Article

http://justc.ustc.edu.cn

Received: April 28, 2022; Accepted: May 22, 2022

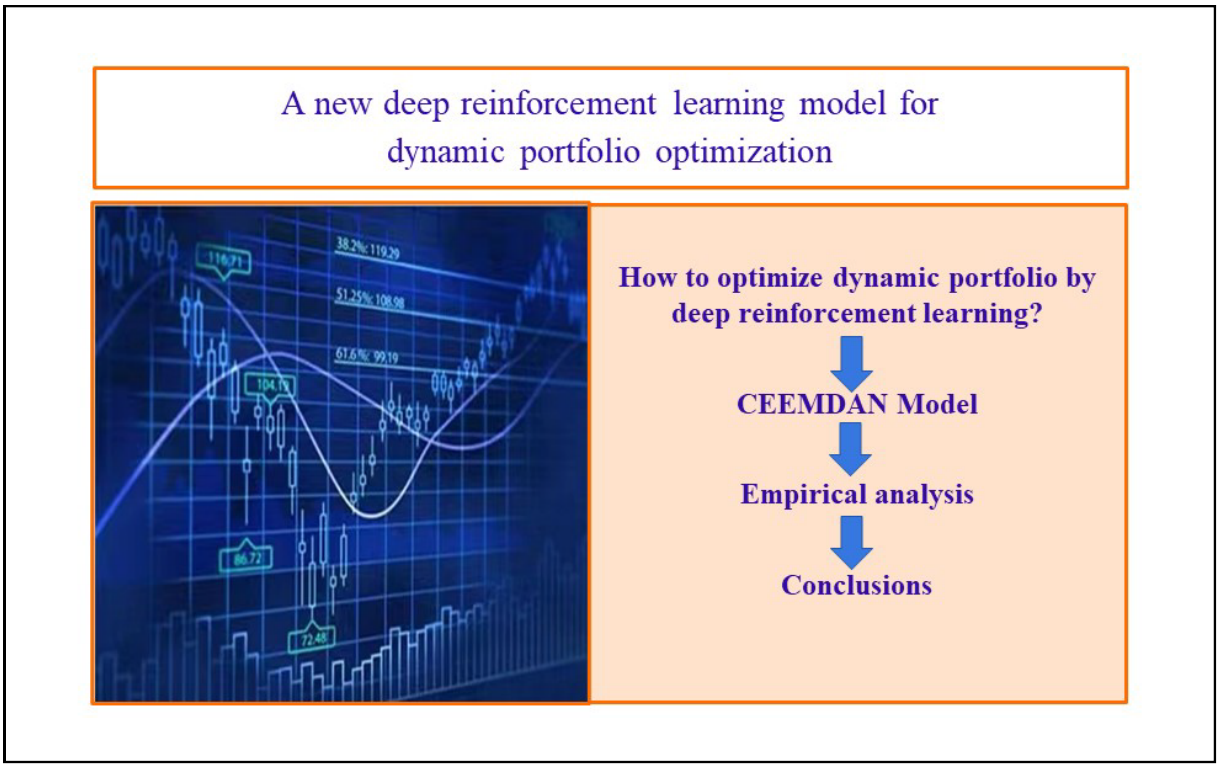
A new deep reinforcement learning model for dynamic portfolio optimization

Weiwei Zhuang[1], Cai Chen[2], and Guoxin Qiu[3] ✉

[1]International Institute of Finance, School of Management, University of Science and Technology of China, Hefei 230601, China;
[2]Department of Statistics and Finance, School of Management, University of Science and Technology of China, Hefei 230026, China;
[3]School of Business, Anhui Xinhua University, Hefei 230088, China

✉Correspondence: Guoxin Qiu, E-mail: qiugx02@ustc.edu.cn

Graphical abstract



A new deep reinforcement learning model for dynamic portfolio optimization

How to optimize dynamic portfolio by deep reinforcement learning?

CEEMDAN Model

Empirical analysis

Conclusions

The overall framework of our research.

Public summary

■ The CEEMDAN-Multi-Att-RL structure advocated in this paper improves the trading ability of financial time series data compared with the deep reinforcement learning without any processing.

■ This paper explores two investment strategies of dynamic portfolio optimization using deep reinforcement learning. Investors can choose them according to their own risk preference.

■ Appropriate deep learning network and reward settings are configured according to the given stock market environment. These enhance learning effectiveness of deep reinforcement learning.

Citation: Zhuang W W, Chen C, Qiu G X. A new deep reinforcement learning model for dynamic portfolio optimization. *JUSTC*, **2022**, 52(11): 3. DOI: 10.52396/JUSTC-2022-0072

JUSTC

Article

http://justc.ustc.edu.cn

Received: April 28, 2022; Accepted: May 22, 2022

A new deep reinforcement learning model for dynamic portfolio optimization

Weiwei Zhuang[1], Cai Chen[2], and Guoxin Qiu[3] ✉

[1]*International Institute of Finance, School of Management, University of Science and Technology of China, Hefei 230601, China;*
[2]*Department of Statistics and Finance, School of Management, University of Science and Technology of China, Hefei 230026, China;*
[3]*School of Business, Anhui Xinhua University, Hefei 230088, China*

✉Correspondence: Guoxin Qiu, E-mail: qiugx02@ustc.edu.cn

Cite This: *JUSTC*, **2022**, 52(11): 3 (13pp)

Read Online

Abstract: There are many challenging problems for dynamic portfolio optimization using deep reinforcement learning, such as the high dimensions of the environmental and action spaces, as well as the extraction of useful information from a high-dimensional state space and noisy financial time-series data. To solve these problems, we propose a new model structure called the complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) method with multi-head attention reinforcement learning. This new model integrates data processing methods, a deep learning model, and a reinforcement learning model to improve the perception and decision-making abilities of investors. Empirical analysis shows that our proposed model structure has some advantages in dynamic portfolio optimization. Moreover, we find another robust investment strategy in the process of experimental comparison, where each stock in the portfolio is given the same capital and the structure is applied separately.

Keywords: CEEMDAN; dynamic portfolio optimization; multi-head attention; Q-network; reinforcement learning

CLC number: F830.9; TP18 **Document code:** A

2020 Mathematics Subject Classification: 68T01; 91G10

1 Introduction

Reinforcement learning, a type of interactive learning without learning labels, requires constant exploration and updating of decisions in an unknown environment. In addition, the agent's strategy must be optimized through feedback from the environment. Reinforcement learning has been applied to financial markets for many years. For example, Neuneier[1] applied reinforcement learning to financial asset trading. Nevmyvaka et al.[2] performed a large-scale empirical application of reinforcement learning to optimize transaction execution in the modern financial market for the first time. Meng and Khushi[3] believed that the transaction cost can have a significant impact on the profitability of reinforcement learning, so they evaluated the impact of the transaction cost and bid/ask spread. Reinforcement learning is also often used in the stock market. However, it encounters bottlenecks in dealing with a large collection of states, such as stock data. Therefore, researchers have considered applying deep reinforcement learning for this purpose. Xiong et al.[4] took the stock prices of each trading day as the market environment. Then, they trained the deep reinforcement learning agent in the market environment and ultimately obtained an adaptive trading strategy. Their deep reinforcement learning method showed a better effect than the two benchmarks in the Sharpe ratio and cumulative return. Brim[5] applied the double deep Q-network (DDQN), a deep reinforcement learning algorithm, to a pair trading strategy in the stock market for profit. Gao et al.[6] applied another deep reinforcement learning technique, the deep Q-network (DQN) algorithm, to stock market production. Moreover, Lee et al.[7] processed high-frequency data through wavelet transformation and then used deep reinforcement learning to trade in financial time series.

