Applied Intelligence
https://doi.org/10.1007/s10489-022-03606-0

Dynamic stock-decision ensemble strategy based on deep reinforcement learning

Xiaoming Yu[1] · Wenjun Wu[1] · Xingchuang Liao[1] · Yong Han[1]

Accepted: 9 April 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract

In a complex and changeable stock market, it is very important to design a trading agent that can benefit investors. In this paper, we propose two stock trading decision-making methods. First, we propose a nested reinforcement learning (Nested RL) method based on three deep reinforcement learning models (the Advantage Actor Critic, Deep Deterministic Policy Gradient, and Soft Actor Critic models) that adopts an integration strategy by nesting reinforcement learning on the basic decision-maker. Thus, this strategy can dynamically select agents according to the current situation to generate trading decisions made under different market environments. Second, to inherit the advantages of three basic decision-makers, we consider confidence and propose a weight random selection with confidence (WRSC) strategy. In this way, investors can gain more profits by integrating the advantages of all agents. All the algorithms are validated for the U.S., Japanese and British stocks and evaluated by different performance indicators. The experimental results show that the annualized return, cumulative return, and Sharpe ratio values of our ensemble strategy are higher than those of the baselines, which indicates that our nested RL and WRSC methods can assist investors in their portfolio management with more profits under the same level of investment risk.

Keywords Investment market · Stock trading · Deep reinforcement learning · Real-time decision-making

1 Introduction

Investing is a means to save money from extra income and idle funds, resulting in more compensation and rewards in the future. Investing undoubtedly increases one's source of income and improves one's personal quality of life. Warren Buffett [45], a famous investor, defines investment as "A process of laying out money now in the expectation of receiving more money in the future." Indeed, successful investing can increase one's finances through a variety of investment tools. To reduce the risks of the investment process, one must weigh and allocate one's money considering a variety of factors. Generally, diversification is considered a safer way to invest one's money in multiple assets rather than a single asset. As the saying goes, "Don't put all your eggs in one basket."

In terms of diversified investment, stock investment is considered to be the most difficult. The stock market is a highly complex and nonlinear dynamic ecosystem composed of market participants who can make decisions freely based on their individual beliefs and personal profits. Many factors affect the stock market, such as political turmoil, news events, public sentiment, and exchange rate fluctuations. Due to the instability and extremely unpredictable features of the stock market, stock decision-making is also affected by various and conflicting attributes, resulting in a typical multiattribute decision-making (MADM) problem [33]. In view of the existence of various factors in the stock market, rational portfolio management is our main goal.

Portfolio management is a continuous process that maximizes accumulated profits by minimizing the overall risk of the portfolio and involves position sizing and resource

✉ Xiaoming Yu
yuxiaoming@buaa.edu.cn

Wenjun Wu
wwj09315@buaa.edu.cn

Xingchuang Liao
liaoxingchuang@buaa.edu.cn

Yong Han
hanyong@buaa.edu.cn

[1] State Key Lab of Software Development Environment, Beihang University, Beijing 100191, China

Published online: 09 May 2022

allocation [1]. Professional investment analysts and retail investors often make stock trading decisions based on their personal experience and views. However, the efficiency of such portfolio management is extremely low in a complex and risky stock market. Some portfolio results recommended by traditional investment analysts present several limitations [34]. Traditional investment analysts fail to serve a large number of low net worth customers [35]. At the same time, they are more vulnerable to behavioral biases and conflicts of interest. However, AI-based investment has the advantages of low thresholds, low costs, and high efficiency [36, 37] and revises recommendations more often than human analysts. Moreover, Coleman et al. [38] provided the first comprehensive comparison of the investment recommendations generated by AI-based and human analysts. Their results suggest that AI-based portfolio systems outperform human analysts and are a valuable, alternative information intermediary to traditional sell-side analysts for investment decisions. In conclusion, the AI-based portfolio can surpass experienced human traders in financial markets [39]. Therefore, it is necessary to develop an AI-based portfolio stock trading strategy that can assist stock investors in coping with a variety of dynamic environments to maximize the expected return and minimize investment risk.

AI-based portfolio management systems can provide financial services and investment consulting for users by adopting easy-to-use and low-cost algorithms [2]. At the same time, the application of artificial intelligence algorithms can balance the risk and return of investment, which optimizes the portfolio to a great extent [8]. Markowitz proposed the standard mean-variance (MV) model to solve the multiobjective optimization problem in portfolio management [3]. In the MV model, portfolio optimization is regarded as the objective function, and the average return of assets is modeled as one of the constraints. Due to the cardinality and boundary constraints, the computational overhead of the MV model is very high, thus limiting its applicability. Based on the classical MV model, Strumberger et al. extended the formula to solve constrained combination problems by combining the bat algorithm with the artificial bee colony heuristic algorithm [4], which is a hard optimization problem suitable for stochastic optimization metaheuristics. Furthermore, to improve the dispersion of investment, Slimane et al. [5] proposed two mean-semientropy portfolio selection models and designed a fuzzy simulation-based genetic algorithm to solve the models to optimality. Recently, Leung et al. [40] formulated the classic Markowitz mean-variance (MV) framework and its variant mean conditional value-at-risk (CVaR) as minimax and biobjective portfolio selection problems and then applied neurodynamic approaches to solve these problems.

In a dynamic stock market environment, the nonlinearity of the time series is prominent and affects the efficacy of stock price forecasts. Thus, Chou et al. [7] designed an intelligent time series prediction system to improve investment profits. In addition, stock prices sometimes represent a similar pattern and are determined by multiple factors. Chou et al. [9] proposed a new and complex method to find similar patterns in historical stock data to obtain daily stock prices with high prediction accuracy and potential rules. Furthermore, some new models such as augmented fuzzy rough neural networks (FRNNs) [41], prediction models based on clustering [6], etc. have gradually been proposed to predict complex stock time series. Among the factors affecting stock prices, the behavior of investors plays a very important role. Therefore, Wang et al. [42] explored the impact of investors' social networks on stock price dynamics. In addition, in current approaches to predicting stock prices, the relationships between stocks and sectors are often neglected. To study this issue, Hsu et al. proposed a novel model, Financial Graph Attention Networks (FinGATs) [43], to recommend the top-k stocks in terms of return ratios using time series of stock prices and sector information.

Recently, reinforcement learning has been widely applied in a variety of fields of decision-making, such as for self-driving [10, 11], medical care [12], robot control [13], and games [14]. The stock trading process can be regarded as an online decision-making process occurring in response to market fluctuations. Agents of reinforcement learning can decide which strategy to use to obtain as many rewards as possible. Accordingly, trading strategies also need to determine which operations (such as buying, selling, and holding) to use to gain more profits in stock trading. Therefore, reinforcement learning seems to be a very good choice for learning optimal stock trading strategies [15]. Given the nonlinear, noisy, and unstable nature of the stock market, it is difficult for decision-making agents to achieve optimal results. To address this challenge, a deep neural network is successfully integrated into reinforcement learning [16, 17] because deep reinforcement learning (DRL) can abstract the characteristics of data from complex nonlinear original data. Due to the fluctuations of stock data, more novel input features are considered in the deep learning model, which will improve the performance of the prediction model [18]. Portfolios, asset allocation, and trading systems could be better optimized by deep reinforcement learning [19].

To design a stock decision-making agent, we need to determine the most suitable deep reinforcement learning model. Because each model has its own advantages and disadvantages, we propose ensemble strategies of stock trading. Our contributions are as follows:

1. We explore benefits of basic reinforcement learning models that are adept at stock trading, laying a foundation for the ensemble strategy.

2. We select the three models Advantage Actor Critic (A2C), Deep Deterministic Policy Gradient (DDPG), and Soft Actor Critic (SAC) in the FinRL library [20] as our basic decision-makers. Based on these models, we propose a nested RL architecture that can dynamically and spontaneously select the active decision-maker and generate the optimal trading strategy for complex and dynamic stock markets. In addition, we compare our approach to several common ensemble algorithms.
3. To obtain a higher return, we consider the confidence factor and propose a weighted random selection with confidence (WRSC) algorithm, which follows the strategy of the strongest decision-maker with high confidence; otherwise, the decision-maker makes a weighted random selection from the remaining base decision-makers according to the annual return rate.
4. We demonstrate our ensemble strategies for U.S., Japanese and British stocks and validate case studies of real-world trading scenarios involving CSCO stocks, showing that our strategies exhibit excellent performance.

