in the virtual currency market. The DRL is an algorithm that integrates the powerful feature representation of Deep learning [14,15] and the efficient strategy search of Reinforcement learning [16,17]. The DRL is not a simple superposition of the Deep learning and the Reinforcement learning, but a fusion of the two algorithms. Jiang et al. tried to manage portfolio in the virtual currency market with deep-reinforcement learning [11]. However, the virtual currency market is not stable and price is always volatile. The deep networks proposed by Jiang were only a simple superposition of convolution kernels. Their proposed method could only extract shallow data information but ignored potential relationships between assets.

In order to further solve the above issues, this paper proposed a network based on the DRL in the virtual currency market. The main contributions of this paper were:

(i) Firstly, the XGBoost is applied to quantify the importance of historical data features and then our system takes the three most important features as inputs. The existing research on virtual currency tended to use all historical data features as inputs including open price, close price, high price, low price and volume [6,8]. In [11] Jiang et al. directly used close price, high price and low price as inputs without theoretical explanation. Our paper use the XGBoost to prove the rationality of choosing close price, high price and low price as inputs.

(ii) Secondly, a novel separable convolution and three-dimensional attention gating networks are proposed to extract historical data features. The input is three-dimensional and respectively represents the number of assets, the length of time series and the corresponding features(features stand for close price, high price and low price). The price changes of the same asset in different periods have certain rules to follow, and the growth rate of different assets at the same time will have some potential relationships. Obviously, the two relationships mentioned above are different concepts. Classical Convolutional Neural Networks(CNN) are not able to dependently deal with information from all channels and the original depth convolutions Xception [18] only worked on a single channel. Therefore, a novel separable convolution is proposed in this paper to handle this issue. In addition, our proposed three-dimensional attention gating networks also act on all three dimensions, whereas other attention networks usually act on one dimension only(like [19]).

(iii) Thirdly, our proposed system is an end-to-end model which directly outputs the final trading action without predicting asset prices. As shown in [8], they established investment portfolios based on the predictions of virtual currency prices. It is clear that building portfolios after prediction is not an efficient method. Moreover, many existing attempts to create portfolios are based on limited assets [12,13,20]. In this paper, we build portfolios using a pool of 20 assets.

2. Methods

2.1. Selection of features

The data used in this paper is from Poloniex Trading Platform. Poloniex founded in 2014 is one of the world's leading virtual currency exchanges. Virtual currency market has the advantages of high transparency, low inflation and convenient trading. The existence of virtual currency market makes it easy for the public to obtain historical data, even minute-level historical data.

The environment in Reinforcement learning is the virtual currency data, which is composed of close price, high price, low price, open price and volume. The historical data obviously contains a lot of redundant information. If we study this useless information, we would waste a lot of computation, failing to meet the original intention of this method. The first task of this paper is to pick out the most important features from historical data.

This paper uses XGBoost to evaluate the importance of all features [21]. The inputs are close price, high price, low price, open price and volume and the outputs are importance scores of the five features. Hence input data X consists of x_1, x_2, $\cdots$, x_5 :

$$X = (x_1, x_2, \cdots, x_5). \tag{1}$$

Table 1
Feature importance scores in each currency.

currency	close(%)	high(%)	low(%)	open(%)	volume(%)
xmr	66.4	10.9	17.4	2.5	2.8
eth	66.1	9.5	12.6	5.9	5.9
zec	39.8	24.7	15.8	10.5	9.2
fct	63.9	9.5	18.9	7.7	0
rusdt	83.3	2.5	9.2	5.0	0
etc	66.6	12.3	19.3	1.8	0
rep	62.8	15.9	17.1	2.5	1.7
xrp	100	0	0	0	0
dash	52.7	18.6	21.0	7.0	0.7
maid	86.7	2.5	10.0	0.8	0
ste	82.8	3.1	12.5	0	1.6
lsk	88.6	5.7	5.7	0	0
ltc	59.4	16.0	14.4	9.8	0.4
gam	100	0	0	0	0
xem	83.7	7.0	7.0	2.3	0
nav	100	0	0	0	0
bts	100	0	0	0	0
dog	42.7	22.3	20.9	9.7	4.4
sc	100	0	0	0	0
xcp	76.3	23.7	0	0	0

The core of XGBoost is to add decision trees continuously. The output score $\hat{y}$ is the sum of K trees.

$$\hat{y} = \sum_{k=1}^{K} f_k(x_i). \tag{2}$$

The objective function of XGBoost consists of a loss function and a regularization term.

$$obj = \sum_{i=1}^{5} loss(y_i, \hat{y}_i) + \sum_{k=1}^{K} \lambda(f_k), \tag{3}$$

where $\sum_{i=1}^{5} loss(y_i, \hat{y}_i)$ is the loss between output score $\hat{y}_i$ and real price y_i and $\sum_{k=1}^{K} \lambda(f_k)$ is regularization term used for alleviating overfitting. $\sum_{k=1}^{K} \lambda(f_k)$ is expressed as follows:

$$\lambda(f_k) = \gamma T + \frac{\alpha}{2}\|\omega\|^2, \tag{4}$$

where T is the number of leaf nodes and ω is the score of a leaf node. γ and α are hyperparameters. For the k-th tree, the predicted price can be expressed as:

$$\hat{y}^{(k)} = \hat{y}^{(k-1)} + f_k(x_i). \tag{5}$$

So the objective function turns to:

$$obj = \sum_{i=1}^{5} loss(y_i, \hat{y}_i^{(k-1)} + f_k(x_i)) + \sum_{k=1}^{K} \lambda(f_k). \tag{6}$$

The final selection is determined by each feature's output score. A higher score indicates a higher importance. Twenty kinds of virtual currencies are chosen to compose the portfolio pool. As shown in Table 1, we calculated the scores of each feature importance in each virtual currency. The aim of feature selection is to pick out three most important features to avoid redundant information, and hence we also calculate the final scores of the portfolio pool shown in Fig. 1. The final scores are the average of twenty assets.

In Fig. 1 the close price gets the highest score, and the high price and the low price are respectively with the second and third highest scores. Therefore, the input data in this paper is chosen as close price, high price and low price.

2.2. Proposed method

2.2.1. Trade with reinforcement learning

The Reinforcement learning focuses on goal-directed learning from interactions. Reinforcement learning is a process where the

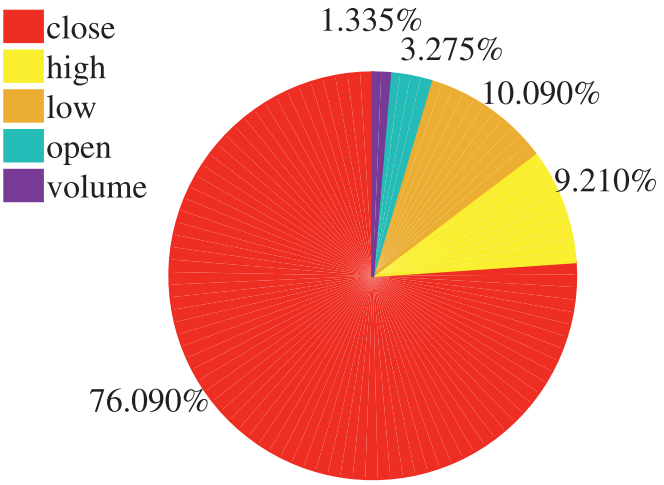


Fig. 1. Final feature importance.