Many works relative to quantitative finance based on deep reinforcement learning have often been limited to single assets and lacking in the dynamic optimization of portfolios, as with Carta et al.[8] and Théate and Ernst[9]. The portfolio can increase capital capacity and have more application scenarios in quantitative transactions. In addition, optimizing the investment portfolio can provide hedging ability and make the return more robust. This paper explores two investment strategies of dynamic portfolio optimization using deep reinforcement learning. The first strategy is to give a fund that can meet the portfolio's daily trading volume and then use deep reinforcement learning to dynamically adjust the position of stocks in the portfolio on each trading day. The other strategy is to allocate the fund to each stock in the portfolio equally and then use the strategy of deep reinforcement learning to act on each stock. Moreover, to restore the details of the stock market environment, high-frequency stock data are used for the simulation. The high-frequency data of multiple stocks in a portfolio lead to high state-space dimensions. Similar to Lei et al.[10], we adjust the network structure to improve the learning ability of a deep learning network. In this paper, the multi-head attention network, introduced by Vaswani et al.[11], is considered. The stock high-frequency data are of-

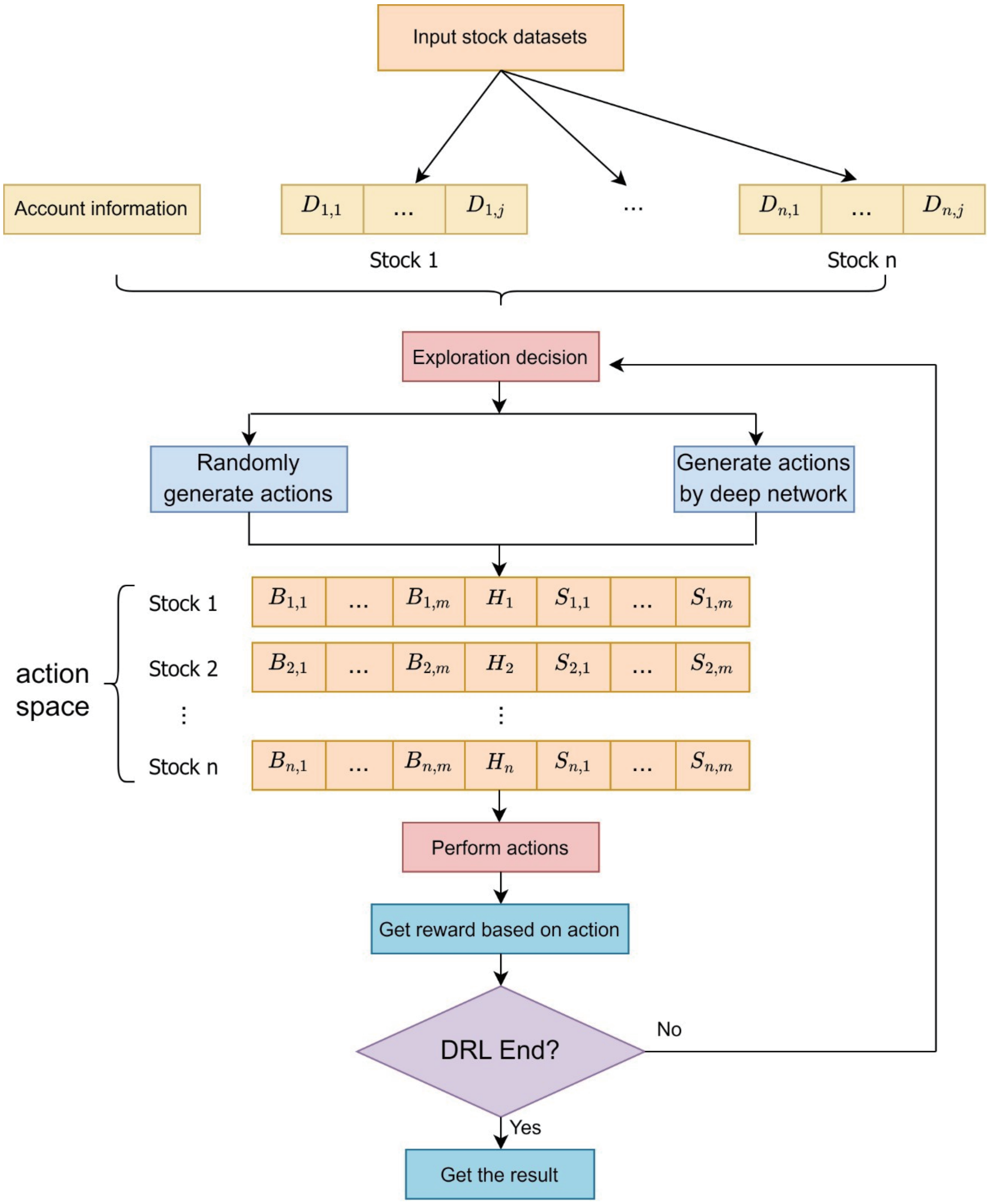

Input stock datasets
Account information
$D_{1,1}$ … $D_{1,j}$
Stock 1
…
$D_{n,1}$ … $D_{n,j}$
Stock n
Exploration decision
Randomly generate actions
Generate actions by deep network
action space
Stock 1
$B_{1,1}$ … $B_{1,m}$ H_1 $S_{1,1}$ … $S_{1,m}$
Stock 2
$B_{2,1}$ … $B_{2,m}$ H_2 $S_{2,1}$ … $S_{2,m}$
Stock n
$B_{n,1}$ … $B_{n,m}$ H_n $S_{n,1}$ … $S_{n,m}$
Perform actions
Get reward based on action
DRL End?
No
Yes
Get the result

Fig. 1. Model structure.

ten mixed with a large amount of noise, so the original high-frequency data are decomposed and denoised by the complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) method to reduce the effect of noise and maintain the form of the time-sharing charts of each factor of the stock as much as possible. The CEEMDAN method is an improved version of the empirical mode decomposition (EMD). The EMD method is an adaptive signal time-frequency processing method proposed by Huang et al.[12], which can analyze nonlinear and nonstationary data. However, the EMD method still has a mode-mixing problem. Torres et al.[13] proposed the CEEMDAN method, in which a specific noise is added at each stage of data decomposition, and a unique residue is computed to obtain each mode. In this paper, we propose the structural CEEMDAN_multi-head_attention_reinforcement learning (Multi_Att_RL) method, which integrates the CEEMDAN method, multi-head attention network, and reinforcement learning. The simulation results show that this structure has a better effect on the two different investment strategies mentioned above.

The rest of this paper is organized as follows. The methodology used in this paper is introduced in Section 2. The empirical part is given in Section 3, and we verify the performance of the proposed model structure and carry out multiple groups of check experiments in this section. Section 4 concludes this work.

2 Methodology

Trading with reinforcement learning requires continuous

DOI: 10.52396/JUSTC-2022-0072

perception and decision-making. Environmental perception requires learning feature representations from high dimensions of environment space and noisy financial time series. Decision-making requires a model to explore the environment online without any supervision. To address these two issues, we propose the CEEMDAN_Multi_Att_RL structure, which combines several methods to improve the state-space representation learning and decision-making abilities. In this section, the operation of the structural CEEMDAN_Multi_Att_RL model is described in detail. Then, several methods involved in the structure are introduced, including the CEEMDAN method and multi-head attention network. Finally, we illustrate the updating of the network in deep reinforcement learning along with the set of the state space and reward.

The CEEMDAN_Multi_Att_RL structure, an improved version of deep reinforcement learning, is described below. First, to reduce noise and eliminate the impact of data dimensions, the original high-frequency stock data are processed with the CEEMDAN method. The high-frequency data are available at the minute level. For each stock, we gather its five basic technical indicators, namely Open, High, Low, Close, and Volume. We also construct other quantitative factors by these basic technical indicators. Then, the CEEMDAN method is used to decompose the factor sequence values of each stock. Hence, several decomposition modes and a residual of each factor sequence values can be obtained. Some of the necessary decomposition modes are extracted to reconstruct the original factor sequence. The reconstructed factor sequence values can better retain the waveform of the original factor sequence. Moreover, the sequence values are normalized. Using these sequence values as input to the model is beneficial for machine learning and training. Next, the account information and the factor sequence values processed by the CEEMDAN method of each stock are set as the input of the whole model. The account information includes the balance and position information of each stock. Then, reinforcement learning with the multi-head attention network is performed on the input data.

The model generates action through the multi-head attention network with the probability of $1-\epsilon$, and generates action randomly with the probability of ϵ, where ϵ decreases with time t. This process can be expressed as

$$\epsilon \leftarrow \epsilon/(t+1).$$

After the model executes the action, the model gets the corresponding reward and reaches a new state. According to the new reward and state, the model obtains new strategies and