The rest of the paper is organized as follows. Section 2 introduces recent progress made in intelligent stock trading algorithms based on deep reinforcement learning and compares existing methods to our approach. Section 3 presents the problem formulation for RL-based stock trading under the MDP framework. Section 4 describes our stock trading methods based on reinforcement learning in more detail. Section 5 presents performance evaluations of our methods and baseline algorithms. In Section 6, we summarize the paper and describe our plans for future work.

2 Related work

The latest development of reinforcement learning and deep learning has introduced new ideas to quantitative trading on the stock market. Recently, deep reinforcement learning has been regarded as an effective method in the field of quantitative finance. For example, Liu et al. introduced a DRL library FinRL [20] that allows users to simplify their own development and compare it to existing schemes. Moreover, Qlib, a new AI-oriented quantitative investment platform, has been proposed [21] and enables the easy exploration of AI technology in quantitative investment. Qlib not only provides high-performance infrastructure but also integrates a number of machine learning tools for quantitative investment scenarios. Both methods lay a solid foundation for us to investigate the possible adoption of reinforcement learning in stock investment.

Following the paradigm of reinforcement learning, many methods for RL-based quantitative stock trading have been proposed in two fields, including the stock environment and portfolio strategies.

In terms of the stock environment, recent studies focus on a single stock investment. One new approach is called the adaptive stock trading strategy based on deep reinforcement learning [22], which uses the gated recurrent unit (GRU) to extract financial feature information to reflect the internal characteristics of the stock market and make adaptive trading decisions. Through the customized design of state and behavior space, researchers have proposed two kinds of trading strategies based on the reinforcement learning Gated Deep Q-learning trading strategy (GDQN) and Gated Deterministic Policy Gradient trading strategy (GDPG) [22]. GDQN and GDPG trading strategies perform well in stock markets, but they only concentrate on single stock investment without considering portfolio management. Because the profit margin of a single stock investment is often limited, its risk is relatively high. To solve this problem, our nested RL method analyzes the market of multiple stocks (this study uses 90 stocks) and constructs an intelligent stock trading strategy to build a flexible portfolio of multiple stocks.

On the other hand, there are representative studies on portfolio strategy. Recently, Li et al. proposed a novel Adaptive Deep Deterministic Policy Gradient scheme (ADDPG) for the portfolio allocation task [23]. The model can distinguish positive and negative feedback and dynamically adjust the learning rate of the Q function in the DDPG according to the prediction error. Despite the ADDPG's improvement of the DDPG, it is still only suitable for steady stock markets and unable to deliver accurate decisions for a more volatile environment. To alleviate this limitation and realize the scalability of decision-making agents, we propose nested RL, which combines multiple RL algorithms to generate more flexible and optimized strategies in a complex and dynamic stock market.

Some research efforts have already explored ensemble reinforcement learning approaches that combine multiple RL methods to exceed the performance of a single method. To apply an ensemble strategy to continuous spaces, Rohan Saphal et al. [24] proposed SEERL, which combines diverse policies, including both discrete and continuous space strategies. The latter continuous space strategy is the equivalent of majority voting in continuous action space. Nevertheless, this voting strategy cannot deal with stock actions with time series and thus only considers current stock prices and ignores historical ones.

Remedies to such a problem have been proposed to carry out automated stocking. One pioneering work was done by Salvatore et al. [25], who proposed a multilayer and multiensemble stock trader to address the issue of using price information in single supervised classifiers leading to poor results. Then, Carta et al. proposed the multi-DQN

method [46], which combines deep Q-learning classifiers that can address the uncertain and chaotic behavior of different stock markets. However, the learning and convergence speed of the multi-DQN method slow as the amount of stock data increases. Our DRL shows great promise in dealing with complex, multifaceted, and sequential decision-making problems. In addition, Yang et al. proposed an automated stock trading (AUST) ensemble strategy based on three RL algorithms: Proximal Policy Optimization (PPO), the Advantage Actor Critic (A2C) and the Deep Deterministic Policy Gradient (DDPG) [26]. The method automatically selects the best agent of the three algorithms to make trading decisions according to the Sharpe ratio, which can adapt to different market environments. Nevertheless, this strategy only makes a greedy selection in n sliding windows of three months. The result only inherits the advantages of each algorithm and does not exceed the best performance of all three agents. Studies have shown that some RL agents make more assertive decisions, whereas other RL agents tend to be more pessimistic in response to the dynamic stock market. To comprehensively combine the strengths of each agent, we add the corresponding weight to each agent and propose the weighted random selection with confidence (WRSC) algorithm.

3 Problem formulation

Some unpredictable facets of the stock market can affect yields, but with a clear understanding of the market, one can make decisions at the best trading time. Stock trading refers to buying and selling the shares of a specific company. If one owns stock, one owns part of a company. The most commonly used stock market terms include buying, selling, holding, closing, the trading volume, the bear market, the bull market, dividends, etc. In a bear market, the stock market shows a downward trend where the prices of multiple stocks are falling. In a bull market, the stock market exhibits an upward trend where the prices of multiple stocks are increasing. Stock prices are divided into the opening price, closing price, highest price, and lowest price.

In the stock market, stock trading involves a stochastic and interactive process; thus, stock trading decisions can be modeled as the Markov decision process (MDP). During the MDP, decision-makers observe Markov stochastic dynamic systems periodically or continuously and make decisions sequentially. The MDP is a quad $\{S, A, P, R\}$. Here, S is a set of finite states, A is a set of finite actions, P is the state transition probability, and R is the expected immediate reward received after performing action A. In this section, we define the state space, action space, reward function and environment of this MDP framework. We use the following indicators to represent the state space of the stock trading environment.

3.1 State Space

Stock investors analyze all kinds of stock information before making decisions (buy, sell, or hold). To learn from the environment, our trading agent needs to observe many different characteristics. The state space describes the observation results obtained by interacting with the environment.

- Balance $b \in R_+$: the total amount remaining in the user's account in time step t.
- Shares own $h \in Z_+^n$: the current shares for each stock, where n represents the number of stocks.
- Closing price $p \in R_+^n$: the closing price of the stock market, which is the weighted average trading volume of all transactions one minute before the last trading of the securities on a given day.
- Opening price $o \in R_+^n$: the price at which a security first trades on the opening of an exchange on a trading day.

  High price $h \in R_+^n$: the highest price among the trading prices on a given day.

  Low price $l \in R_+^n$: the lowest price among the trading prices on a given day. The three prices reflect the changes in stock prices.
- Trading volume $v \in R_+^n$: the total number of stocks traded by investors in a period of time.

3.2 Action Space

Action space refers to the allowed actions of the trading agent interacting with the stock market environment. Generally, $a \in A$ includes three actions: $\{-1, 0, 1\}$, where -1, 0, and 1 respectively denote selling, holding and buying a stock. A single action can be used in multiple stocks. We define action space $\{-k....., -1, 0, 1.....k\}$, where k represents the number of stocks. For example, when one buys 20 NKE stocks or sells 20 NKE stocks (k=20), the corresponding action is denoted as 20 or - 20, respectively. As shown in Fig. 1, the total value of an investor's stock is 'v' at time t. After taking different actions (buy, hold or sell), the corresponding total value of the stock will change and eventually become 'total value 1,' 'total value 2,' or 'total value 3' at $t + 1$. Since stock trading occurs daily and stock decision-making occurs in real time, we believe that a daily decision-making frequency can effectively measure the performance of the model. That is, after obtaining the stock information of the current day, our model gives stock decision suggestions for the next day.

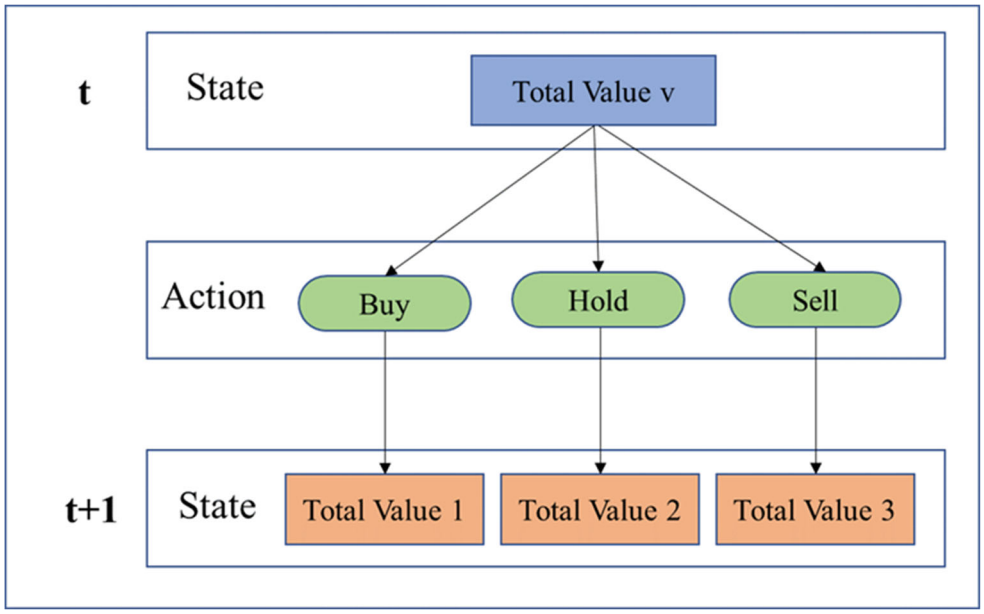

Fig. 1 The total value of an investor's stock changes after 3 different actions (buy, hold, and sell)