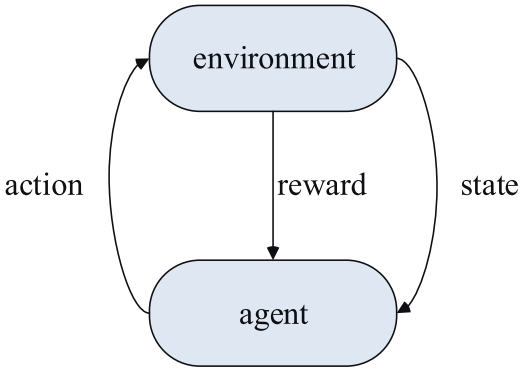


Fig. 2. Interactive process of Reinforcement learning.

system acts spontaneously to influence the environment [22]. Generally speaking, the Reinforcement learning builds models through the Markov decision process(MDP), and hence the Reinforcement learning could be represented by <S, A, T, R>, where S is the state space, A is the action space, T is the state transition function, and R is the reward function. As shown in Fig. 2, the Reinforcement learning includes five elements: environment, agent, action, reward and state. Upon receiving the agent's action, the environment's state will immediately change and a reward signal will be sent to the agent as a feedback. Afterwards the agent will take action according to the reward and the state signals, and the agent is able to learn from experience without priori knowledge.

DRL updates parameters in each iteration by a policy gradient algorithm which is optimized by gradient descent [9]. It calculates the expectation of the strategy repeatedly and reports the noise estimation of the gradient. The policy is updated according to the gradient direction. The policy gradient algorithm is able to directly optimize the expectation of a reward, thus it is more advanced than other reinforcement learning methods. In the portfolio problem, the agent is the one who executes trading actions in the market. The market includes all historical transaction data. In deep-reinforcement learning, the transaction actions are outputs from a deep network. The structure of the deep-reinforcement learning algorithm of the portfolio is shown in Fig. 3.

For a market with continuous data, the closing price at time t is also the opening price at time $t+1$. The relative price vector at time t is defined as:

$$y_t = \left[1, \frac{v_{1,t}}{v_{1,t-1}}, \frac{v_{2,t}}{v_{2,t-1}}, \ldots, \frac{v_{m,t}}{v_{m,t-1}}\right], \tag{7}$$

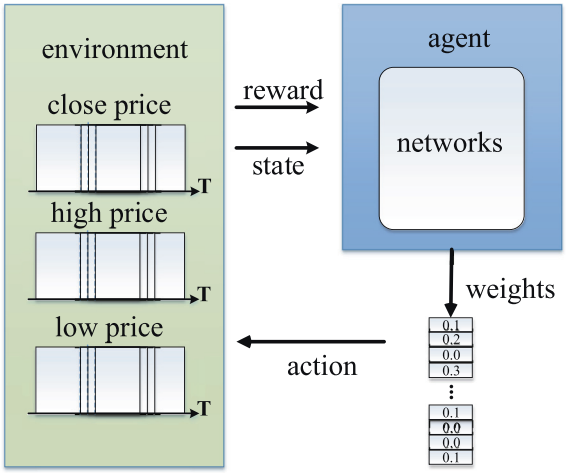


Fig. 3. Structure of deep-reinforcement learning.

where y_t is the quotient of the closing and opening prices of all assets at time t, then it can be used to calculate relative changes of the total portfolio value in a period. Assuming P_{t-1} is the portfolio value at the beginning of time t, we have:

$$P_t = P_{t-1} y_t \cdot w_{t-1}, \tag{8}$$

where w_{t-1} is a weight vector of each asset at the beginning of time t. The dimension of the vector is $m+1$. The first dimension stands for the remaining capital and m which is equaled to 20 represents the number of total assets. Therefore, the initial value of the weight vector of each asset is set to $w_0 = [1, 0, 0, \cdots, 0]^T$ indicating that no fund is allocated to any asset at the beginning.

Transactions in a practical market are not free, and hence each transaction will cost a certain commission. At the beginning time t, the portfolio's action vector is w_{t-1}. Because of the price changes, the action vector transforms to w_t after time t:

$$w_t = \frac{y_t \times w_{t-1}}{y_t \cdot w_{t-1}}, \tag{9}$$

where $\times$ represents the elementwise production and the agent's task is to redistribute the weight vector w_t by buying or selling relevant assets. w_t is composed of the asset weight and the capital weight. The capital weight represents the proportion of idle funds to total funds. The asset weight represents the proportion of each investment asset to total funds. Portfolio values with a factor μ_t represent the practical values after deducting commission. The practical portfolio values at time t is $P'_t = \mu_t P_t$. The relative return ρ_t at time t is:

$$\rho_t = \frac{P'_t - P'_{t-1}}{P'_{t-1}} = \frac{P'_t}{P'_{t-1}} - 1 = \mu_t y_t \cdot w_{t-1} - 1, \tag{10}$$

where P'_{t-1} is the practical value at time $t-1$. The corresponding logarithmic return is expressed as:

$$r_t = \ln \frac{P'_t}{P'_{t-1}} = \ln(\mu_t y_t \cdot w_{t-1}). \tag{11}$$

The final value of the portfolio is expressed as:

$$P_f = P_0 exp\left(\sum_{t=1}^{t_f+1} r_t\right) = P_0 \prod_{t=1}^{t_f+1} (\mu_t y_t \cdot w_{t-1}), \tag{12}$$

where P_0 is the initial funds 1 BTC. The goal of deep-reinforcement learning is to maximize P_f.

The Reinforcement learning aims at conducting appropriate transactions between the market and the agent. An action is taken by Reinforcement learning networks and whether to sell or to buy relevant assets is determined by their weights.

Specifically speaking, price data is sampled every 30 minutes and hence 30 minutes is defined as a time step. At the end of each time step, funds will be reallocated to all assets by Reinforcement learning networks. We assume that the portfolio is composed of m kinds of assets and that the number of total time steps is t_f. V_t is the close price vector composed of m elements, where t means the t-th time step. Particularly, $V_{i,t}$ stands for the close price of the i-th asset at the t-th time step. V_t is defined as follows:

$$V_t = (v_{1,t}, \ldots, v_{m,t}). \tag{13}$$

Close price, high price and low price constitute input data X_t. Assuming the length of input is n, the relative price V'_t is defined as:

$$V'_t = \left[\frac{V_{t-n+1}}{V_t}, \frac{V_{t-n+2}}{V_t}, \ldots, \frac{V_{t-1}}{V_t}, 1\right], \tag{14}$$

where V'_t is the relative change compared to V_t and 1 stands for a m-dimension vector $[1, 1, \ldots, 1]^T$. Similarly,hV'_t, lV'_t are defined as:

$$hV'_t = \left[\frac{hV_{t-n+1}}{hV_t}, \frac{hV_{t-n+2}}{hV_t}, \ldots, \frac{hV_{t-1}}{hV_t}, 1\right]. \tag{15}$$

$$lV'_t = \left[\frac{lV_{t-n+1}}{lV_t}, \frac{lV_{t-n+2}}{lV_t}, \ldots, \frac{lV_{t-1}}{lV_t}, 1\right]. \tag{16}$$