Algorithm 2.1 CEEMDAN_Multi_Att_RL algorithm.

```
Require: Technical indicators, including the Open, High, Low, Close, Volume, Amount, Open_MA_2, Open_MA_5, and Open_MA_10 (All technical indicators are at the minute level);
Ensure: Sequence value processed by the CEEMDAN method;
Perform CEEMDAN so that $\overline{\mathrm{IMF}_1},\cdots,\overline{\mathrm{IMF}_n}$ and $R_n(t)$ of each factor sequence are obtained;
For the $j$th factor of the $i$th stock, add the values of the corresponding positions of arrays $\overline{\mathrm{IMF}_2},\cdots,\overline{\mathrm{IMF}_{n-1}}$ to obtain a new sequence value $D_{i,j}$;
Require: Sequence value $D$ of technical indicators processed by the CEEMDAN method, Account information (balance and position information of each stock), Q network architecture and $\epsilon$;
Initialize all parameters $\theta$ of the Q network randomly;
Initialize the action-value function $Q$ corresponding to all states and actions based on $\theta$;
Initialize replay memory $\mathcal{D}$;
for $i = 1,\dots,t$ do
  Initialize state $S$ to obtain $s_1$;
  Initialize a random process $\epsilon$ for action;
  Take $s_t$ as the input of Q network to obtain the Q value outputs corresponding to all actions;
    Select $a_t = \mathrm{argmax}_{a_t} Q(s_t,a_t,\theta)$;
    Execute the action $a_t$ in the state $s_t$ to obtain the new state $s_{t+1}$ and reward $r_t$;
    Decide whether to terminate the states (is_end = true/false);
    Save ($s_t$, $a_t$, $r_t$, $s_{t+1}$, is_end) to replay memory $\mathcal{D}$;
    $S = s_{t+1}$;
    M samples ($s_k$, $a_k$, $r_k$, $s_{k+1}$, is_end) are sampled from replay memory $\mathcal{D}$, and calculate the current target Q value $y_k$;
    $y_k = \begin{cases} r_k, & \text{is\_end} = \text{true}; \\ r_k + \gamma \max_{a'} Q(s',a';\theta_{k-1}), & \text{is\_end} = \text{false}; \end{cases}$
    Use the mean squaresp loss function:
    $L_k(\theta_k) = \mathbb{E}_\pi[(r_k + \gamma \max_{a'} Q(s',a';\theta_{k-1}) - Q(s,a;\theta_k))^2]$;
    The gradient back propagation of the neural network is used to update all the parameters $\theta$ of the Q network;
    If $s_{t+1}$ is in the termination state, the current round of iteration is completed; otherwise, continue to iterate;
  end for
```

generates new actions. When the structure is applied to stock trading, the actions include buying, holding, and selling. The model continues to learn until the end of the training. The flowchart of the whole model framework is shown in Fig. 1. In the flowchart, $D_{i,j}$ shows the sequence value of the jth factor of the ith stock in the stock dataset. These sequence values are processed by the CEEMDAN method. It is assumed that there are n stocks in the flow chart, and each stock has j factors. All $B_{i,j}$, H_i, and $S_{i,j}$ constitute the action space in reinforcement learning, in which $B_{i,j}$ represents the jth buying volume of the ith stock, H_i represents holding the ith stock without any trading, and $S_{i,j}$ represents the jth selling volume of the ith stock. In Fig. 1, m represents the number of discrete values of the possible trading volume. The learning process of the proposed CEEMDAN_Multi_Att_RL structure is presented in Algorithm 2.1 through simple pseudocode. Later in the paper, we introduce each part of the proposed structure, including the specific data processing process, the multi-head attention, the description of the network update, and the set of the state space and reward. Finally, the two investment strategies mentioned in Section 1 are described.

2.1 CEEMDAN

To overcome the difficulty of the neural network processing stock data with large fluctuations and much noise, it is necessary to preprocess the original stock data. Thus, the CEEMDAN method is adopted. This method can not only eliminate mode mixing effectively but also make the reconstruction error of the sequence almost zero. Moreover, the computational cost is greatly reduced. The CEEMDAN method decomposes the original signal as follows:

$$X(t) = \sum_{i=1}^{n} \overline{\mathrm{IMF}_i} + \mathrm{R}_n(t),\ t > 0, \tag{1}$$

where $\overline{\mathrm{IMF}_i}$ is the ith decomposition mode, n represents the degree of signal decomposition and $\mathrm{R}_n(t)$ is the residual.

According to Eq. (1), we combine $\overline{\mathrm{IMF}_2},\cdots,\overline{\mathrm{IMF}_{n-1}}$ to obtain $D_{i,j}$ to approximate the waveform of the original signal by the CEEMDAN method. The combined waveform can eliminate the influence of dimension. In addition, the waveform oscillates around zero, so the value of the waveform is conducive to machine training. As the noise of $\overline{\mathrm{IMF}_1}$ is relatively large, it is not combined with $\overline{\mathrm{IMF}_2},\cdots,\overline{\mathrm{IMF}_{n-1}}$.

2.2 Multi-head attention

The attention function can be described as a mapping query ($\boldsymbol{q}$) and a set of key-value ($\boldsymbol{k}$-$\boldsymbol{v}$) pairs to the output, where $\boldsymbol{q}$, $\boldsymbol{k}$, $\boldsymbol{v}$, and the output are vectors. The output is calculated as the weighted sum of $\boldsymbol{v}$, where the weight assigned to each $\boldsymbol{v}$ is in terms of $\boldsymbol{q}$ and $\boldsymbol{k}$. Scaled dot product attention is a normalized dot product attention, where the dimensions of query and key are d_k, while the dimension of value is d_v. In practice, a set of attention functions of queries are calculated simultaneously, and queries are packed together into a matrix $\boldsymbol{Q}$. Meanwhile, keys and values are also packed together into matrices $\boldsymbol{K}$ and $\boldsymbol{V}$. Therefore, the scaled dot product attention function is calculated as follows:

$$\mathrm{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V}. \tag{2}$$

Usually, it is not enough for one scaled dot product attention to operate among $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$. This deficiency leads to the proposal of multi-head attention. The operation process of multi-head attention involves making a linear transformation for $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$, and then applying the scaled dot product attention to calculate the corresponding results. If we carry out the above operations many times and combine the results obtained each time, then we can make a linear transformation to obtain the output. The formula obtained by Vaswani et al.[11] is as follows:

$$\mathrm{Multihead}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{Concat}(\mathrm{head}_1,\ldots,\mathrm{head}_h)\boldsymbol{W}^O, \tag{3}$$

$$\mathrm{head}_i = \mathrm{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V),$$

where h indicates the number of parallel attention layers or heads, $\boldsymbol{W}_i^Q \in \mathbb{R}^{d_{\mathrm{model}}\times d_k}$, $\boldsymbol{W}_i^K \in \mathbb{R}^{d_{\mathrm{model}}\times d_k}$, $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{\mathrm{model}}\times d_v}$, $\boldsymbol{W}^O \in \mathbb{R}^{hd_v\times d_{\mathrm{model}}}$, and $d_k = d_v = d_{\mathrm{model}}/h$. Compared with scaled dot product attention, multi-head attention allows the model to focus on the information of different representation subspaces from different locations; however, its operation efficiency is not reduced. The structure of multi-head attention is shown in Fig. 2.

2.3 Construction and updating of the network

In this paper, the Q-network in deep reinforcement learning is modified, as shown in Fig. 3, in which the multi-head attention network is abbreviated as Multi_Att. The attention mechanism embedded in the multi-head attention network is a problem-solving method that mimics human attention. Usually, the attention mechanism is applied to filter out high-value information from large amounts of information. High-frequency stock data are large and irregular. In order to learn different parts of the state space with different degrees of attention in reinforcement learning, the hidden layer of the Q-network adopts a two-layer multi-head attention structure to improve the learning accuracy. Finally, a dense layer (fully connected layer) is added before using the softmax layer as the output. Specifically, the dense layer can greatly reduce the impact of the feature location on classification.

To obtain the maximum return, the deep reinforcement learning guides the actions of every step through continuous interaction with the environment and learning strategies. In reinforcement learning, the action-value function is defined as $Q(s,a)$. The specific formula proposed by Sutton and Barto[14] is as follows:

$$Q(s,a) = \mathbb{E}_\pi(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s, A_t = a),$$

where S_t is a state at time t in its environment state set, and A_t is an action at time t in its action set. R_{t+1} obtained in terms of action A_t taken in state S_t, is the reward at time $t+1$. γ is the discount factor that weights the importance of future rewards versus immediate rewards. π represents the strategy, which is the basis for taking action. Using the Bellman equation[15], the action-value function can be simplified as follows:

$$Q(s,a) = \mathbb{E}_\pi(R + \gamma Q(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a).$$

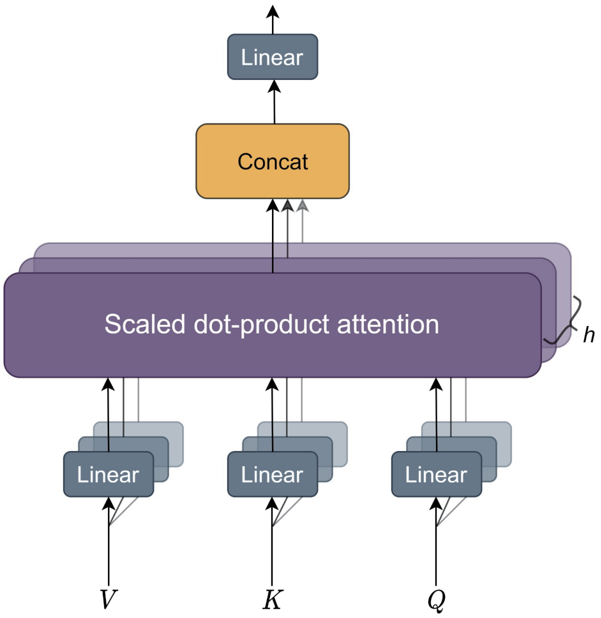


Fig. 2. Multi-head attention.