3.3 Reward function

Reward function $R(s, a, s')$ is an incentive mechanism that encourages trading agents to identify better behavior strategies. Here, we provide the commonly used reward function template [27] as follows:

- In state s, when action a reaches new state s', the change in the portfolio value is $R(s, a, s') = V' - V$, where V' and V represent the portfolio value in s' and s, respectively.
- In state s, action a is taken and reaches new state s', the log value of the portfolio change ratio:

$$r(s, a, s') = log\left(\frac{v'}{v}\right) \tag{1}$$

- During trading period t, the Sharpe ratio is defined as follows:

$$S_T = \frac{mean(R_t)}{std(R_t)} \tag{2}$$

where $R_t = V_t - V_{t-1}$, $t \in [1, ..., T]$. V_t is portfolio's current value at t.

4 Methodology

The complexity of the stock market presents volatility, vulnerability, and uncertainty. Deep reinforcement learning agents can make a dynamic adjustment at any time according to changes in the environment, which can be successfully applied to stock decision-making. We choose the DRL models (the A2C, DDPG, and SAC) in the FinRL library as basic decision-makers. As data grow exponentially, tagging large datasets becomes time-consuming and strenuous. However, DRL does not use large label training datasets, which is a key advantage. The purpose of stock trading is to maximize returns while avoiding risks. To achieve this, DRL maximizes the total expected returns through trading actions.

4.1 Selection of the Base Decision-maker

The actor critic approach has been recently applied in designing reinforcement-based stock trading systems. Its main purpose is to simultaneously update an actor network that represents the policy and an opposite critic network that represents the value function. The actor critic approach has proven to be able to learn and adapt to large and complex environments. Thus, the actor critic approach performs well in trading with a large stock portfolio. We adopt the following three models (the A2C, DDPG, and SAC) as our basic decision-makers, each of which shows its own advantages in different stock markets.

These three models are chosen as basic decision-makers because they offer their own advantages and can provide dynamic decisions that conform to different trading environments. First, one of the A2C's [28] major advantages lies in its capacity to manage large collections of complex stock data and support multiple stock trading scenarios: single-stock trading, multiple-stock trading and portfolio allocation. Second, portfolio management is a process of continuously changing the distribution weight of funds in financial assets. The DDPG [29] is capable of handling high-dimensional continuous action spaces and can learn continuously. Therefore, the DDPG is an ideal candidate

that can automatically adjust the weight of stocks in each trading period to find the optimal decision-making action. Third, the SAC [30] is suitable for significant changes in stock environments because it adopts a random strategy with certain advantages over the deterministic strategy. Next, we introduce the structures and principles of the three models in more detail.

4.1.1 Advantage Actor Critic (A2C)

In the original actor critic approach, the Q value output of the critic network is used to calculate the policy gradient. However, this method produces noise and high levels of variance. To address this issue, Wu et al. [44] proposed subtracting a baseline from the cumulative reward $Q(s_t, a_t)$ (stock return) when calculating the expectation. This method can reduce the gradient such that the step of gradient descent is gentler and makes the training process more stable. This also helps the A2C to construct a loss function. Based on this idea, the advantage function is constructed as follows:

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t) \tag{3}$$

where $Q(s_t, a_t)$ is the value after executing decision action a_t (buy, sell, hold) in state s_t or the asset return value. $V(s_t)$ is the state value function. Therefore, the loss function of the A2C is as follows:

$$\nabla J_\theta(\theta) = E[\sum_{t=1}^{T} \nabla_\theta log\pi_\theta(a_t|s_t)A(s_t, a_t)] \tag{4}$$

where $\pi_\theta(a_t|s_t)$ is a policy network representing the probability of selecting action a_t in state s_t, and θ is the parameter to be updated. $A(a_t|s_t)$ is the advantage function of (3).

As shown in Fig. 2, the A2C uses multiple agents working in parallel to update gradients $\nabla\theta$ with different data samples. Each agent works independently to interact with the same stock environment to obtain independent sampling experience, and these experiences are also independent of each other, which breaks the coupling between experiences and has the same effect as experience replay. After all parallel agents complete the gradient calculation, the A2C uses the coordinator to transfer the average gradient on all agents to the global network. In this way, the global network can update the network of actors and critics.

The global network increases the diversity of training data. Synchronous gradient updating is more cost effective and efficient and has a better effect in large batches. At the same time, in view of the stability and robustness of the A2C, it is an ideal model for stock trading. Based on these advantages of the A2C, we choose it as our basic decision-maker.

4.1.2 Deep deterministic policy gradient (DDPG)

The DDPG is a policy learning method that integrates deep learning neural networks into the deterministic policy gradient (DPG). Inspired by the deep Q network (DQN), Lillicrap et al. improved the DPG and used the convolutional neural network as policy function μ and Q function simulation. Then, the deep learning method is used to train these neural networks so that large-scale states and action space can be learned online.

The agent of the DDPG takes action a_t (buy, sell, or hold) in state s_t and obtains reward value r_t (the return value of stock assets) when it reaches new state s_{t+1}. As shown in Fig. 3, the DDPG actor first stores the transition data (s_t, a_t, s_{t+1}, r_t) into experience replay buffer R and then randomly samples the mini-batch N data from experience replay buffer R during training. The DDPG uses a function approximator parameterized by θ^Q to update the critic network by minimizing the following losses:

$$L = \frac{1}{N}\sum_t (y_t - Q(s_t, a_t|\theta^Q))^2 \tag{5}$$

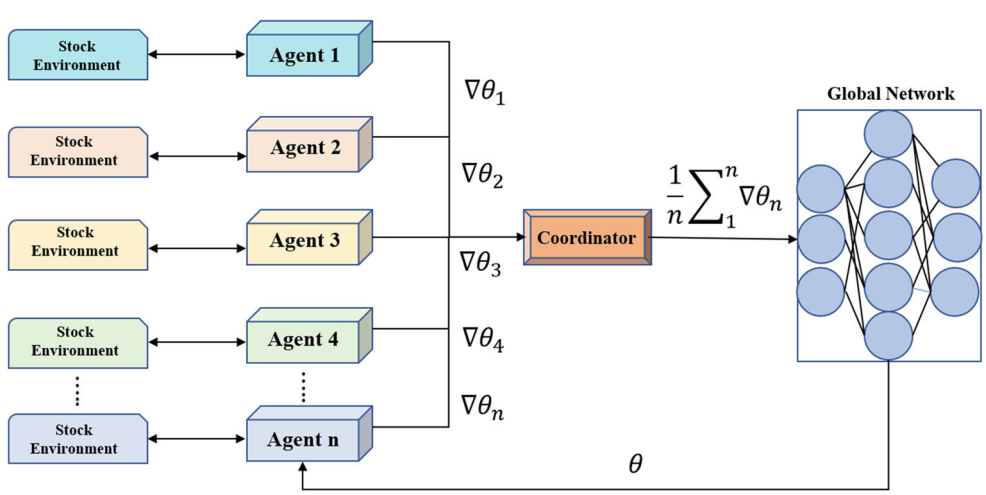

Fig. 2 A2C model framework in the stock environment

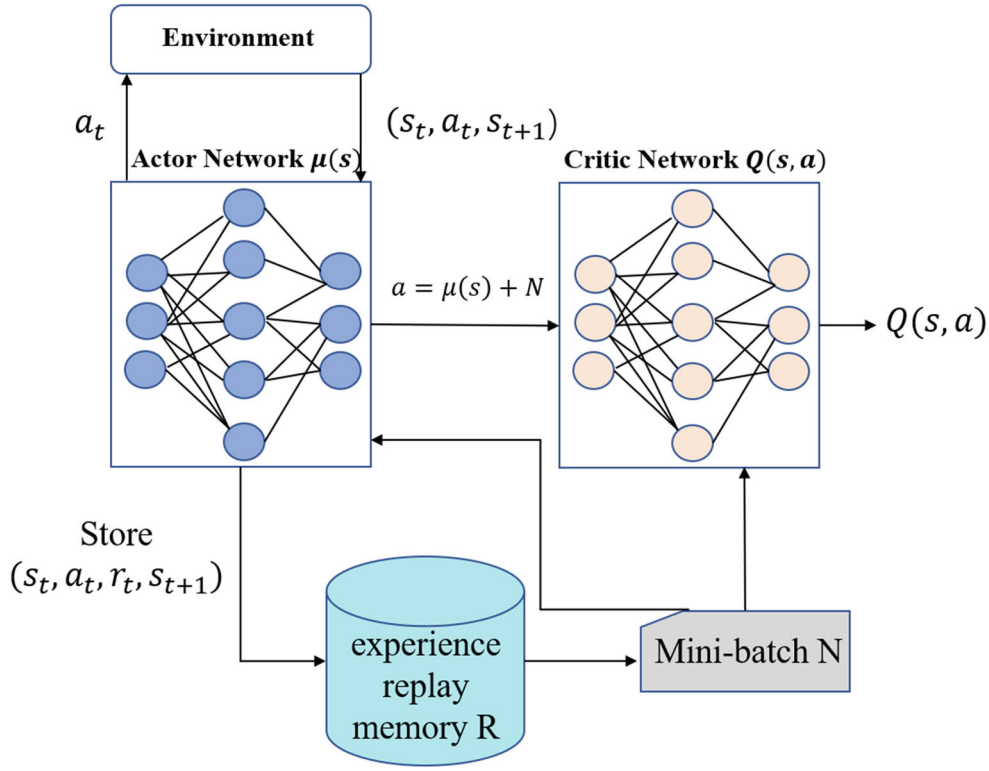

Fig. 3 Framework of the DDPG model

where $Q(s_t, a_t|\theta^Q)$ is an action value function describing the expected return after taking action a_t in state s_t. y_t is obtained by the following formula:

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})|\theta^Q) \tag{6}$$

where $\gamma \in [0, 1]$is the discount factor. Then, we update the actor policy using the sampled policy gradient:

$$\begin{aligned}\nabla_{\theta^\mu} J &\approx E_{s_t \sim \rho^\beta}[\nabla_{\theta^\mu} Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t|\theta^\mu)}] \\ &= E_{s_t \sim \rho^\beta}[\nabla_a Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta_\mu} \mu(s|\theta^\mu)|_{s=s_t}]\end{aligned} \tag{7}$$

where ρ^β denotes the discounted state visitation distribution for different stochastic behavior policy β.

For the target networks, Lillicrap et al. used "soft" updates rather than directly copying the weights. The authors created copies of the actor and critic networks, $Q'(s, a|\theta^{Q'})$ and $\mu'(s, a|\theta^{\mu'})$, respectively, used to calculate the target values.

$$\begin{aligned}\theta^{Q'} &\leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \tau \ll 1 \\ \theta^{\mu'} &\leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}, \tau \ll 1\end{aligned} \tag{8}$$

The weights of the target networks are then updated by slowly tracing the learned networks, which means that the target values can only change slowly, significantly improving the stability of learning. This is the critical reason for the construction of the target network.

The DDPG can better deal with high-dimensional continuous action space, so it can be effectively applied for stock trading. We thus to choose it as the basic decision-maker.

4.1.3 Soft Actor Critic (SAC)

The Soft Actor Critic (SAC) [30] is an off-policy algorithm developed for maximum entropy reinforcement learning. Compared to the DDPG, the SAC uses stochastic policy, which has certain advantages over deterministic policy. The SAC requires the actor to maximize the entropy of reward expectation and strategy distribution at the same time. The introduction of maximum entropy enhances action exploration ability, enabling the exploration of more stock decisions and achieving more stable performance under complex circumstances.

The iterative process of the SAC is divided into soft policy evaluation and soft policy updating. For fixed strategy π, its soft Q value can be iterated by Bellman backup operator T^π:

$$T^\pi Q(s_t, a_t) \triangleq r(s_t, a_t) + V E_{s_{t+1} \sim p}[V(S_{t+1})] \tag{9}$$

$$V(s_t) = E_{a_t \sim \pi}[Q(s_t, a_t) - log\pi(a_t|s_t)] \tag{10}$$

where T^n is the Bellman backup operator, and $Q^{k+1} = T^\pi Q^k$. In practice, tractable policies are preferred. Thus, we additionally restrict the policy to set of policies II that can

correspond to a parameterized family of distributions such as Gaussians. The soft policy is updated as follows:

$$\pi_{new} = argmin_{\pi' \in II} D_{KL}(\pi'(\cdot|s_t)||\frac{exp(Q^{\pi_{old}(s_t,\cdot)})}{Z^{\pi_{old}}(s_t)}) \quad (11)$$

where $Z^{\pi_{old}}(s_t)$ is the partition function used to normalize the distribution of Q values. Different from the usual off-policy method used to maximize the Q value, the policy of the SAC is updated in the direction of an exponential distribution proportional to Q. In practice, to facilitate the processing of the policy, we still output the policy as a Gaussian distribution and minimize the gap between the two distributions by minimizing KL divergence.

By using soft policy evaluation and soft policy updating repeatedly and alternately, the final policy will converge to the optimal value. The learning objective of the SAC is as follows:

$$\pi^* = argmax_{\pi} \sum E_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t) + \alpha H(\pi(\cdot|s_t))] \quad (12)$$

where hyperparameter α measures the relative importance of entropy for reward. The randomness of the optimal control policy controlled by α is determined by the following formula:

$$\alpha_t^* = argmin_{a_t} E_{a_t\sim\pi_t^*}[-\alpha_t log\pi_t^*(a_t|s_t;\alpha_t) - \alpha_t\overline{H}] \quad (13)$$

Relative to the deterministic policy, the stochastic policy of the SAC also requires entropy maximization, which means that the neural network needs to explore all possible optimal paths, which can produce the following advantages. 1) The policy will learn many ways to complete tasks through maximum entropy, which is more conducive to learning new tasks. 2) Clearly, the policy's stronger exploration ability makes it easier to find better modes under multimodal rewards. For example, stock decision-making agents should not only obtain high returns but also reduce trading risks. 3) The policy is more robust and generalizable by exploring various optimal possibilities in different ways, so it is easier to adjust in the face of interference. For example, when facing different stock markets, agents can make different decisions in dealing with different environments.

4.2 Proposed strategies

4.2.1 Nested RL trading strategy

Based on the significant application advantages of the three DRL models (the A2C, DDPG, and SAC) mentioned in Section 4.1 in reference to stock trading, we adopt them as our basic decision-makers and combine them to integrate the advantages of the three agents and obtain higher returns with minimal risk. It is critical to select an agent that behaves the best from the A2C, DDPG and SAC according to annualized returns. Furthermore, choosing a suitable agent from the three agents as the final decision-maker in different trading environments is a major research problem. In view of the effective application of reinforcement learning in decision-making problems, we design two-layered reinforcement learning for the three agents and propose a nested reinforcement learning (Nested RL) framework including A2C RL, DDPG RL and SAC RL. At the second layer of nested RL, the three agents attempt to learn their own trading strategies independently and present their recommendations, while at the first layer, a primary agent aims to learn a selection strategy of determining which recommendations to adopt. Fig. 4 displays the nested RL frameworks (A2C RL, DDPG RL and SAC RL) on which the primary agent acts based on the three base decision-makers.

The first layer of nested A2C/DDPG/SAC RL contains five elements: A2C-1/DDPG-1/SAC-1 agent, Action, Reward, State and Environment on the left side of Fig. 4. Here, the action space of the primary agent involves choosing the A2C-2, DDPG-2 and SAC-2 strategies, where three actions follow function $G(a, S)$ in (14).

$$G(a, S) = \begin{cases} A2C(S), & if \quad a \in [a_0, a_1] \\ DDPG(S), & if \quad a \in [a_1, a_2] \\ SAC(S), & if \quad a \in [a_2, a_3] \end{cases} \quad (14)$$

where the value range of action a is $[a_0, a_3]$, and a_1 and a_2 are thresholds. The value range of action a represents the annualized return obtained when three agents make decisions. The greater annualized return is, the greater the value range of action a becomes.