So the practical input to Deep learning networks X_t is:

$$X_t = [V'_t, hV'_t, lV'_t]^T. \tag{17}$$

The total volume of each virtual currency is high, but the investors usually purchase in the volume of 0.001. Therefore the transactions are relatively small. In a virtual currency market, the agent's buying and selling behaviors will not affect the future price. But the trading action made at time t will affect the reward value at time $t+1$. The reward is a feedback signal which will affect the fund allocation in the following tradings. The agent's action a_t at time t can be expressed by portfolio vector w_t:

$$a_t = w_t. \tag{18}$$

Previous action w_{t-1} affects the following reward signal and X_t is considered as one part of the trading environment. Current state s_t consists of two parts: the external state and the internal state. The external state is a three-dimensional price matrix X_t and the internal state is the last portfolio action weight vector w_{t-1}.Thus the state at time t can be expressed as:

$$s_t = (X_t, w_{t-1}). \tag{19}$$

The agent's objective is to maximize the final portfolio value of P_f. Because the agent is not able to control the initial investment value and the time range of the whole portfolio process, the goal of deep-reinforcement learning is just to maximize the average logarithmic cumulative return R:

$$R = \frac{1}{t_f} ln(\frac{P_f}{P_0}) = \frac{1}{t_f} \sum_{t=1}^{t_f+1} ln(\mu_t y_t \cdot w_{t-1}) = \frac{1}{t_f} \sum_{t=1}^{t_f+1} r_t, \tag{20}$$

where R is a cumulative reward and $\frac{1}{t_f}$ ensures the fairness of the reward function with different lengths. The strategy is implemented by a mapping function from state space to action space, namely $A = \pi_\theta(S)$. Therefore, the reward function during $[0, t_f]$ is:

$$J_{[0,t_f]}(\pi_\theta) = R(s_1, \pi_\theta(s_1), \cdots, s_{t_f}, \pi_\theta(s_{t_f}), s_{t_f+1}). \tag{21}$$

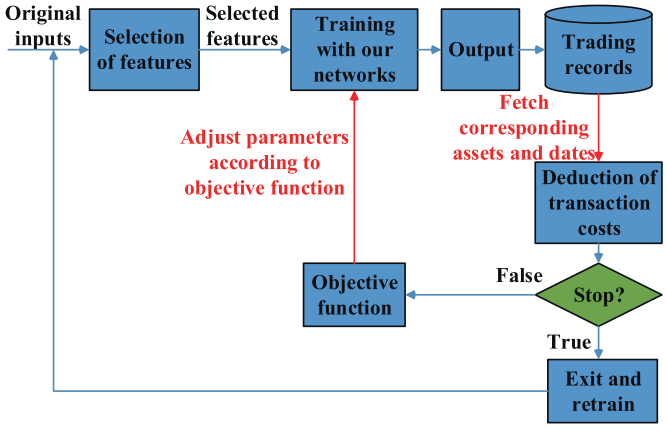


Fig. 4. Structure of Deep learning networks.

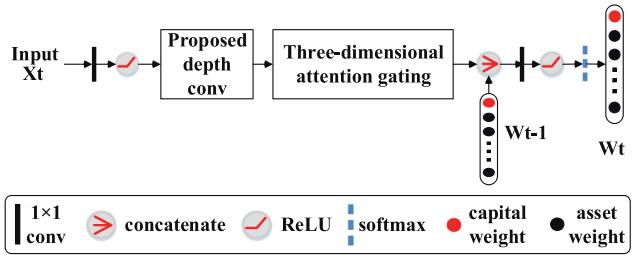


Fig. 5. Structure of Deep learning networks.

2.2.2. Reward optimization with deep learning

As shown in Fig. 4, the original inputs X are composed of x_1, x_2, x_3, x_4 and x_5 in formula (1). After feature selection, the three most important features will be chosen as the practical inputs to our networks. $X_t = [V_t', hV_t', lV_t']^T$ is the practical inputs to our networks. Our networks will output the trading actions W_t and then the relevant actions and historical data will be recorded. Next, our system will deduct transaction costs. The stop signal is the end of our transaction date. If the transaction date is not the deadline, then our network will continue to run. Our network will adjust parameters according to the objective function in formula (20). If the transaction date is the deadline, our solution will exit and start a new training cycle. The whole process will be repeated 300,000 times.

The agent is responsible for the purchase and sale of assets. This progress can be depicted by a mapping function $\pi: S \rightarrow A$, which maps state space to action space. In fact, the mapping function is just realized by Deep learning networks. In order to get the best result, the gradient descent algorithm is used to adjust mapping function's parameters θ. The cost function during [0, tf] is defined as follows:

$$C_{[0,tf]}(\pi_\theta) = -R(s_0, \pi_\theta(s_1, \ldots, s_{tf}, \pi_\theta(s_{tf}, s_{tf+1}))). \tag{22}$$

Our goal is to minimize $-R$ and it is equivalent to maximize R. The parameters θ are continuously updated with a learning rate α:

$$\theta \rightarrow \theta - \alpha \bigtriangledown_\theta C_{[0,tf]}(\pi_\theta). \tag{23}$$

Our proposed Deep learning networks are illustrated in Fig. 5. The input X_t composed of close price, high price and low price, which will be processed independently. The original CNN is proposed by LeCun et al. to solve handwritten digit recognition [23]. The algorithm of the CNN can be expressed as:

$$Conv_K(X) = X * W. \tag{24}$$

In the above formula, X is the input of the CNN, and W represents weights. K represents the kernel size of the CNN. In this paper, the above formula is used to express the CNN algorithm.

The 1 × 1 convolution will be activated by the ReLU [24] which can be expressed as:

$$ReLU(x) = \begin{cases} x, x & > \quad 0 \\ 0, x \leq 0 \end{cases} \tag{25}$$

Then the features will be processed by the proposed depth convolution, which combines 1 × 5 and 5 × 1 kernels. In the following three-dimensional attention gating module, the features are accurately calibrated. w_{t-1} will be concatenated afterwards. The subsequent 1 × 1 convolution sorts out the channels and the network outputs action w_t after being regulated by a softmax function which can be expressed as:

$$P_{xi} = \frac{e^{xi}}{\sum_j e^{xj}} \tag{26}$$

On one hand, in feature selection it has been shown that the importance of each channel is not identical. On the other hand, the inputs are 20 assets in a time series, each importance is different both in time dimension and category dimension. Since the emergence of convolution, many researchers have been continuously improving and optimizing it. Xception proposed the idea of depth separable convolutions [18]. Deep separable convolution is an efficient and lightweight convolution structure, because it operates on the channels separately. Its characteristics motivated us to propose an unique depth separable convolution.

As shown in Fig. 6(a), the six-channel feature map is obtained after a 1 × 1 convolution. Each channel will be further processed by a separate 5 × 5 convolution which helps distinguish channels with different importances. The height of inputs represents asset types and the width represents the length of time series. In this paper, the depthwise convolution is further decomposed, namely a combination of a 1 × 5 and a 5 × 1 convolutions is used to replace a 5 × 5 convolution. In Fig. 6(b), six groups of a 1 × 5 and a 5 × 1 convolution are used to handle the feature map. The 1 × 5 convolution integrates different time information and the 5 × 1 convolution integrates information of different currencies. Finally, the output is 12 channel feature maps. Features with the red channel capture time dimension correlation and features with the blue channel capture different currency correlation.

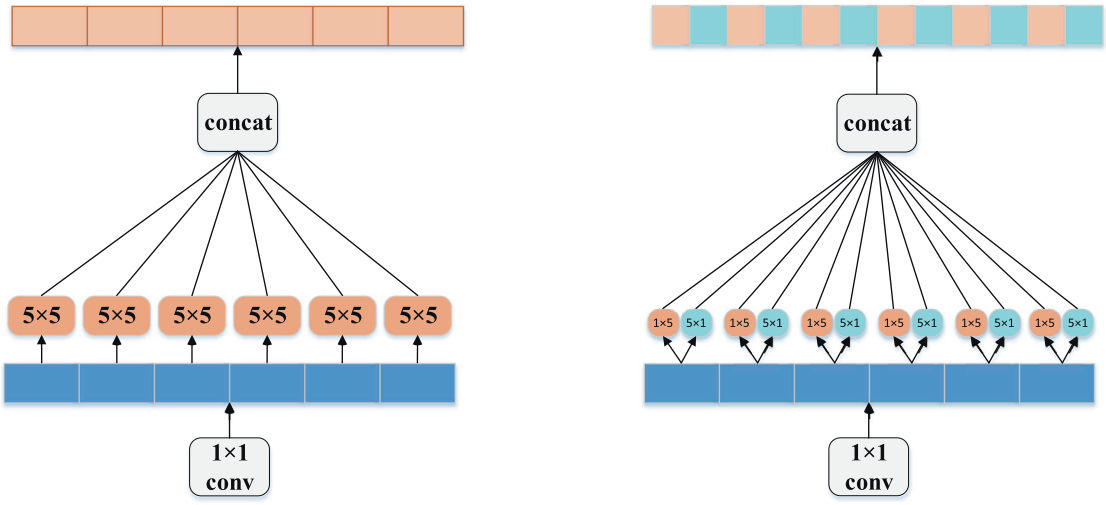
Portfolio is committed to maximizing the final return. In the financial market, asset prices soar quickly and rarely. If investors fail to seize these opportunities, the final investment returns will probably be greatly discounted. However, it is difficult to seize the opportunity of price rise only with convolution networks. In dealing with financial time series data, more attention needs to be paid to stages of price rise. Introducing an attention mechanism into Deep learning has become an effective means to improve the network performance. Hu et al. proposed a SE module based on the gated mechanism [19]. As shown in Fig. 7, the SE module assigns different weights to each channel by squeeze and excitation operations to recalibrate the features.