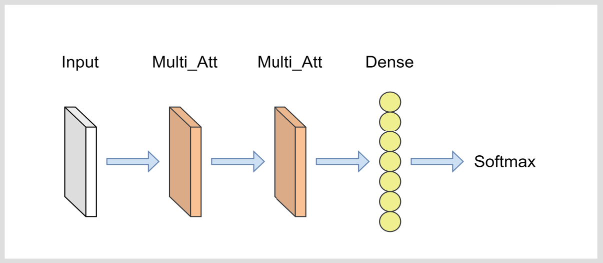


Fig. 3. The structure diagram of the Q-network.

For all possible actions a', if the optimal value $Q(s',a')$ of sequence s' is known in the next time step, we need to choose action a' to maximize the expected value of $R+\gamma Q(S_{t+1},A_{t+1})$. Then, the optimal action-value function is defined as follows:

$$Q(s,a)=\mathbb{E}_\pi\left(R+\gamma\max_{a'}Q(s',a')\mid S_t=s,A_t=a\right).$$

Deep reinforcement learning uses the Q-network to estimate the action-value function Q; namely, $Q(s,a;\theta)\approx Q(s,a)$. The loss function $L_k(\theta_k)$ of the Q-network can be set at the kth iteration, and the formula is given by

$$L_k(\theta_k)=\mathbb{E}_\pi[(R+\gamma\max_{a'}Q(s',a';\theta_{k-1})-Q(s,a;\theta_k))^2].$$

The Q-network can be trained by minimizing a series of loss functions, $L_k(\theta_k)$.

2.4 Setting of state space and reward

This section mainly introduces the state space and reward. Deep reinforcement learning is open. A series of factors can be customized, such as the environment, state, action, reward, and deep learning network. For the stock market, the target of deep reinforcement learning is to maximize reward. In this paper, the reward is set as the return value of the portfolio. At the same time, the agent can have the environment for training resemble a real stock market transaction environment by adding various constraints (e.g., transaction cost). Referring to Lee et al.[7], some basic stock factors after preprocessing are used to constitute the state in reinforcement learning, such as the stock price, trading volume, and other quantitative factors constructed by these basic factors.

To restore the basic information of the stock market as much as possible, we make a personalized construction of the state space for the stock market environment. The state space contains account balance information, stock position information, stock price information (according to the investor's trading time, such as open price), and various factors of the stock processed by the CEEMDAN method. The information is updated every trading day. Therefore, the agent can continuously improve the strategy ability through trial and error.

Meanwhile, if we take the investor's trading at the opening as an example, then the return of each stock between adjacent trading days is given by

$$\mathrm{R}_{i,t}=\frac{\mathrm{Open}_{i,t+1}-\mathrm{Open}_{i,t}}{\mathrm{Open}_{i,t}}\cdot(1-\mathrm{tax\_fee}),\quad i=1,\cdots,n,$$

where $\mathrm{R}_{i,t}$ represents the rate of return of the ith stock at time t, $\mathrm{Open}_{i,t}$ represents the open price of the ith stock at time t, and tax_fee is the transaction fee. tax_fee does not exist if there is no position change in the corresponding stock.

2.5 Description of the two strategies

This paper applies the CEEMDAN_Multi_Att_RL structure to two different investment strategies. Section 1 briefly introduces the principles of these two strategies. In this section, we introduce these two strategies in more detail through Fig. 4. First, we assume that the total capital is M, and the stock pool of the portfolio consists of n stocks. For the first investment strategy, the CEEMDAN_Multi_Att_RL structure dynamically allocates the total capital M to n stocks on each trading day. Not all capital is used for trading on each of these trading days. When extreme market conditions are predicted, the structure frees up part of the capital for hedging. However, in the second investment strategy, the total capital is divided into n equal parts in advance to ensure that each stock has the same initial capital. Each stock is dynamically adjusted by an independent CEEMDAN_Multi_Att_RL structure with the same initialization parameters. Similar to the first strategy, a portion of the funds are freed up for each structure. For each trading day, the average of the returns obtained by n structures is the return of the second strategy.

3 Empirical analysis

In this section, we empirically show the effectiveness of the CEEMDAN_Multi_Att_RL structure. Aiming at China's stock market, ten stocks in the SSE 50 Index are set as the stock pool of the model. Then, a variety of structures are applied to the portfolio to verify whether the data preprocessing and the change in the deep network improve the effect. Moreover, the two investment strategies mentioned in Section 1 are adopted to compare their performance. Additionally, the uniqueness of this structure in single-stock trading is verified through experiments. Details of the experiments are described below.

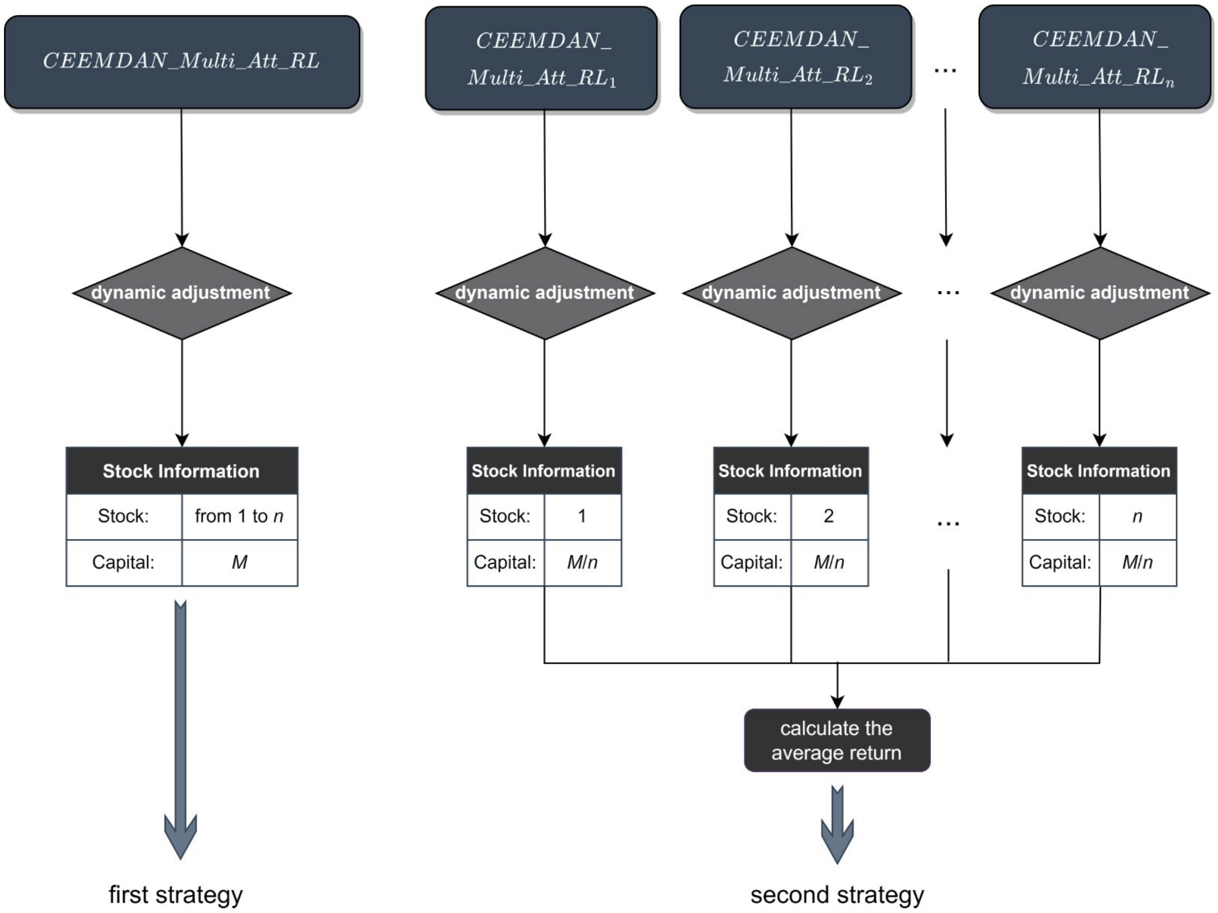


Fig. 4. Principles of the two strategies.

3.1 Training set and test set

The reason for choosing the constituent stocks of the SSE 50 Index as the stock pool of the model is that these stocks have a large market capitalization and high liquidity. These characteristics can reflect the current market circumstances to a certain extent. To verify the effect of the proposed CEEMDAN_ Multi_Att_RL structure, the structure is trained and tested in three different time periods. As shown in Fig. 5, three periods span 2017–2021. Moreover, the interval of each training set and test set is a year. The reason for choosing these time periods is that the characteristics of the stock market in these years are different. The market fell in 2018, rose in 2019, and was volatile in 2020. Testing separately in different markets is helpful to test the effect of the model.