These three actions indicate which recommendation Nested RL uses to make a decision. For instance, the Nested A2C RL's agent (the A2C-1) selects an action from the DDPG-2 agent that acts as a basic decision-maker to carry out a stock trading strategy. Similarly, the DDPG-2 also contains five elements: the DDPG-2 agent, action, reward, state and environment. Upon receiving the DDPG-2 agent's action, the environment's state immediately changes and sends a reward signal to the DDPG-2 agent as its feedback. Afterward, the DDPG-2 agent makes decisions according to reward and feedback signals and gives trading recommendations to the A2C-1 agent. As a result, Nested A2C RL will eventually follow the decision made by the DDPG-2 agent. The DDPG's environmental factors include opening/high/low prices, closing prices, trading volume and balance. Its reward is the annualized return. Figure 4 shows the overall process: the first layer selects a basic decision-maker, and the second layer follows this decision-maker to buy/sell/hold stocks. The pseudocode of Nested RL is listed in Algorithm 1. First, Nested RL obtains the state

representation of environment S on Lines 1-2. Then, on Line 3, our method obtains the original action and S' through the actor network. Lines 4-5 map the original action to the base decision-maker's action. On Line 6, our algorithm obtains reward R of the mapped base decision-maker's action through the actor network. On Lines 7-8, the algorithm computes the TD loss to further update the network. On Lines 9-10, Nested RL updates critic network parameters. Lines 11-12 of this algorithm run gradient descent to update the actor network parameters. On Line 13, the algorithm saves the 5-tuple $(\phi(S), sub_{action}, R, \phi(S'), is_{end})$ into the experience pool. Finally, the base decision-makers are trained by experience pool D in every time period m on Line 14.

Algorithm 1 Nested RL.

Require: Number of iterations T; Base learning algorithm ξ;
Ensure: $action$;
1: for $t = 0$ to T do;
2: S = Transform To State(T_t);
3: in the Actor network to obtain the original action and the next S': action, $S' = \phi(S)$;
4: Map the original action to the sub action for the base decision-makers:
5: $sub_{action} = \begin{cases} A2C(S), & if \quad action \in [a_0, a_1] \\ DDPG(S), & if \quad action \in [a_1, a_2] \\ SAC(S), & if \quad action \in [a_2, a_3] \end{cases}$
6: Compute reward R in the actor network: $R = GetRewardInActorNetwork(sub_{action}, S)$;
7: In the critic network, use S and S' to get $V(S)$, $V(S')$;
8: Compute the TD loss: $\delta = R + \gamma V(S') - V(S)$;
9: Use the Mean square error loss function to update critic network parameter ω:
10: $\sum(R + \gamma V(S') - V(S))^2$
11: To update parameter θ, apply the following:
12: $\theta = \theta + \alpha \nabla_\theta log \pi_\theta(S, A)\delta$;
13: put $(\phi(S), sub_{action}, R, \phi(S'), is_{end})$ in experience replay D;
14: Use D to update base decision-makers ξ for every time period m;
15: **return** $action$;

4.2.2 WRSC trading strategy

Our nested RL approach is designed to achieve the 'maximal annualized return for different stock market environments by dynamically fusing trading strategies provided by different RL agents. In addition to the layered RL framework defined by Nested RL, we also explore the combined Weighted Random Strategy with Confidence (WRSC) method to balance the likelihood of strong agents and weak agents being selected. This method selects the optimal trading strategy from the A2C, DDPG, and SAC based on their weight and confidence. Figure 5 illustrates the computation workflow of WRSC. First, WRSC runs the three agents to calculate the annualized return of stocks as AR(A2C), AR(DDPG) and AR(SAC). Then, it selects an agent's strategy with the maximal annualized return and its confidence among the three candidates. Here, confidence refers to the probability that an agent will make a certain decision. When the agent's confidence is greater than a threshold, WRSC complies with actions made by the current agent; otherwise, WRSC randomly selects the remaining two agents according to their respective weights as the current decision-maker and follows this agent's actions.The pseudocode of our WRSC strategy is shown in Algorithm 2.

Algorithm 2 Weighted random strategy with confidence (WRSC).

Require: Dataset T; Base learning algorithm ξ; Weighted matrix ω;
1: $T1$(Training), $T2$(Validation) = SpiltAccordingtoTime (T);
2: best ξ = max Annual(ξ, $T1$);
3: **for** $t = 0$ to $T2$, **do**
4: S = Transform To State($T2_t$)
5: Use the best base decision-maker to get $action$, R, S'
6: S'=best ξ. GetReward(S, $action$)
7: confidence = best ξ. getConfidence($action$, $T2_t$)
8: **if** (confidence <threshold) **then**
9: $action$ = Randomly Choose one of the other algorithms according to their confidence
10: **end if**
11: put $(\phi(S), action, R, \phi(S'), is_{end})$ into experience replay D
12: **end for**
13: **return** $action$

In Algorithm 2, T is first divided into training set $T1$ and validation set $T2$ according to time in step 1. On Line 2, WRSC obtains the best strategy from the base decision-maker by $T1$. Then, on Lines 3-5, the algorithm obtains S, S', R, the $action$, the action value of the best base decision-maker. On Line 6, WRSC computes the confidence (probability) of this action and time step. On Lines 7-9, the algorithm determines whether to select the base agent's strategies by comparing their confidence values against a predefined threshold. On Lines 10-11, WRSC continues to train base decision-makers using the experience