The SE module is a computing unit, which is able to further deal with feature channels. Assuming input X transfers to U after previous convolutions:

$$U = F_{tr}(X), \tag{27}$$

where U is composed of c-channel information. $U = (u_1, u_2, \cdots, u_c)$ and $X \in R^{H' \times W' \times C'}, U \in R^{H \times W \times C}$

Squeeze operation is the aggregation of global spatial information through global average pooling. In detail, it reduces the spatial information of U to get the SE module $z \in R^C$. The c-th element of



(a) Xception (b) Designed depth convolution

Fig. 6. Xception and our designed depth convolution.

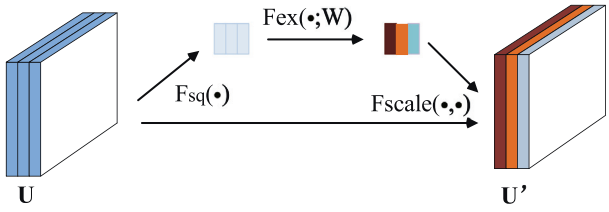


Fig. 7. SE module.

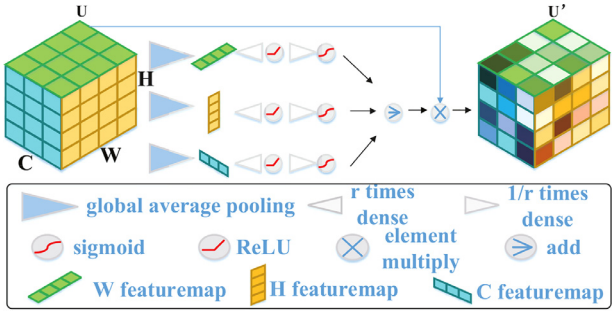


Fig. 8. Three-dimensional attention gating network.

Z can be expressed as follows:

$$Z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j). \tag{28}$$

Then the SE module performs the excitation operation:

$$s = F_{ex}(Z, W) = \sigma(g(Z, W)) = \sigma(W_2 \delta(W_{1,z})), \tag{29}$$

where $s = (s_1, s_2, \cdots, s_c)$, δ represents a ReLu function and σ represents a sigmoid function. Besides, $W_1 \in R^{\frac{C}{r} \times C}$, $W_2 \in R^{C \times \frac{C}{r}}$, in which r is the reduction ratio equivalent to 16.

The final output of the SE module is:

$$U' = [U'_1, U'_2, \cdots, U'_c], \tag{30}$$

where $U'_c = F_{scale}(u_c, s_c) = s_c \cdot u_c$ and $F_{scale}(u_c, s_c)$ involves the multiplication of scalars and features.

The SE module calibrates the weights of feature channels, which is effective in processing image information. But the SE module only compresses spatial information roughly without accurate calibration. The importance of financial information varies from time to time, especially when dealing with portfolio problems, where the rise and fall of assets always occur suddenly. In addition, the proportion of the total fund allocated to an asset needs careful consideration. Obviously, the classical Convolution Neural Networks(CNN) and ordinary gated mechanism are not able to effectively solve portfolio problems.

To solve this dilemma, a three-dimensional attention gating network is proposed. The structure of the three-dimensional attention gating network is shown in Fig. 8. The size of the feature graph is $W \times H \times C$. Firstly, the network respectively performs global average pooling on three dimensions and the corresponding weight vectors are z_W, z_H and z_C. Formulas (31) to (33) are as follows:

$$z_W = F_{sq}(u_W) = \frac{1}{H \times C} \sum_{i=1}^{H} \sum_{k=1}^{C} u_W(i, k), \tag{31}$$

$$z_H = F_{sq}(u_H) = \frac{1}{C \times W} \sum_{k=1}^{C} \sum_{j=1}^{W} u_H(k, j), \tag{32}$$

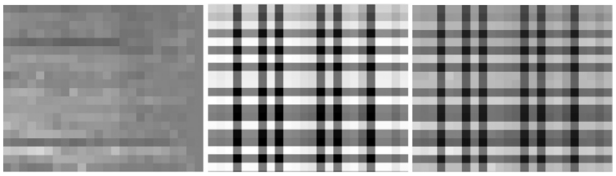
$$z_C = F_{sq}(u_C) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_C(i, j), \tag{33}$$

where$z_W \in R^W$, $z_H \in R^H$ and $z_C \in R^C$. These three vectors represent information in each corresponding dimension. The three-dimensional attention gating network is similar to the gating mechanism in the recurrent neural networks, which is composed of two fully connected layers. In the proposed attention gating network, the output number of the first fully connected layer is r times the number of inputs,and it is activated by the ReLu. The second fully connected layer restores features to their previous sizes and assigns weights between (0,1) to them by a sigmoid activation function as:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{34}$$

The specific process is shown in formula (35) to (37):

$$s_W = \sigma(W_2 \delta(W_1 z_W)), W_1 \in R^{rW \times W}, W_2 \in R^{W \times rW}, \tag{35}$$



(a) Inputs (b) Weights (c) Outputs

Fig. 9. Visualization of feature maps.

$$s_H = \sigma(W_4\delta(W_3 z_H)), W_3 \in R^{rH\times H}, W_4 \in R^{H\times rH}, \tag{36}$$

$$s_C = \sigma(W_6\delta(W_5 z_C)), W_5 \in R^{rC\times C}, W_6 \in R^{C\times rC}, \tag{37}$$

where δ stands for a ReLU function and σ stands for a sigmoid function. r is the increasing ratio and the selection of r will be given in the following section.

The last step of the three-dimensional attention gating network is the fusion of three vectors. Assuming that the weights of the three vectors are respectively $[m_0, m_1, \cdots, m_{w-1}]$, $[n_0, n_1, \cdots, n_{h-1}]$ and $[k_0, k_1, \cdots, k_{c-1}]$, the final weight vector will be:

$$W_{opq} = m_o + n_p + k_q, \tag{38}$$

where $o \in [0, w-1]$, $p \in [0, h-1]$ and $q \in [0, c-1]$. Compared with the SE module, our three-dimensional attention gating network evaluates the importance of all three dimensions. The value of W_{opq} ranges from 0 to 3. When W_{opq} belongs to (0,1), it suppresses corresponding features. And when the weight belongs to (1,3), it enhances the features.