3.2 Data preprocessing

Data cleaning has always been a key part of using models to simulate and backtest historical data in financial markets. These models are based on algorithmic trading methods, such as machine learning and deep learning. Combining the characteristics of China’s stock market and the requirements for the reinforcement learning model, this paper performs the following data cleaning of the market data of individual stocks.

First, we should fill in the gaps in the individual stock data owing to the suspension of some stocks and other reasons to ensure that the length of each stock data sample is the same as

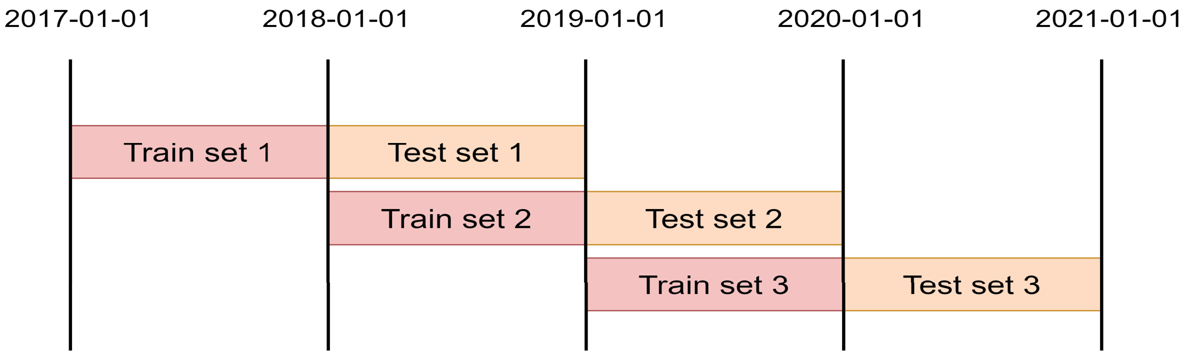


Fig. 5. Train set and test set.

DOI: 10.52396/JUSTC-2022-0072

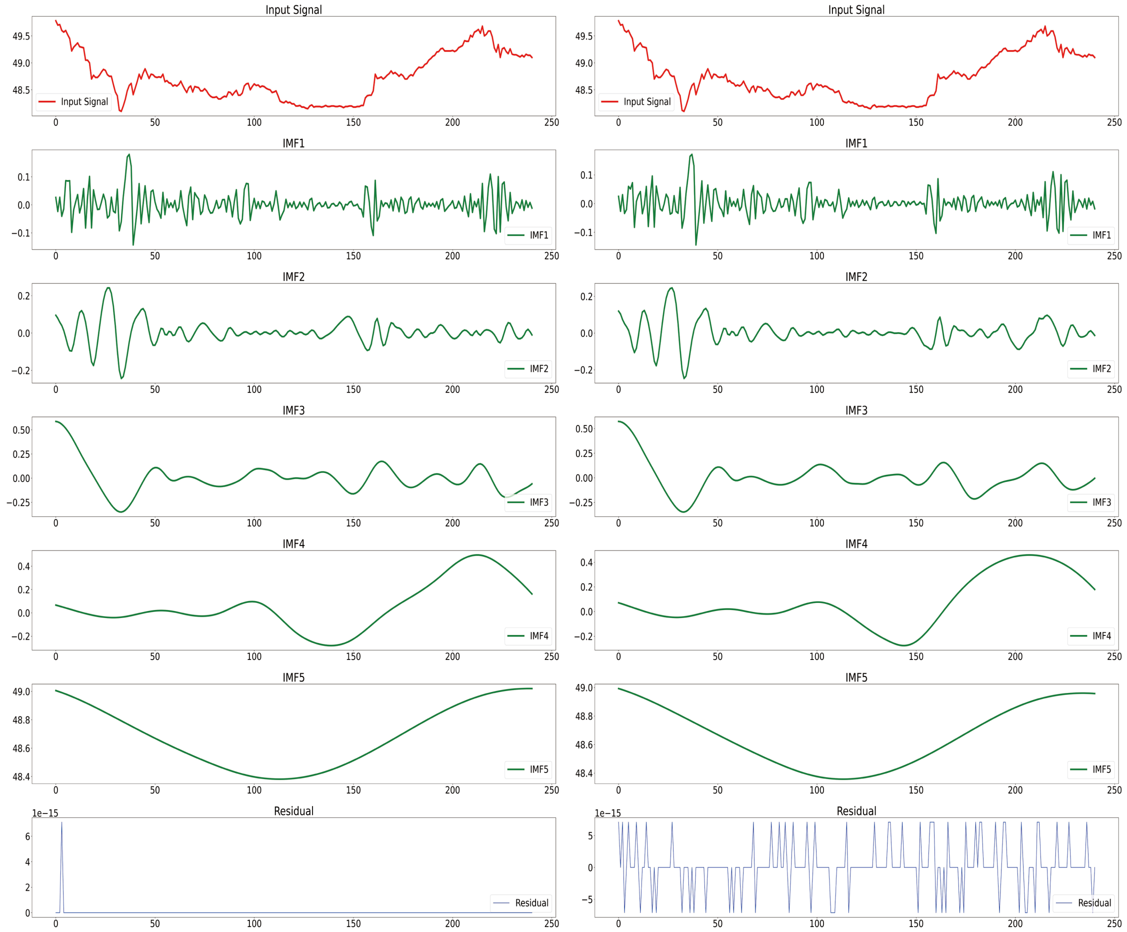


Fig. 6. The IMF components of code 600009, the left column for EMD and the right column for CEEMDAN.

the transaction date. This approach ensures that the dimensions of the stock in the process of setting the environment space are the same. Second, we should limit the ups and downs of stock data. China’s stock market is limited to 10% for the main board. However, due to errors in the market database or other reasons, there are occasionally overshot and oversold data in the raw data. To avoid the interference of abnormal price information, it is necessary to restore the stock price that exceeds the decline or the increase to the prescribed limit level.

Considering China’s stock market, each stock generates 241 minutes of data during the 4-hour trading period from 9:30 to 11:30 and 13:00 to 15:00 on each trading day. The fluctuation of the data in minutes is often abnormally large. As there are many retail investors in China’s stock market, the operation of retail investors produces strong noise. If the minute data are directly used, the amount of data and the data noise will be significant. Therefore, CEEMDAN is considered to reduce noise. The frequency decomposition of data often brings unexpected results. To justify the CEEMDAN method, the stock numbered 600009 is chosen randomly. The EMD and CEEMDAN methods are used to process the minute data of the opening price of this stock on January 11, 2019. The processing results are shown in Fig. 6. We can see that the later IMF curve is smoother than the original signal. However, the general trend of the original signal is retained. According to Section 2.1, we combine $\overline{\mathrm{IMF}_2},\cdots,\overline{\mathrm{IMF}_{K-1}}$ to approximate the waveform of the original signal. K is a parameter that can be set according to the requirements. The comparison between the combined result of $\overline{\mathrm{IMF}_2},\cdots,\overline{\mathrm{IMF}_{K-1}}$ and the original signal is shown in Fig. 7. Similarly, the EMD method is used for the same processing. Its results are shown in Fig. 7.

3.3 Analysis of trading points using the structure

A good stock market investment strategy model reasonably allocates funds to the stocks in the portfolio. In addition, the strategy can be used to buy low and sell high for these stocks as much as possible. This means that the strategy can predict a low price of the stock within a trading time range. Hence, the model buys the stock. When the stock is predicted to reach a high price, the model sells the stock to earn the spread. The CEEMDAN_Multi_Att_RL structure is used to analyze the trading time points of three different stocks in different years. The results are shown in Fig. 8. The ordinate labels of the corresponding subgraphs in the figure indicate the open prices of the stock. For example, price_601166 is the open price of the stock numbered 601166. The curves in Fig. 8 show the price trend of the corresponding stock in the corresponding year. The red dots indicate that the model has bought the stock at these time points, the green dots indicate that the model has sold the stock, and other points mean that