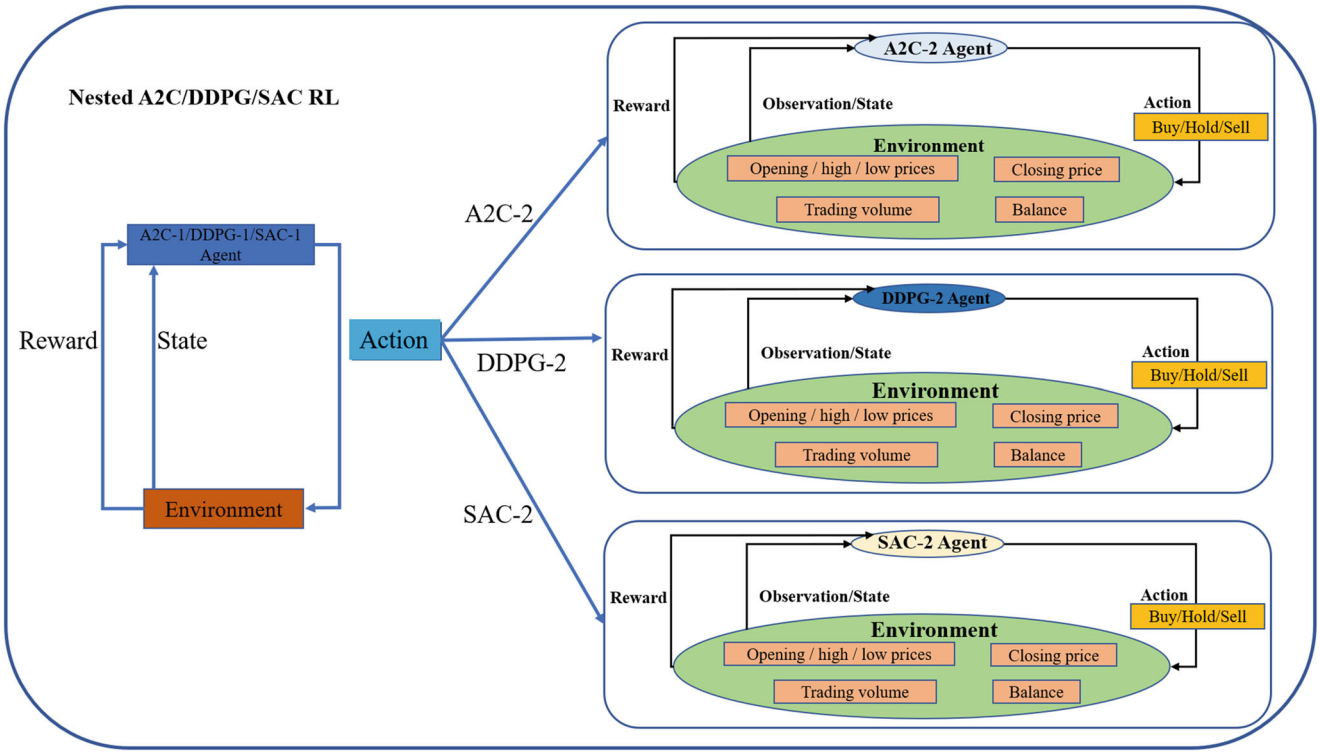

Fig. 4 Framework of nested A2C/DDPG/SAC RL

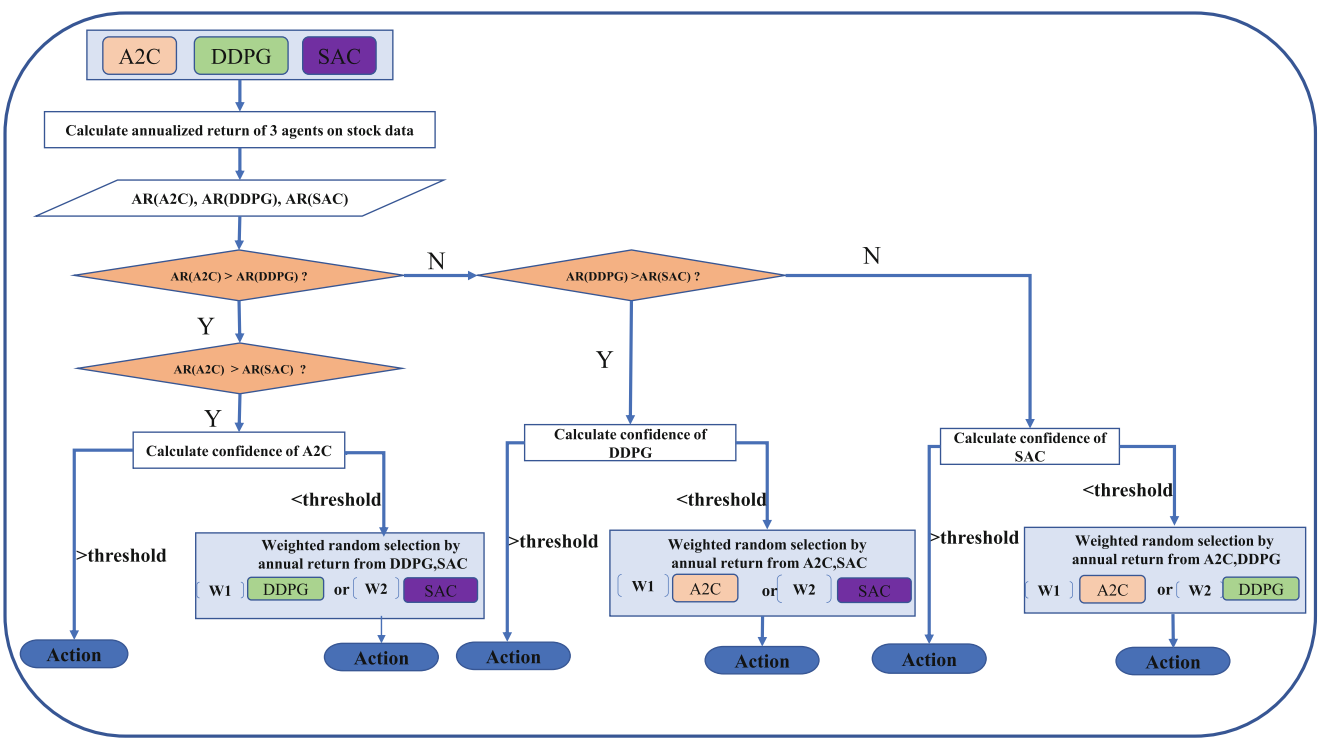

Fig. 5 Workflow of the Weighted Random Strategy with Confidence (WRSC) strategy

replay mechanism, where $\phi(S)$ is the function used to decode the state.

5 Performance evaluation

5.1 Datasets and experimental environment

We first explain the selection of stock data used for our trading strategy. As is well known, 30 Dow Jones stocks cover representative companies from many different industries, mainly including financial services, pharmaceutical industry, information technology, etc., and can reflect the state of the U.S. stock market to a certain extent. Therefore, these data are useful to train the robustness, effectiveness, and universality of our proposed model. The Dow Jones industrial average (DJIA) is also calculated based on the reputation, market value, and several other features of these 30 stocks. It is considered an indicator of the overall health of the market and is one of the most popular stock market indices. Therefore, these 30 Dow Jones stocks are very suitable for the training and test trading of our proposed strategy.

In addition to 30 U.S. stocks, we also choose 30 Japanese and British stock data for our experiments to verify and train the generality and applicability of the model. The company names of the stocks are shown in Table 1. Along the timeline of the original datasets, we further partition the data samples for 2000/01/01 to 2015/01/01 as a training set and those for 2015/01/01 to 2021/01/01 as a validation/trading set as shown in Fig. 6.

In actual trading scenarios, an intelligent trading agent needs to consider all kinds of relevant information, such as historical stock prices, current holding shares, and technical indicators. In this paper, our trading environment is established based on the OpenAI Gym framework, and according to the principle of time-driven simulation, we run the collected real data to simulate the stock market. Here, we adopt the FinRL library, which can simulate trading environments across various stock markets.

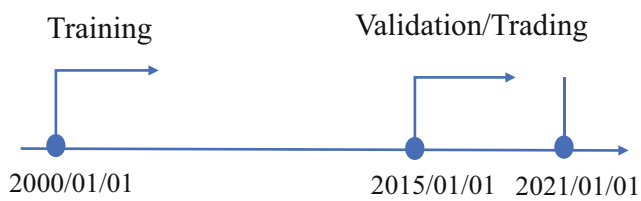

Fig. 6 Data splitting of stocks on the U.S./Japanese/British markets

5.2 Performance comparison

5.2.1 Evaluation indicator

1. Annualized return

Since investors pay more attention to final investment benefits no matter which operation is taken for each stock, we use the measurement index - annualized return - to measure the performance of stock trading strategies. The annualized return refers to rate of return obtained by investors during a one-year investment period. The calculation formula is as follows:

$$Annualized\ return| = \frac{Return \cdot 365}{Principal \cdot Investment\ days} \cdot 100\% \tag{15}$$

2. Cumulative return

The cumulative return of an investment is the total amount of the investment gained or lost over time, regardless of the time involved. The cumulative return is expressed as a percentage and can be calculated by following function:

$$Cumulative\ return = \frac{Current\ Values - Original\ Values}{Original\ Values} \tag{16}$$

3. Annualized volatility

Annualized volatility is the annualized standard deviation of portfolio return.

Table 1 The company name abbreviation of 30 stocks selected in the U.S., Japanese, and British markets respectively

Market	Company Symbol
The U.S. stock (30 stocks)	AXP, AMGN, AAPL, BA, CAT, CSCO, CVX, GS, HD, HON, IBM, INTC, JNJ, KO, JPM, MCD, MMM, MRK, MSFT, NKE, PG, TRV, UNH, CRM, VZ, V, WBA, WMT, DIS, DOW
The Japanese stock (30 stocks)	Advantest Corp, Alps Electric, Amada, Chiba Bank, Chugai Pharmaceutical, Concordia Financial Group, Dainippon Screen Mfg., DIC Corp, Eneos Holdings Inc, Fujikura Ltd., GS Yuasa Corp., Hitachi Zosen Corp., J.Front Retailing Co., Ltd., Japan Steel Works Ltd, JR West Japan, KDDI, Keisei Electric Railway Co.,Ltd., Konami Corp., Maruha Nichiro Corp, Minebea Mitsumi Inc, NEC, Tokyo Electric Power, Casio, Mazda, Hitachi, Mitsui chemical, Mitsubishi Motors, Sony, Isuzu Motors Ltd., Yamaha
The British stock (30 stocks)	ABDN, EDEV, BGUK, BLND, CRH, ENT, FLTRF, HIK, HLMA, ICP, JETJ, MRON, NWG, OCDO, POLYP, PSHP, RMV, RTO, SGE, SMDS, SMT, AZN, CPG, RR, LSEG, SSE, TW, NG, BP, BATS

4. Sharpe ratio

In finance, the Sharpe ratio [32] measures the performance of an investment compared to a risk-free asset, after adjusting for its risk. It is defined as the difference between the returns of the investment and the risk-free return, divided by the standard deviation of the investment. Its calculation of the Sharpe ratio is as follows:

$$S_a = \frac{E(R_a - R_b)}{\sigma_a} = \frac{E(R_a - R_b)}{\sqrt{var[R_a - R_b]}} \tag{17}$$

where R_a is the asset return and R_b is the risk-free return (such as a U.S. Treasury security). $E[R_a - R_b]$ is the expected value of the excess of the asset return over the benchmark return, and σ_a is the standard deviation of the asset excess return.

5. Max drawdown

A maximum drawdown (MDD) is the maximum observed loss from a peak to a trough of a portfolio, before a new peak is attained. Maximum drawdown is an indicator of downside risk over a specified time period.The formula for maximum drawdown is as follows:

$$MDD = \frac{Trough\ Value - Peak\ Value}{Peak\ Value} \tag{18}$$

5.2.2 Benchmark Comparison

In this paper, we use five baselines to compete against our approach:

Multi-DQN This method is an ensemble of the same deep Q-learning classifiers with different experiences with the environment. It can tackle the uncertain and chaotic behavior of different stock markets through a flexible ensemble strategy.