The original input, weight matrix W_{opq} and the recalibrated features are visualized in Fig. 9. The input with 20 rows and 24 columns is shown in Fig. 9(a), and its features are not yet recalibrated. For this messy input, we are not able to clearly identify important features. After the proposed three-dimensional attention gating network, the darker elements of the weight matrix indicate that they have relatively larger values and the figure indicates that six moments of historical data and nine assets are enhanced. As shown in Fig. 9(c), the recalibrated figure directly shows those more important features.

3. Experiments & results

Unlike traditional financial products, the virtual currency could be influenced by a number of factors (including investors' confidence and relevant regulation policy). Based on this, this paper chooses virtual currencies as the assets in a portfolio to demonstrate the efficiency of our proposed method. The experimental data is obtained from Poloniex Trading Platform and the data is sampled every 30 minutes. The portfolio is composed of 20 digital currencies, which are the most popular 20 digital currencies in the market. As an effective way to judge trading strategies, back test experiments will test the effectiveness of strategies with historical data. Four back test experiments and two validation experiments are performed in this paper. The time ranges of these experiments are shown in Table 2. Virtual currencies soared sharply in the market of Back test1. The markets in Back test2 and Back test3 were relatively stable. While the market environment in Back test4 was bad because most virtual currencies declined sharply. In order to adjust network parameters in different market environments, we totally set two validations, where validation1 is a stable market and validation2 is a recession market. In all experiments, shorting is not allowed and 0.25% of transaction cost will be deducted for each transaction. The initial capital for all experiments is 1 BTC.

Table 2
Details of time.

Experiments	Training time	Testing time
validation 1	14/02/01 to 15/12/04	15/12/05 to 16/02/01
validation 2	16/03/01 to 18/01/04	18/01/05 to 18/03/01
Back test 1	14/08/01 to 16/06/04	16/06/04 to 16/08/01
Back test 2	15/02/01 to 16/12/04	16/12/05 to 17/02/01
Back test 3	15/08/01 to 17/06/04	17/06/04 to 17/08/01
Back test 4	16/02/01 to 17/12/04	17/12/05 to 18/02/01

Markowtiz held the view that an effective portfolio management method can maximize returns under certain risks or minimize risks when returns are consistent. Therefore, this paper will evaluate the portfolio from both perspectives of the return and the risk.

Assuming that the value of assets in the t-th time interval of the i-th currency is $P_{i,t}$, in order to evaluate the return of the portfolio, we define the average return as:

$$R_{ave} = \sum_{i=1}^{m}\sum_{t=1}^{t_f}\frac{R_{i,t}}{m}, \tag{39}$$

where m is the number of assets(m equals to 20 in this paper). t_f represents the total length of the time series and $R_{i,t}$ represents the return in the t-th time interval of the i-th asset.

The Sharpe ratio is one of the indicators used to assess the portfolio risk, the index was created by Sharpe. It uses the standard deviation of returns as a measurement to evaluate the excess earning per unit risk. The Sharp ratio is shown in the following formula:

$$Sharperatio = \frac{E(R_t) - R_f}{\sigma(R_t)}, \tag{40}$$

where Rf is a risk-free return, $E(R_t)$ and $\sigma(R_t)$ respectively represent expectations and variances of returns.

Another risk assessment is the maximum drawdown. The maximum drawdown is used to assess the relative risk of two adjacent periods, it measures the largest value decline rate in the portfolio. The smaller the maximum drawdown is, the smaller the short-term loss is. The definition of the maximum drawdown is defined as follows:

$$maxdrawdown = max\frac{R_t - R_{t+1}}{R_t}, t \in (1, t_f), \tag{41}$$

where R_t and R_{t+1} represent the return in the t-th and the $(t+1)$-th periods respectively.

3.1. Experiments on structure of a three-dimensional attention gating module

Inspired by the SE module, we proposed an attention gating module. Although Hu et al. conducted a lot of experiments on the SE module structure [19], the data used by Hu et al. is image information only. To solve the portfolio problem, it needs to do further experimental verifications with digital currency data. In this part, we carried out a series of experiments on the three-dimensional attention gating module. The purpose of these experiments is to finetune the module's structure. All experimental data belongs to validation1 and validation2.

In the proposed attention gating network, the hyperparameter r is used to construct the fully connected layer. The selection of r determines the model size and memory consumption. In order to balance the computational efficiency and the gating module size, we perform relevant experiments on an increasing ratio r. The up-

Table 3
Results of different increasing ratio.Squeeze operation is Average Pooling and the excitation operation is Sigmoid function. Kernel size is $1 \times 5\&5 \times 1$.

Ratio	Return(BTC)	SR	MDD(%)
4	5.828×10^{10}	0.2854	26.32
8	6.046×10^{6}	0.2257	25.77
16	$\mathbf{2.136 \times 10^{11}}$	**0.3001**	**21.43**
32	1.336×10^{10}	0.2845	27.50
4	0.253	0.0437	37.47
8	0.487	0.0600	**29.03**
16	**1.258**	**0.0784**	31.57
32	0.105	0.0248	37.62

Table 4
Results of different squeeze operation with increasing ratio=16. The excitation operation is Sigmoid function and kernel size is $1 \times 5\&5 \times 1$.

Squeeze	Return(BTC)	SR	MDD(%)
Max	5.745×10^{8}	0.2242	24.83
Avg	$\mathbf{2.136 \times 10^{11}}$	**0.3001**	**21.43**
Max	0.246	4.40	32.59
Avg	**1.258**	**0.0784**	**31.57**

Table 5
Results of different excitations with increasing ratio r=16. The squeeze operation is Average Pooling and kernel size is $1 \times 5\&5 \times 1$.

Excitation	Return(BTC)	SR	MDD(%)
ReLU	2.248×10^{10}	0.2887	21.76
Tanh	2.248×10^{10}	0.2813	26.24
Sigmoid	$\mathbf{2.136 \times 10^{11}}$	**0.3001**	**21.43**
ReLU	0.213	0.0432	37.62
Tanh	0.145	0.0322	39.13
Sigmoid	**1.258**	**0.0784**	**31.57**

Table 6
Results of different kernel sizes with increasing ratio r=16. The squeeze operation is Average Pooling and excitation operation is Sigmoid function.

Kernel size	Return(BTC)	SR	MDD(%)
3×3	2.532×10^{10}	0.2925	27.25
$1 \times 3\&3 \times 1$	1.089×10^{11}	0.2996	27.09
5×5	7.167×10^{9}	0.2645	26.69
$1 \times 5\&5 \times 1$	2.136×10^{11}	0.3001	21.43
7×7	1.327×10^{9}	0.2661	22.69
$1 \times 7\&7 \times 1$	2.064×10^{10}	0.2764	26.71
3×3	0.429	0.0575	32.68
$1 \times 3\&3 \times 1$	0.398	0.0526	37.71
5×5	1.125	0.0756	34.75
$1 \times 5\&5 \times 1$	1.258	0.0784	31.57
7×7	1.018	0.0734	32.72
$1 \times 7\&7 \times 1$	0.298	0.0463	37.07

per half of Table 3 [1] shows the results of validation1, and the lower half is the results of validation2. A portfolio generates the highest return when r is equal to 16 in both validations. The Sharp ratio is also the highest at this time, and it shows that under the same risk, increasing ratio to 16 is the most reasonable choice. From the perspective of short-term risk, $r = 16$ reduces the short-term risks as much as possible. In validation2 $r = 16$ also effectively controls the short-term risk. When the increasing ratio is 16, the goal of maximizing benefits is achieved.

[1] Table 3–6 are divided into two parts, the upper part is the result of validation1 and the lower part is the result of validation2.

In squeeze operation, we need to compress a feature map into a real number. In squeeze operation, we actually have two choices which are global average pooling and global max pooling. As shown in Table 4, the effects of these two squeeze operations are different. Average pooling is obviously superior to max pooling. Although they are very close in max drawdown index, global average pooling is still better than global max pooling.Therefore, the most suitable squeeze operation is global average pooling.