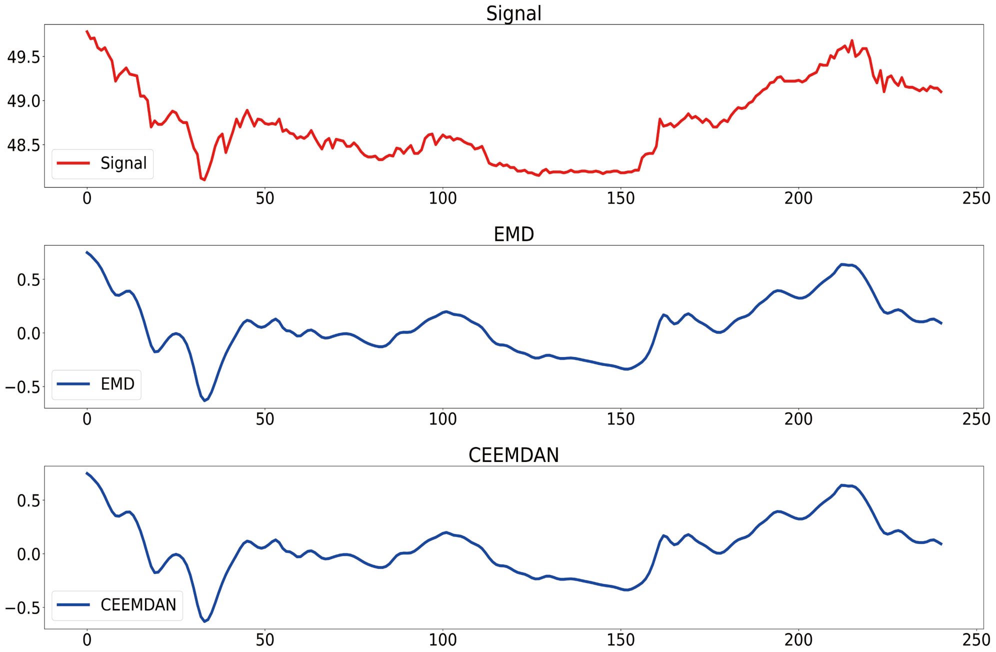


Fig. 7. The CEEMDAN and EMD method to process stock minute data.

the model has not carried out any operation. Observing the three subgraphs in Fig. 8, we find that when the model predicts an extreme downward market, it reduces the corresponding stock positions as much as possible or even directly selects short positions. At the same time, when a volatile or rising market is predicted, the model reasonably adjusts the position to capitalize on the price difference as much as possible.

3.4 Setting of comparative experiment

In this part, two investment strategies of dynamic portfolio optimization using deep reinforcement learning are explored to compare their performance. Moreover, we test, in terms of stock market trading, whether the CEEMDAN_Multi_Att_RL structure improves the effect relative to deep reinforcement learning without any changes. For the purpose of subsequent comparison of the backtest and statistical values, the structures of various modes are abbreviated. The modes include benchmark, CMAR_10, MAR_10, CDRL_10, DRL_10, CMAR_1_m, MAR_1_m, CDRL_1_m, and DRL_1_m. The benchmark represents the average compound interest value for a portfolio composed of 10 stocks. In the abbreviation of these modes, 10 means using the first investment strategy, while 1_m means using another investment strategy. CMAR represents the CEEMDAN_Multi_Att_RL structure. MAR represents the CMAR without data processing. Unlike CMAR, CDRL does not change the deep learning network. Similarly, unlike CDRL, DRL does not include data processing.

To carry out the dynamic position, the structures perform operations according to the model's strategy on each trading day. For structures using the first investment strategy, the actions are for the portfolio. Similarly, for structures using another investment strategy, the actions are for a single stock. For practical operability, the actions taken by each stock are integers. If the action value is positive, we buy the quantity of the corresponding amount. If the action value is negative, we sell the quantity of the corresponding amount. Finally, if the action value is zero, we do not take any action. All structures consider adjusting positions when trading opens.

3.5 Analysis of the compound interest results of the backtest

Figs. 9−11 show the performance of different structures in different years. The left subgraphs of each graph represent the result of the first investment strategy, and the right subgraphs represent the result of another investment strategy. We can see from Fig. 9 that the compound interest of CMAR_10 structure is mostly in the leading state, but it fluctuates greatly. In addition, although the CMAR_1_m structure and MAR_1_m structure do not stay ahead most of the time, their compound interest trend is stable, and the CMAR_1_m structure even exceeds the leading CMAR_10 structure in the final months of 2018. Other structures are weaker than the CMAR_10 structure and fluctuate greatly, indicating that data preprocessing and the adoption of the multi-head attention network can improve the effectiveness of deep reinforcement learning.

In Fig. 10, the CMAR_10 structure performs well and maintains a leading position but still fluctuates greatly. The CMAR_1_m structure and MAR_1_m structure are still robust, although the compound interest is weaker than that in the CMAR_10 structure. The MAR_1_m structure performed poorly in 2018, even worse than the benchmark multiple times. Because the MAR_1_m structure does not preprocess the data, the noise of minute data affects the decision-making ability of the structure. The performance of other structures is still weaker than that of the CMAR_10 structure. This result shows the steady improvement brought by data preprocessing and network changes again.

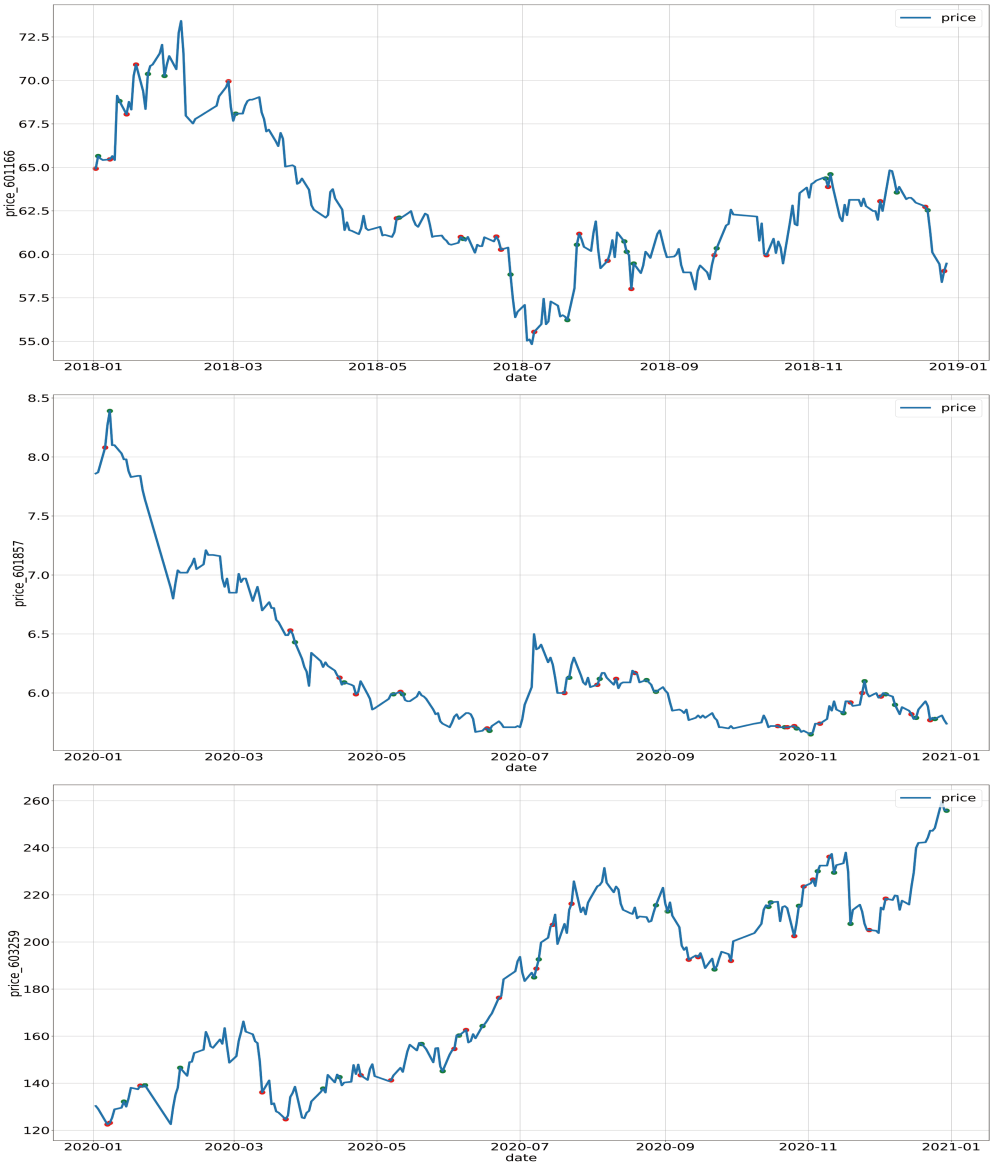


Fig. 8. Analysis of trading point.

In Fig. 11, the CMAR_1_m structure and MAR_1_m structure perform well, and the trend is steady. Although the CMAR_10 structure occasionally exceeds the CMAR_1_m structure, the CMAR_1_m structure is ahead of the CMAR_10 structure most of the time. Similarly, the performances of other structures are still more volatile and weaker than that of the CMAR_10 structure.

3.6 Statistical analysis of backtest results

To compare the performances of the above different structures accurately, Sharpe ratio, max drawdown, and relevant statistical indicators of the portfolio's simple interest are introduced. The max drawdown is calculated as follows:

$$\mathrm{MD} = \min_{i=1,\cdots,n}\left\{\frac{\mathrm{CI}_i}{\max\limits_{j=1,\cdots,i}\mathrm{CI}_j} - 1\right\},$$

where CI_i represents the compound interest value of the portfolio on the ith trading day, and n represents the backtest days.