AUST This is an ensemble strategy based on deep reinforcement learning designed as an automated stock trading (AUST) method [26]. The strategy automatically selects the best agent from three models to trade according to the Sharpe ratio.

The other three methods are the average weighted (Average-W), weighted by return (Weight-R), and weighted-random by return (RandomWeight-R) strategies.

Average-W This method simply assigns the same weight to the three agents (the A2C, DDPG, and SAC) to make a final decision.

Weight-R This approach gives greater weight to agents that can produce higher annualized returns and will follow the agents with the greatest weight.

RandomWeight-R This method mainly randomly selects agents to make decisions by annualized returns of stocks.

The premise of the latter three baselines are close to the majority voting method that combines the best action of each algorithm and determines its final decision on the frequency by which an action is preferred by each algorithm. Such a combination of basic decision-makers can yield a refined model of high accuracy and robustness.

The two methods proposed in this paper are nested RL (A2C RL, DDPG RL and SAC RL) and the weighted random selection with confidence (WRSC) strategy. The experimental results of the algorithms are shown in Tables 2, 3, and 4. The four evaluation indicators used in our paper are objective and not data related and do not depend on the time, country, etc. considered. Our nested RL and WRSC decision models are universal and robust, which is only related to the parameters of the model itself, the number of training times, the basic decision-makers, etc.

Tables 2 and 3 clearly show that the three performance indicators (the annual return, cumulative returns and Sharpe ratio) of A2L RL are higher than the RandomWeight-R, Weight-R, Average-W, Multi-DQN and AUST values, which shows that A2C RL gains more in the U.S. and Japanese markets. Tables 2 and 4 clearly show that the three performance indicators of DDPG RL are higher than the RandomWeight-R, Weight-R, Average-W, Multi-DQN and AUST values, which shows that DDPG RL behaves better in the U.S. and British markets. We also find from Tables 2, 3, and 4 that the three indicators of WRSC are greater than those of the traditional three methods, Multi-DQN and AUST, indicating that WRSC can achieve high returns in three different markets.

The low performance of the RandomWeight-R, Weight-R and Average-W approaches is caused by the limitation of simple ensemble approaches such as the weighted average and majority voting methods, which cannot be well applied to the long-term decision-making stock environment. These three traditional methods only select the basic decision-maker according to the weight of the stock return. When other relevant factors need to be considered, including technical indicators such as annual volatility and maximum drawdown (MDD), the traditional methods may not be able to deal with this information well. However, our proposed method can dynamically select the basic decision-maker that is most in line with the current state in real time according to changes in technical indicators in the trading environment. Our method reduces trading risk as much as possible while considering a high return. Note that the Multi-DQN achieves a very low annualized return because the model gradually performs poorly with an increase in stock data. The AUST [26] strategy does not achieve the same level of annual return as our methods because it only makes a greedy selection among the models in a sliding

Table 2 Index values of Nested RL, WRSC and other baselines on 30 U.S. stocks

2015/01/01-2021/01/01	WRSC (ours)	Nested RL(ours)			Multi-DQN [46]	AUST [26]	Random Weight-R	Weight-R	Average-W
		A2C RL	DDPG RL	SAC RL					
Annual return	**16.01%**	12.27%	9.23%	7.57%	0.326%	8.51%	6.36%	8.77%	6.28%
Cumulative return	**145.36%**	100.69%	70.15%	55.11%	2.54%	63.43%	44.95%	65.86%	44.24%
Sharpe ratio	**1.02**	0.81	0.68	0.58	0.12	0.58	0.43	0.66	0.44
Annual volatility	15.8%	15.76%	14.46%	14.24%	15.23%	16.79%	17.79%	14.22%	16.97%
Max drawdown	-17.76%	-27.09%	-20.04%	-18.08%	-28.96%	-25.2%	-32.33%	-19.79%	-28.22%

Table 3 Index values of Nested RL, WRSC and other baselines on 30 Japanese stocks

2015/01/01-2021/01/01	WRSC (ours)	Nested RL(ours)			Multi-DQN [46]	AUST [26]	Random Weight-R	Weight-R	Average-W
		A2C RL	DDPG RL	SAC RL					
Annual return	0.64%	**3.15%**	-1.98%	-1.17%	-6.18%	-2.67%	-1.90%	-4.72%	-3.56%
Cumulative return	3.76%	**19.71%**	-10.99%	-6.62%	-33.36%	-14.53%	-10.57%	-24.48%	-18.99%
Sharpe ratio	0.149	**0.294**	-0.404	-0.342	-0.652	-0.316	-0.583	0.27	-0.538
Annual volatility	5.19%	45.7%	4.68%	3.28%	10.15%	7.6%	3.21%	69.85%	6.36%
Max drawdown	-6.72%	-37.5%	-10.99%	-11.8%	-34.69%	-19.7%	-16.97%	-62.11%	-21.4%

Table 4 Index values of Nested RL, WRSC and other baselines on 30 British stocks

2015/01/01-2021/01/01	WRSC (ours)	Nested RL(ours)			Multi-DQN [46]	AUST [26]	Random Weight-R	Weight-R	Average-W
		A2C RL	DDPG RL	SAC RL					
Annual return	25.32%	23.11%	**26.28%**	22.81%	9.42%	25.65%	17.77%	25.21%	14.42%
Cumulative return	289.53%	249.87%	**307.77%**	244.82%	84.74%	295.74%	167.79%	287.36%	125.16%
Sharpe ratio	0.65	0.62	**0.66**	0.61	0.51	0.65	0.54	0.65	0.50
Annual volatility	62.47%	62.11%	62.80%	60.17%	63.15%	62.93%	59.73%	59.87%	55.48%
Max drawdown	-44.23%	-43.40%	-42.14%	-43.51%	-51.67%	-43.14%	-42.55%	-40.03%	-39.65%

window of three months and does not exceed the best performance of all three agents. Moreover, AUST may not be able to dynamically select a reasonable agent when the stock market changes. In contrast, our nested reinforcement learning methods (the A2C RL and DDPG RL) are able to match the dynamics of the stock environment in real time, thereby achieving higher returns.

Furthermore, it is clear from Table 2 (U.S. stocks) that WRSC's performance exceeds that of Nested RL. The reason is that the majority of Nested RL decisions tend to choose the strongest agent with the best chance of winning the highest return and only select the weak agent with the lowest return in very few cases. The WRSC approach is designed to balance the possibility of a strong agent and weak agent being selected by combining weight random selection with confidence. Our WRSC algorithm follows a decision made by the strongest agent when its confidence is higher than the predefined threshold; otherwise, we randomly select the remaining agents by the weight of annual return. In this way, WRSC is able to locate more trading opportunities for profits.

From the perspective of the max drawdown index, the value of WRSC (-44.23%) in Table 4 is lower than the values of all other methods, which indicates that our approach has a smaller maximum loss when their annual volatility is roughly the same. Overall, the experimental results confirm that our method can generate a high-performance trading strategy.

Finally, to analyze the overall trend of the methods more clearly, we plot curves of different cumulative return changes of four baselines and our two methods for 2015/01/01-2020/12/30 in Figs. 7, 8, and 9:

The cumulative return curve in Fig. 7 shows that the three traditional methods, Multi-DQN and AUST strategy have always fallen behind A2C RL, and the gap with WRSC is becoming increasingly larger, which indicates that the application of deep reinforcement learning as a stock integration strategy can indeed improve investment return. It is evident that Nested A2C RL complies with a decision of the optimal base agent in most cases, and it can also adaptively choose the remaining decision-makers in other cases. Such behavior resembles the premise of WRSC, thereby demonstrating its best performance and robustness among the Nested RL methods proposed in this paper. Therefore, the overall performance of Nested A2C RL is the best and most robust of the three Nested RL methods. Furthermore, the performance and trend of WRSC and A2C RL are consistent because they share similar ensemble ideas. In some cases, WRSC sacrifices a small amount of stability in exchange for higher profits. In general, WRSC is more suitable for pursuing high profits, whereas Nested RL is more suitable for the common scenario of low risk tolerance.