The excitation is realized by two fully connected layers. It is more reasonable to choose a non-linear activation function in the fully connected layers. Scholars generally believed that the ReLU is better than other non-linear activation functions, which is verified in the first fully connected layer. However, in the second fully connected layer, an unexpected situation occurres. As shown in Table 5, the ReLU is almost the same as tanh, and even inferior to tanh in max drawdown index. Surprisingly, the sigmoid function not only greatly improves returns, but also significantly controls the short-term risk. So the excitation in the second fully connected layer is the sigmoid function.

3.2. Experiments on structure of separable convolution

Each channel of the feature maps will be calculated by separable convolutions, and hence the size of convolution kernel has a great impact on the final return. Convolution kernels with the size of 3×3, 5×5 and 7×7 are used. Smaller convolution kernels are introduced in this paper: for example, a 3×3 convolution kernel is replaced by a combination of a 1×3 and a 3×1 kernel. The experimental results are shown in Table 6. In validation1, the combination of small convolutions is obviously better than ordinary convolutions, especially the $1 \times 5\&5 \times 1$ convolution achieves the best results. In validation2, the effects of the two methods are very close, but the results of the $1 \times 5\&5 \times 1$ are still the best.

3.3. Results of back test

Several algorithmic models and two benchmarks are introduced to test our model. The Best Stock (Best) is a benchmark widely used in portfolio selection, whose trading strategy is to invest in assets that have the best returns in the past [25]. The Uniform Constant Rebalanced Portfolio(CRP) is a more challenging benchmark, which distributed the investor's funds equally to each asset in each round [26,27]. Inspired by the mean reversion principle of financial algorithm and the confidence weighting techniques in machine learning, the Confidence Weighted Mean Reversion(CWMR) built a model on the basis of the Gauss distribution and updated the model according to the mean reversion trading principle [5]. Unlike traditional trend tracking methods, the Passive Aggressive Mean Reversion(PAMR) built models on the mean regression relationship of financial markets [4]. In addition, we also carried out relevant experiments on classical Convolutional Neural Networks(CNN) as comparisons. The structure of the classical CNN is shown in the Fig. 10 where the input X_t is processed by a 5×5 convolution separately. Without the proposed depth convolution and three-dimensional attention networks, the model structure of the CNN is consistent with that of our proposed model.

Four sets of Back tests are conducted in this paper. Back test1 uses the historical data from 2014.08.01 to 2016.08.01. At that time, many currencies showed upward trends, therefore it was the so-called bull market. Markets in Back test2 and Back test3 were very stable and the currency trends were relatively steady. Back test4 is from 2016.02.01 to 2018.02.01, during which all currencies suffered severe losses. The final experimental results of each algorithm are shown in Fig. 11 and the specific evaluation indicators are also presented in Table 7–10. In Back test 1, both the PAMR and the CWMR are profitable trading strategies. Their final returns are high, but

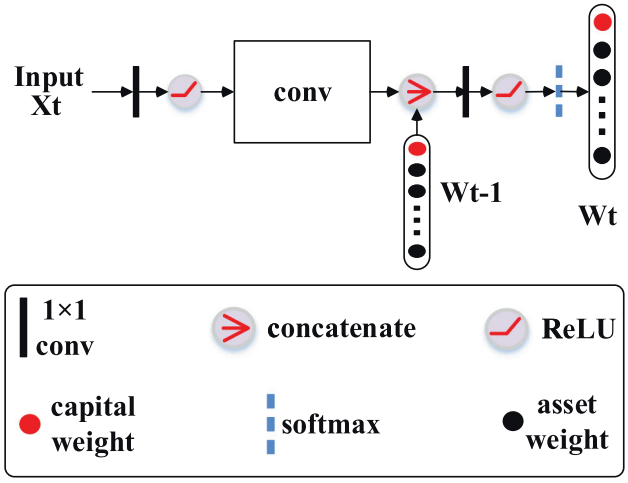


Fig. 10. Classical CNN model.

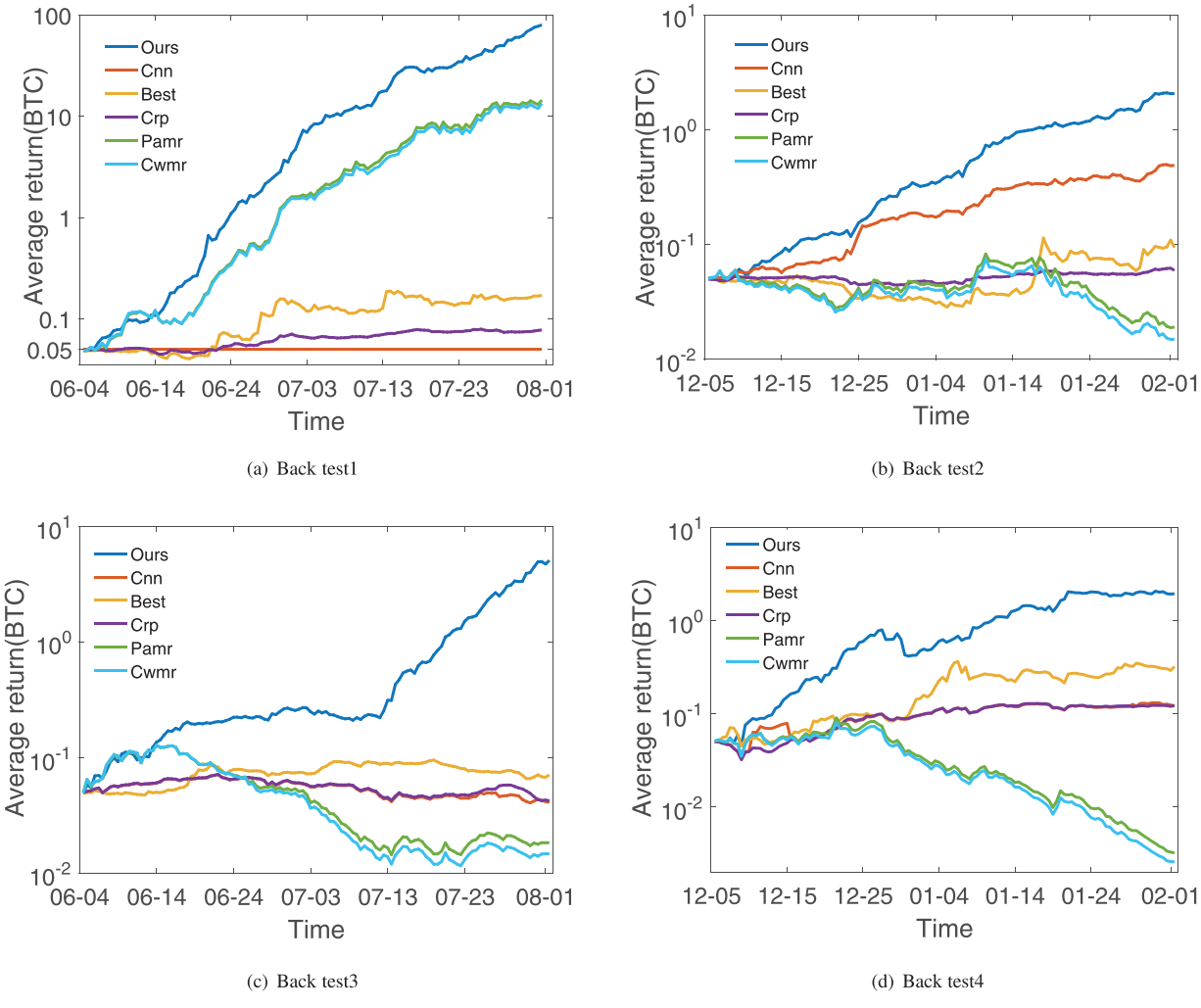
Table 7
Back test 1.