The Sharpe ratio can measure the relationship between portfolio risk and return, that is, how much excess return will be generated for each unit of risk. Hence, the larger the value, the better the performance. Max drawdown reflects the decline of the portfolio compound interest; thus, this indicator is expected to be small. The statistical parameters in this paper are simplified and explained in Table 1. Statistical value analyses are shown in Tables 2–7.

Analyzing the six tables, we found that the CMAR_1_m structure and MAR_1_m structure perform well regardless of the Sharpe ratio or max drawdown, especially the CMAR_1_m structure. The CMAR_1_m structure can be robust when the compound interest value is high. The CMAR_10 structure has a considerable compound interest value but fluctuates greatly. In addition, comparing different structures using the same investment strategy, it can be confirmed that data processing and the network change improve the effect. However, the impact of the network change is

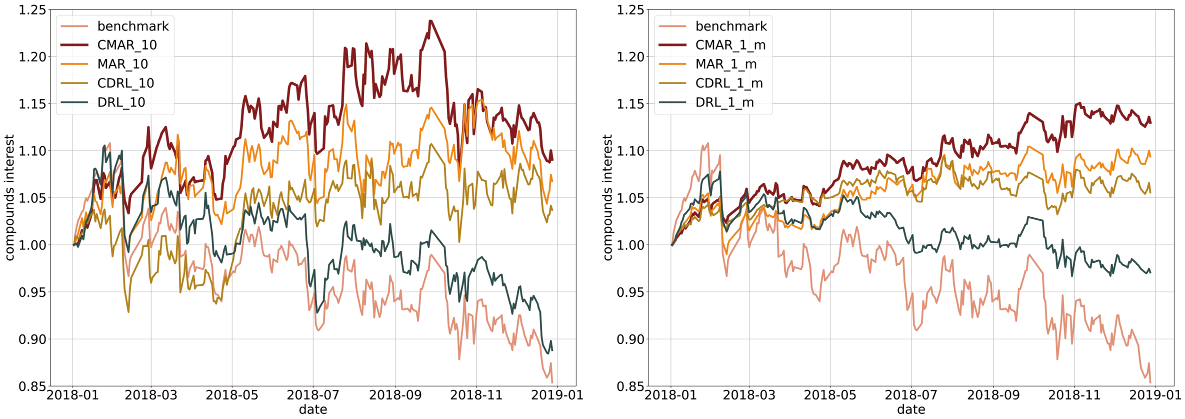


Fig. 9. Compound interest curve in 2018. The left subgraph represents the result of the first investment strategy, and the right subgraph represents the result of another investment strategy.

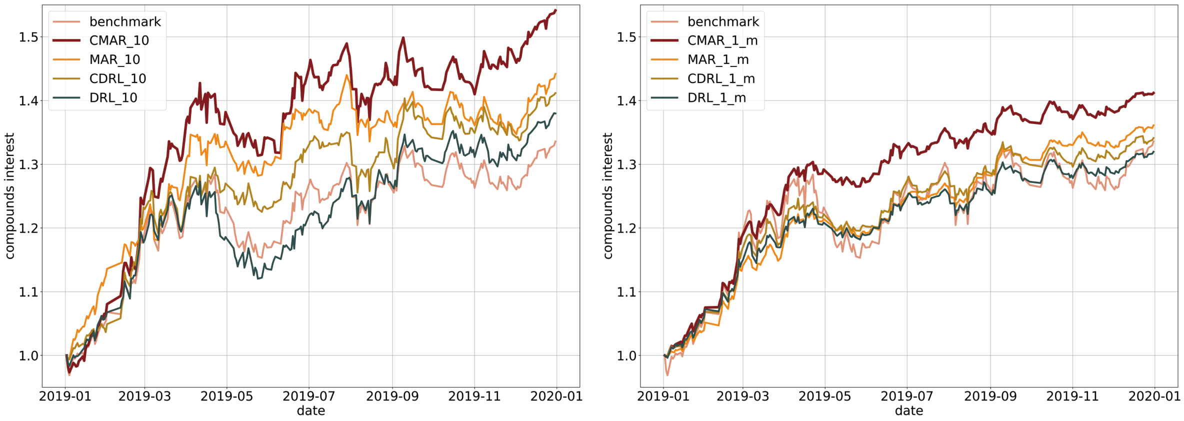


Fig. 10. Compound interest curve in 2019. The left subgraph represents the result of the first investment strategy, and the right subgraph represents the result of another investment strategy.

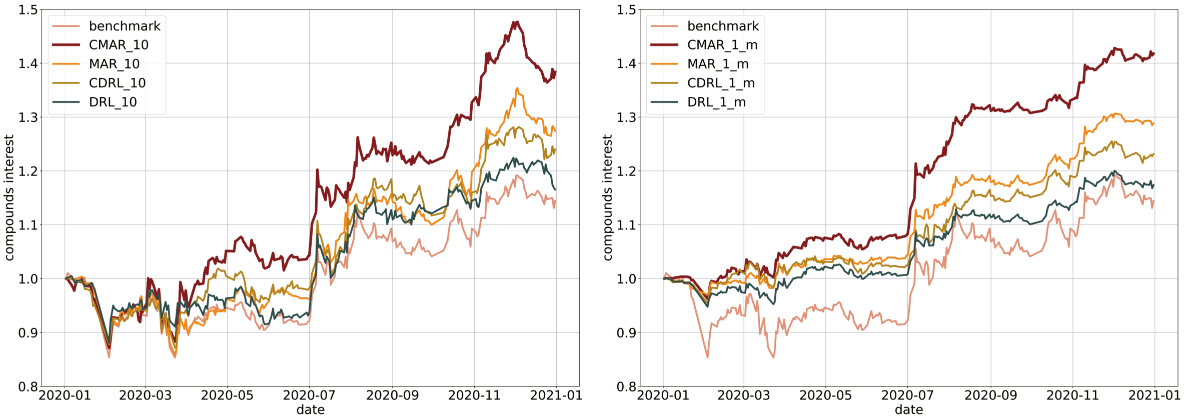


Fig. 11. Compound interest curve in 2020. The left subgraph represents the result of the first investment strategy, and the right subgraph represents the result of another investment strategy.

greater. Comparing Tables 2 and 3, it can be found that the Sharpe ratio of the CMAR_1_m structure is 2.85 times higher than that of the CMAR_10 structure. Moreover, the compound interest is also slightly higher. Comparing different investment strategies with the same structure, the second investment strategy is significantly better in terms of the max drawdown, while the first investment strategy is generally better in statistical indicators of simple interest. Comparing Tables 4 and 5, it is shown that the first investment strategy has great advantages in compound interest and various statistical indicators of simple interest. However, in terms of the Sharpe ratio and max drawdown, the effect of the second investment strategy is still robust. In addition, the compound interest results in the four structures are still considerable. Observing

Table 1. Descriptions of abbreviations of statistical parameters.

Abbreviation	Description
CI	Compound interest value of structure
SR	Sharpe ratio of structure
MD	max drawdown of structure
Mean of dr	Mean of simple interest of structure
Std of dr	Standard deviation of simple interest of structure
Min of dr	minimum of simple interest of structure
Qn of dr	Quartile n of simple interest of structure
Med of dr	Median of simple interest of structure
Max of dr	median of simple interest of structure

Table 2. Backtest results of the first strategy in 2018.

2018	benchmark	CMAR_10	MAR_10	CDRL_10	DRL_10
CI	0.8538	1.0907	1.0676	1.0370	1.0000
SR	−0.5790	0.5609	0.4503	0.2900	0.1046
MD	−0.2294	−0.1336	−0.0957	−0.1050	−0.1256
Mean of dr	−0.0005	0.0004	0.0004	0.0002	0.0001
Std of dr	0.0142	0.0134	0.0131	0.0140	0.0130
Min of dr	−0.0531	−0.0407	−0.0412	−0.0498	−0.0708
Q1 of dr	−0.0089	−0.0079	−0.0077	−0.0079	−0.0075
Med of dr	0.0002	0.0006	0.0007	0.0005	−0.0001
Q3 of dr	0.0078	0.0084	0.0078	0.0086	0.0081
Max of dr	0.0463	0.0457	0.0482	0.0437	0.0349

Table 3. Backtest results of the second strategy in 2018.

2018	benchmark	CMAR_1_m	MAR_1_m	CDRL_1_m	DRL_1_m
CI	0.8538	1.1298	1.0937	1.0555	0.9706
SR	−0.5790	1.6010	1.0398	0.7624	−0.2508
MD	-0.2294	−0.0360	−0.0612	−0.0561	−0.1032
Mean of dr	−0.0005	0.0005	0.0004	0.0002	−0.0001
Std of dr	0.0142	0.0055	0.0063	0.0051	0.0064
Min of dr	−0.0531	−0.0144	−0.0240	−0.0147	−0.0250
Q1 of dr	−0.0089	−0.0029	−0.0032	−0.0029	−0.0033
Med of dr	0.0002	0.0001	0.0001	0.0004	−0.0002
Q3 of dr	0.0078	0.0033	0.0042	0.0030	0.0036
Max of dr	0.0463	0.0199	0.0200	0.0154	0.0201

Tables 6 and 7, we can see that the CMAR_1_m structure and CMAR_10 structure are much better than the benchmark in various indicators. The effect of various structures using the second investment strategy is still stable. The indicators of the first investment strategy show that it is a high-yield and high-risk strategy.

4 Conclusions

In this paper, we advocate for the CEEMDAN_Multi_Att_RL structure, in which the CEEMDAN method, multi-head attention network, and reinforcement learning are combined. Applying the CEEMDAN method in decomposing and denoising stock high-frequency data made the state space more conducive to machine learning. In the meantime, the processed data can better simulate the real stock market environment. Multi-head attention was introduced into reinforcement learning, as this network made the model focus on some key parts of the state space and improved the learning ability of reinforcement learning. In addition, this paper explored two investment strategies with this structure. The two strategies had advantages and disadvantages; accordingly,

Table 4. Backtest results of the first strategy in 2019.

2019	benchmark	CMAR_10	MAR_10	CDRL_10	DRL_10
CI	1.3357	1.5413	1.4418	1.4116	1.3796
SR	2.0358	3.1193	3.1507	2.6594	2.4129
MD	−0.1019	−0.0827	−0.0724	−0.0702	−0.1200
Mean of dr	0.0013	0.0019	0.0016	0.0015	0.0014
Std of dr	0.0115	0.0120	0.0096	0.0107	0.0109
Min of dr	−0.0346	−0.0363	−0.0282	−0.0316	−0.0298
Q1 of dr	−0.0051	−0.0043	−0.0033	−0.0043	−0.0053
Med of dr	0.0013	0.0009	0.0011	0.0008	0.0020
Q3 of dr	0.0076	0.0081	0.0072	0.0063	0.0073
Max of dr	0.0450	0.0521	0.0358	0.0432	0.0392

Table 5. Backtest results of the second strategy in 2019.

2019	benchmark	CMAR_1_m	MAR_1_m	CDRL_1_m	DRL_1_m
CI	1.3357	1.4118	1.3610	1.3411	1.3196
SR	2.0358	4.2549	4.1825	3.2599	3.4278
MD	−0.1019	−0.0295	−0.0317	−0.0421	−0.0394
Mean of dr	0.0013	0.0014	0.0013	0.0012	0.0012
Std of dr	0.0115	0.0065	0.0058	0.0070	0.0062
Min of dr	−0.0346	−0.0134	−0.0132	−0.0196	−0.0151
Q1 of dr	−0.0051	−0.0023	−0.0019	−0.0023	−0.0027
Med of dr	0.0013	0.0007	0.0009	0.0005	0.0005
Q3 of dr	0.0076	0.0039	0.0041	0.0044	0.0042
Max of dr	0.0450	0.0330	0.0282	0.0326	0.0231

Table 6. Backtest results of the first strategy in 2020.

2020	benchmark	CMAR_10	MAR_10	CDRL_10	DRL_10
CI	1.1444	1.3836	1.2743	1.2397	1.1653
SR	0.7393	1.7615	1.4447	1.2661	0.8979
MD	−0.1553	−0.1300	−0.1443	−0.1286	−0.1235
Mean of dr	0.0007	0.0015	0.0011	0.0010	0.0007
Std of dr	0.0161	0.0160	0.0139	0.0140	0.0143
Min of dr	−0.1138	−0.0948	−0.0930	−0.0781	−0.1025
Q1 of dr	−0.0065	−0.0060	−0.0057	−0.0059	−0.0058
Med of dr	−0.0005	0.0001	−0.0001	0.0003	−0.0005
Q3 of dr	0.0083	0.0081	0.0074	0.0072	0.0070
Max of dr	0.0662	0.0706	0.0511	0.0719	0.0744

investors could choose them according to their risk preferences. Finally, the position adjustment strategy of the model was only carried out at the opening or closing of each trading day, and this time point was favorable for investors to perform corresponding operations.

We only considered investment strategies for investors of two different risk preference types in this study. Later, to achieve stability and profitability of the model, we will consider predicting different market circumstances, thereby dynamically adjusting the two investment strategies. The prediction of market circumstances is the greatest challenge we currently face, as it requires the model to perform well in selecting the trading time point. For China’s stock market, it is possible to predict the market from traditional indicators, such as the decline of the three major stock market indices (Shanghai Stock Exchange Index, Shenzhen Stock Exchange Index, and ChiNext Index), as well as the consumer price index (CPI) and producer price index (PPI) to make adjustments for investment strategy.

Table 7. Backtest results of the second strategy in 2020.

2020	benchmark	CMAR_1_m	MAR_1_m	CDRL_1_m	DRL_1_m
CI	1.1444	1.4176	1.2883	1.2313	1.1735
SR	0.7393	3.6530	2.9738	2.1343	1.6955
MD	−0.1553	−0.0413	−0.0301	−0.0461	−0.0528
Mean of dr	0.0007	0.0015	0.0011	0.0009	0.0007
Std of dr	0.0161	0.0077	0.0065	0.0074	0.0070
Min of dr	−0.1138	−0.0353	−0.0169	−0.0351	−0.0394
Q1 of dr	−0.0065	−0.0021	−0.0022	−0.0027	−0.0030
Med of dr	−0.0005	0.0006	0.0002	0.0005	−0.0001
Q3 of dr	0.0083	0.0041	0.0036	0.0038	0.0033
Max of dr	0.0662	0.0560	0.0375	0.0344	0.0384

Acknowledgements

This work was supported by the National Nature Science Foundation of China (71871208, 71971204) and the Anhui Provincial Natural Science Foundation (1908085MG236).

Conflict of interest

The authors declare that they have no conflict of interest.

Biographies

Weiwei Zhuang received her Ph.D. degree in Probability and Statistics from the University of Science and Technology of China (USTC) in 2006. She is an Associate Professor with the Department of Statistics and Finance, USTC. Her research interests include statistical dependence, stochastic comparisons, semiparametric model, and their applications.

Guoxin Qiu received his Ph.D. degree in Statistics from the University of Science and Technology of China (USTC) in 2017. He is currently a Professor with the Business School, Anhui Xinhua University. His research interests include information theory, stochastic comparisons, semiparametric model, and their applications.

References

[1] Neuneier R. Optimal asset allocation using adaptive dynamic programming. In: Proceedings of the 8th International Conference on Neural Information Processing Systems. New York: ACM, **1995**: 952–958.

[2] Nevmyvaka Y, Feng Y, Kearns M. Reinforcement learning for optimized trade execution. In: ICML '06: Proceedings of the 23rd International Conference on Machine Learning. New York: ACM Press, **2006**: 673–680.

[3] Meng T L, Khushi M. Reinforcement learning in financial markets. *Data,* **2019**, *4*: 110.

[4] Liu X, Xiong Z, Zhong S, et al. Practical deep reinforcement learning approach for stock trading. **2022**. https://arxiv.org/abs/1811.07522. Accessed April 1, 2022.

[5] Brim A. Deep reinforcement learning pairs trading with a double deep Q-network. In: 2020 10th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, **2020**: 222–227.

[6] Gao Z, Gao Y, Hu Y, et al. Application of deep Q-network in portfolio management. In: 2020 5th IEEE International Conference on Big Data Analytics (ICBDA). IEEE, **2020**: 268–275.

[7] Lee J, Koh H, Choe H J. Learning to trade in financial time series using high-frequency through wavelet transformation and deep reinforcement learning. *Applied Intelligence,* **2021**, *51*: 6202–6223.

[8] Carta S, Corriga A, Ferreira A, et al. A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. *Applied Intelligence,* **2021**, *51*: 889–905.

[9] Théate T, Ernst D. An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications,* **2021**, *173*: 114632.

[10] Lei K, Zhang B, Li Y, et al. Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. *Expert Systems with Applications,* **2020**, *140*: 112872.

[11] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. n: Advances in Neural Information Processing Systems. Red Hook, NY: Curran Associates Inc., **2017**: 6000–6010.

[12] Huang N E, Shen Z, Long S R, et al. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London Series A: Mathematical, Physical and Engineering Sciences,* **1998**, *454*: 903–995.

[13] Torres M E, Colominas M A, Schlotthauer G, et al. A complete ensemble empirical mode decomposition with adaptive noise. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Prague, Czech Republic: IEEE, **2011**: 4144–4147.

[14] Sutton R S, Barto A G. Reinforcement Learning: An Introduction. Cambridge, Massachusetts: The MIT Press, **2018**.

[15] Bellman R. Dynamic Programming. Princeton: Princeton University Press, **1972**.

DOI: 10.52396/JUSTC-2022-0072