Algorithm	Return(BTC)	SR	MDD(%)
Best	0.170	0.0308	39.07
CRP	0.078	0.0407	15.85
PAMR	14.497	0.1031	35.13
CWMR	13.397	0.1007	36.33
CNN	0.050	-0.0008	**10.01**
Ours	**80.019**	**0.1369**	24.79

Table 8
Back test 2.

Algorithm	Return(BTC)	SR	MDD(%)
Best	0.094	0.0208	56.33
CRP	0.060	0.0197	17.96
PAMR	0.019	-0.0101	79.60
CWMR	0.015	-0.0142	81.39
CNN	0.490	0.0779	**17.04**
Ours	**2.069**	**0.0946**	18.83

their maximum drawdown indices are 35.13% and 36.33% respectively, which indicates that even in a promising market their short-term risks are quite high. In other three market conditions, the two strategies suffer great loss. It seems that the PAMR and the CWMR are not effective in stable or turbulent conditions. Traditional CNN is profitable in Back test2 and Back test4. However, it suffers loss in Back test1 and Back test3 and it proves that the basic CNN is not able to utilize deep features of historical data. The Best profits in all four experiments and the CRP also profits in three experiments. However, due to their inflexible investment strategies, these two algorithms also have some limitations. On one hand, both of them can make profits, but their ultimate returns are negligible,



(a) Back test1

(b) Back test2

(c) Back test3

(d) Back test4

Fig. 11. Results of backtests.

Table 9
Back test 3.

Algorithm	Return(BTC)	SR	MDD(%)
Best	0.071	0.0184	36.89
CRP	0.042	-0.0078	44.67
PAMR	0.018	-0.0125	90.00
CWMR	0.015	-0.0166	91.68
CNN	0.041	-0.0001	46.21
Ours	**5.102**	**0.0989**	**33.42**

Table 10
Back test 4.

Algorithm	Return(BTC)	SR	MDD(%)
Best	0.323	0.0425	49.25
CRP	0.123	0.0405	37.81
PAMR	0.003	-0.0298	96.67
CWMR	0.002	-0.0327	97.01
CNN	0.122	0.0409	**36.68**
Ours	**1.963**	**0.0696**	50.81

Table 11
Extension 1.

Algorithm	Return(BTC)	SR	MDD(%)
Best	1.992	0.0208	28.33
CRP	0.984	0.0007	29.12
PAMR	0.112	-0.0169	89.98
CWMR	0.088	-0.0194	92.17
CNN	0.890	-0.0206	15.13
Ours	4944.069	0.1352	31.38

Table 12
Extension 2.

Algorithm	Return(BTC)	SR	MDD(%)
Best	10.401	0.0306	58.54
CRP	2.088	0.0209	51.62
PAMR	0.001	-0.0466	99.87
CWMR	0.001	-0.0495	99.91
CNN	1.987	0.0200	51.46
Ours	1445.633	0.0779	33.09

Table 13
Ablation study in validation 1.

Algorithm	Return(BTC)	SR	MDD(%)
CNN	1.620×10^{9}	0.2430	21.83
depth	1.329×10^{11}	0.2884	23.15
depth + three	$\mathbf{2.136 \times 10^{11}}$	**0.3001**	21.43
/depth + three	1.604×10^{10}	0.2401	33.15
Xception+three	1.045×10^{11}	0.2693	25.31
depth+SE	2.114×10^{11}	0.2778	**21.40**

Table 14
Ablation study in validation 2.

Algorithm	Return(BTC)	SR	MDD(%)
CNN	0.875	-0.0056	25.53
depth	1.037	0.0387	53.72
depth + three	1.258	**0.0784**	31.57
/depth + three	1.721	0.0402	52.18
Xception+three	1.053	0.0434	36.78
depth+SE	1.121	0.059983	31.8364

on the other hand, their extremely high maximum drawdown indicators imply that investors may suddenly lose a huge amount of wealth. Our proposed method makes profits in four experiments. Compared with the Best and the CRP, the proposed system in this paper earns the highest returns, especially in Back test 1. The experimental results show that our method profits in different market environments. In the meanwhile, the maximum drawdown index of our method is quite low in the first three experiments. In Back test4 with a bad market condition, its maximum drawdown index is only 1.5% higher than that of the Best. This proves that the short-term risk can be limited within a reasonable range by the proposed system.

3.4. Robustness in a more complicate market

Digital currencies soared sharply in the market of Back test1. The markets in Back test2 and Back test3 were relatively stable. The market environment in Back test4 was bad because most digital currencies declined sharply. Although the four back tests have different market conditions, they are in a single market environment and do not involve with more complex changes. Prolonging training time will complicate the situations of digital currency market. In order to verify the robustness of the model in a more complex market environment, the time is extended to three years(2014/02/01-2017/02/01) and four years(2014/02/01-2018/02/01) respectively.

Extension 1 ranges from 2014/02/01 to 2017/02/01, during which the initially stable market gradually became prosperous and went to stable finally. The results of extension 1 are shown in Table 11. Although all assets did not fall sharply during this period, most of the algorithms still failed to achieve richprofits. Only Best and our proposed system achieved profits. Obviously, compared with the Best, the system in this paper achieved higher returns. The main reason contributing to this result was that the proposed system focused on the potential relationships between each asset and its change rules in different time periods.

Extension 2 is a more complicate market. During four years, the market gradually prospered after a stable period, and then went through another stable period and finally entered the declining situation. As shown in Table 12, algorithms of PAMR and CWMR caused different degrees of loss and other algorithms achieved profits. Unlike in Extension 1, both CRP and CNN were ultimately profitable, but their five-year investment returns were very low. After five years of investment management, their returns were improved. Unfortunately, the MDD indexes of CRP and CNN were further deteriorated, which indicated that the algorithm would suffer a huge loss in the short term. Our proposed system achieved the highest profits (though slightly lower than extension 2). The MDD index of the proposed system was not deteriorated significantly even in a bad market.

Two experiment groups with more complex market situations were added, the proposed system still achieved stable returns. The main reason is that the proposed model is not designed according to a single investment strategy, instead,it studies the potential relationship of each asset and its rules of price change at different times.

4. Ablation study

The ablation studies were performed on validation 1 and validation 2. As mentioned above, data of validation 1 came from a bull market, and data of validation 2 came from a bear market. The metrics are Sharpe ratio, maximum drawdown and final return. The results are shown in Table 13 and 14, respectively. The CNN stands for the classical convolutional networks whose concrete structure is shown in Fig. 10. The depth and three respectively represent our proposed depth convolutions and three-dimensional attention networks. In addition, if feature

selection is not conducted, we will add a / before the algorithm. The depth+three is our proposed model.

4.1. Feature selection

The three most important features were picked out through the XGBoost. The model without feature selection was tested in both validations. In validation 1, our proposed model achieved attractive profits but the profit without feature selection was only one tenth of that of the proposed model. Two risk indicators showed that the model without feature selection was prone to make risky investment. Although the model without feature selection achieved higher returns in validation 2, its extremely high MDD index indicated that this model seemed to be very dangerous in a recession market. It is effective to select important features to manage portfolios in our model.

4.2. Depth convolutions

Our designed depth proposal is able to work on three dimensions of data. Original depth convolutions only recalibrated weights on the channel dimension (the third dimension of data), and the CNN was unable to calibrate weights on data. When our proposed depth convolution works alone, it beats CNN in both validation 1 and validation 2. When comparing the depth convolution with the Xception, the depth model outperformed the Xception. Besides, the indexes of Sharp ratio and MDD showed that the risk of our model was lower than that of the Xception in both validations. The comparisons showed that the convolution acting on all three dimensions is indeed effective in a virtual currency market.

4.3. Three-dimension attention networks

Three-dimension attention networks were dedicated to the weight calibration of features emphasizing those moments when prices rise. Results in both validations implied that the introduction of three-dimension attention networks increased the portfolios profits (increased by 60% in validation 1 and 21% in validation 2). In terms of risk, the combination of depth convolution and three-dimension attention networks was better than a single depth convolutional model. The SE module in [19] was introduced as a comparison. The SE module only affected the third dimension of data and it actually improved profits over a single depth convolutional model. Our proposed model made more profits than the combination of the SE module and a depth convolution. One possible explanation was that the three-dimensional attention networks were prone to control the risk. Observing the change of Sharp ratio, we found that the model in this paper improved the index by 2% to 3%.

5. Conclusion & conclusion

Portfolio optimization is a hot topic in practical financial engineering. Its purpose is to reasonably invest a group of assets to maximize the investor's return. In this paper, the XGBoost is used to select the three most important features, which are close price, high price and low price. Obviously, the importance of each feature is not the same, and the classical convolutional neural networks can't distinguish them. Therefore a depth convolution is proposed in this paper. It processes each channel individually with a combination of a 1×5&5×1 kernel, and this approach not only accurately calculates the channel information, but also integrates the information of time and currency category. A three-dimensional attention gating module is also proposed to seize opportunities for asset growth. Finally, the Deep learning networks predict the weights of all assets and then Reinforcement learning networks buy and sell related assets according to the extracted features. The experimental results show that compared with the traditional algorithms, the proposed system in this paper can achieve higher returns in different market conditions and also controls the short-term risk within a reasonable range. In reality, the financial market is a complex and volatile environment, thus it requires portfolio management system to conduct real-time tranactions. In order to improve the transaction speed, it is necessary to minimize parameters of the network without compromising the performance. Besides, the traditional algorithms still show guiding significance in portfolio. Therefore it is feasible to combine our method with the traditional algorithms.

Disclosure statement

No potential conflict of interest was reported by the authors

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Liguo Weng: Conceptualization, Methodology. **Xudong Sun:** Software, Writing - original draft. **Min Xia:** Data curation, Supervision. **Jia Liu:** Visualization, Investigation. **Yiqing Xu:** Validation, Writing - review & editing.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of PR China (61773219), the Natural Science Foundation of Jiangsu Province (BK20161533).

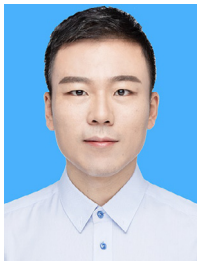
References

[1] B. Li, S.C. Hoi, Online portfolio selection: A survey, ACM Computing Surveys (CSUR) 46 (3) (2014) 35.
[2] H. Markowitz, Portfolio selection, The journal of finance 7 (1) (1952) 77–91.
[3] C.B. Kalayci, O. Ertenlice, M.A. Akbay, A comprehensive review of deterministic models and applications for mean-variance portfolio optimization, Expert Systems with Applications 125 (2019) 345–368.
[4] B. Li, P. Zhao, S.C. Hoi, V. Gopalkrishnan, Pamr: Passive aggressive mean reversion strategy for portfolio selection, Machine learning 87 (2) (2012) 221–258.
[5] B. Li, S.C. Hoi, P. Zhao, V. Gopalkrishnan, Confidence weighted mean reversion strategy for online portfolio selection, ACM Transactions on Knowledge Discovery from Data (TKDD) 7 (1) (2013) 4.
[6] H. Jang, J. Lee, An empirical study on modeling and prediction of bitcoin prices with bayesian neural networks based on blockchain information, Ieee Access 6 (2017) 5427–5437.
[7] S. McNally, J. Roche, S. Caton, Predicting the price of bitcoin using machine learning, in: 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), IEEE, 2018, pp. 339–343.
[8] L. Alessandretti, A. ElBahrawy, L.M. Aiello, A. Baronchelli, Machine learning the cryptocurrency market, Available at SSRN 3183792 (2018).
[9] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[10] D. Ormoneit, P. Glynn, Kernel-based reinforcement learning in average-cost problems, IEEE Transactions on Automatic Control 47 (10) (2002) 1624–1636.
[11] Z. Jiang, J. Liang, Cryptocurrency portfolio management with deep reinforcement learning, in: 2017 Intelligent Systems Conference (IntelliSys), IEEE, 2017, pp. 905–913.
[12] S. Almahdi, S.Y. Yang, An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown, Expert Systems with Applications 87 (2017) 267–279.
[13] S. Almahdi, S. Yang, A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning, Expert Systems With Applications 130 (2019) 145–156.
[14] M. Xia, K. Wang, X. Zhang, Y. Xu, et al., Non-intrusive load disaggregation based on deep dilated residual network, Electric Power Systems Research 170 (2019) 277–285.

[15] M. Xia, W. Liu, B. Shi, L. Weng, J. Liu, Cloud/snow recognition for multispectral satellite imagery based on a multidimensional deep residual network, International journal of remote sensing 40 (1) (2019) 156–170.
[16] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, D. Silver, Rainbow: Combining improvements in deep reinforcement learning, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[17] M. Xia, W. Song, X. Sun, J. Liu, T. Ye, Y. Xu, Weighted densely connected convolutional networks for reinforcement learning, International Journal of Pattern Recognition and Artificial Intelligence 34 (4) (2020).
[18] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.
[19] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7132–7141.
[20] J. Cumming, D.D. Alrajeh, L. Dickens, An investigation into the use of reinforcement learning techniques within the algorithmic trading domain, 2015 Ph.D. thesis.
[21] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, ACM, 2016, pp. 785–794.
[22] R.S. Sutton, A.G. Barto, Reinforcement learning: An introduction, MIT press, 2018.
[23] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.
[24] G.E. Hinton, Rectified linear units improve restricted boltzmann machines vinod nair (2010).
[25] A. Borodin, R. El-Yaniv, V. Gogan, Can we learn to beat the best stock, in: Advances in Neural Information Processing Systems, 2004, pp. 345–352.
[26] J.L. Kelly Jr, A new interpretation of information rate, in: The Kelly Capital Growth Investment Criterion: Theory and Practice, World Scientific, 2011, pp. 25–34.
[27] T.M. Cover, Universal portfolios, in: The Kelly Capital Growth Investment Criterion: Theory and Practice, World Scientific, 2011, pp. 181–209.

Liguo Weng, professor, he received his PhD degree from North Carolina A&T State University, Greensboro, NC, in 2010 and he also served as a research assistant at the National Institute of Aerospace (NIA), Hampton, VA, from2006 to 2010. He is currently a professor at the School of Information and Control Engineering, Nanjing University of Information Science and Technology, China. His research interests include intelligent systems, navigation and control, bio-inspired adaptive and control systems, and machine learning methods.

Xudong Sun, postgraduate student in Nanjing University of Information Science and Technology. His research interests are machine learning, evolutionary computation and its application.



Min Xia, associate professor, he received the Ph.D. degree in Cybernetics Control Engineering at Donghua University. He was the Assistant Researcher in Hongkong polytechnic university from 2008.8 to 2010.4. Now, he is an Associate Professor of Nanjing University of Information Science and Technology. His principal research interests are computational neuroscience and applications of machine learning methods.

Jia Liu received the Ph.D. degree in measuring technology and instrument in 2008 from Southeast University, Nanjing, P. R. China. She is currently an associate professor at Nanjing University of Information Science and Technology. Her research interests are machine learning and computational neuroscience.

Yiqing Xu, Ph.D., Assistant professor of Nanjing Forestry University, current research are Bioinformatics and Artificial Intelligence.