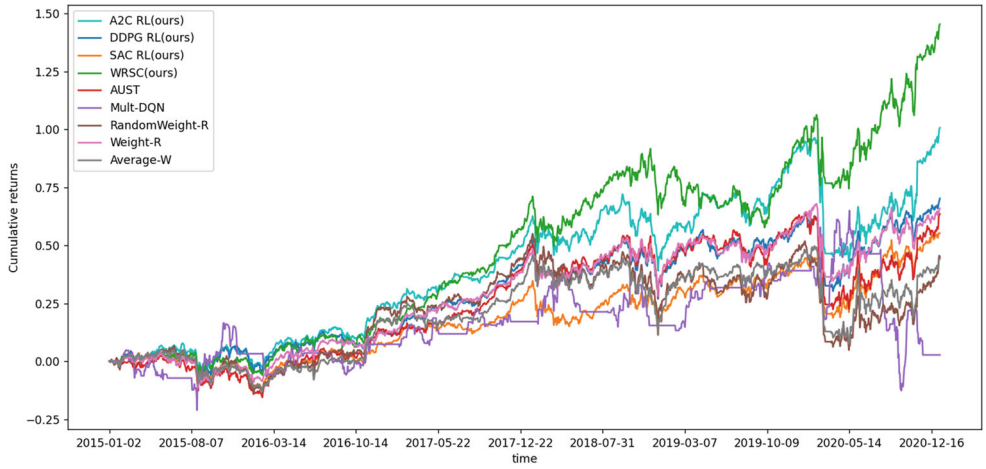

Fig. 7 Trends of cumulative returns of nested RL, WRSC and baselines for 2015/01/01-2020/12/30 on 30 U.S. stocks

As shown in Fig. 8, the Average-W/RandomWeight-R/AUST approaches are almost always in a loss state. Although Weight-R is profitable at first, it enters a loss state over time. In contrast, our A2C RL approach finally achieves high returns. Fig. 9 shows that the traditional Average-W/Randomweight-R curve is always at the bottom. Over time, our DDPG RL approach leads and achieves the largest annualized return.

The stock market is considered a risky prospect, as shares bought can increase or decrease in value for various reasons. We also observe that all the methods show a downward trend during 2020/02, 2020/03 and 2020/10 in Fig. 7, as the global stock market, including U.S. shares, was affected by the U.S. election and COVID-19, resulting in a negative trend and a sharp drop in the stock market. Therefore, the cumulative return obtained by our approaches and other baselines also declines or even has a low value for the above period of time. Similarly, the Japanese and British markets shown in Figs. 8 and 9 also show fluctuations.

To compare the performance of our ensemble strategy (Nested A2C RL and WRSC) with the three independent agents (the A2C, DDPG, and SAC) employed as the trading agent separately, we plot their dynamic curves on cumulative returns for U.S. stocks. Fig. 10 clearly demonstrates that the cumulative returns of our strategies gradually exceed those of the three independent agents. This further emphasizes that it is significant to adopt an ensemble strategy in stock decision-making.

In summary, the WRSC and nested RL methods proposed in this paper can generate an excellent stock trading strategy. When investment risk is similar, our integrated strategy can dynamically make appropriate decisions and gain high profits with a low loss.

5.2.3 Case Study

To verify the effectiveness of our proposed method for real trading, we compare the decision-making processes of the AUST and A2C RL strategies for U.S. stock CSCOs. Figs. 11 and 12 show the decision-making processes for CSCO stock for 2020/09/30-2020/12/30, respectively, where the vertical axis denotes the stock price. The blue

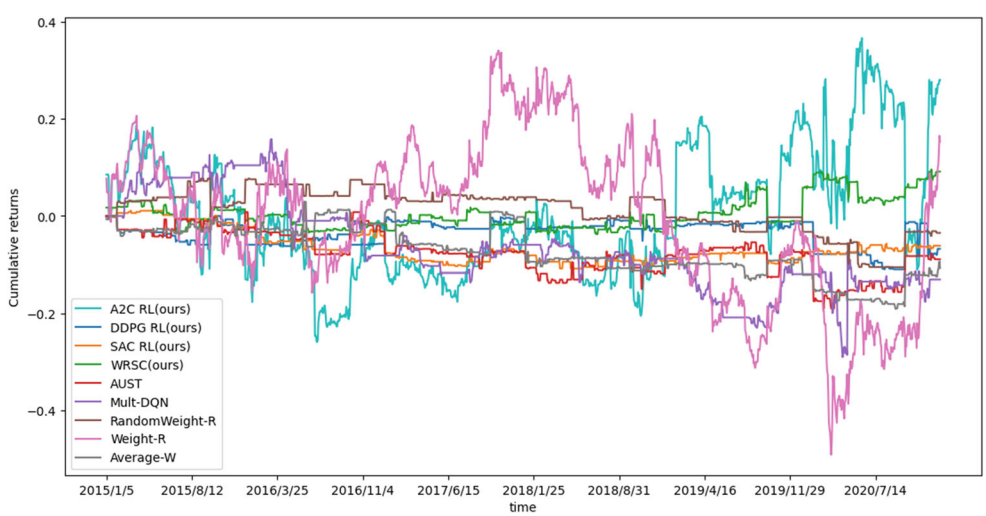

Fig. 8 Trends of cumulative returns of nested RL, WRSC and baselines for 2015/01/01-2020/12/30 on 30 Japanese stocks

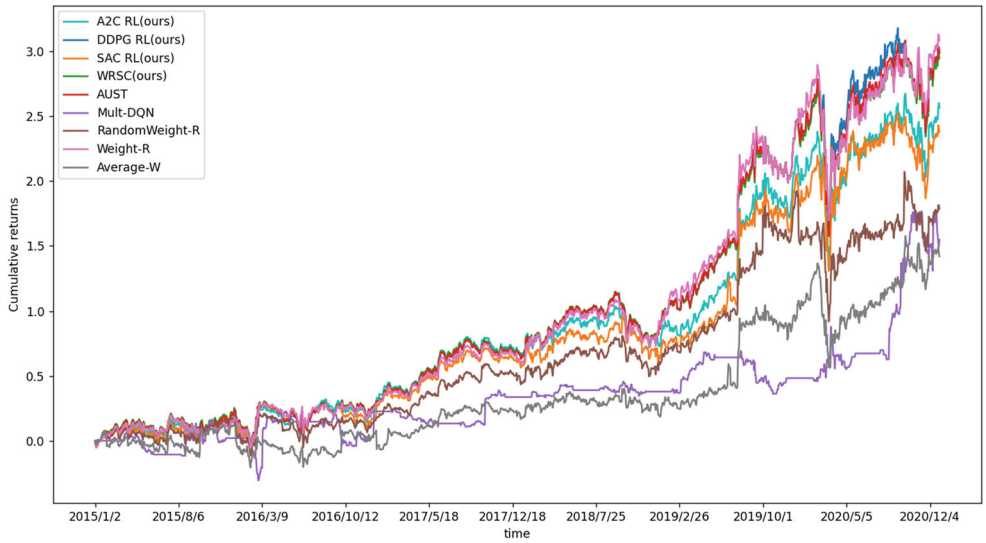

Fig. 9 Trends of cumulative returns of nested RL, WRSC and baselines for 2015/01/01-2020/12/30 on 30 British stocks

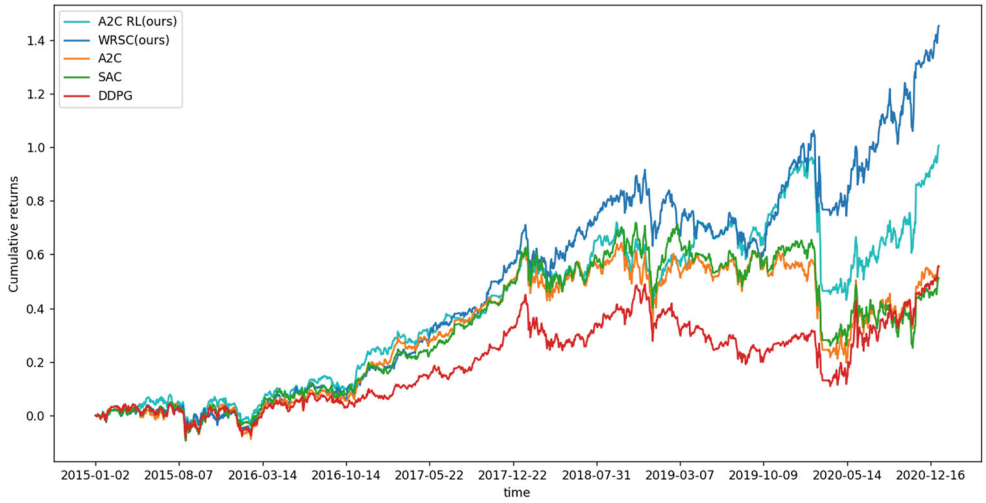

Fig. 10 Trend of cumulative returns of nested A2C RL, WRSC and the DDPG, the SAC, and the A2C on the U.S. stocks

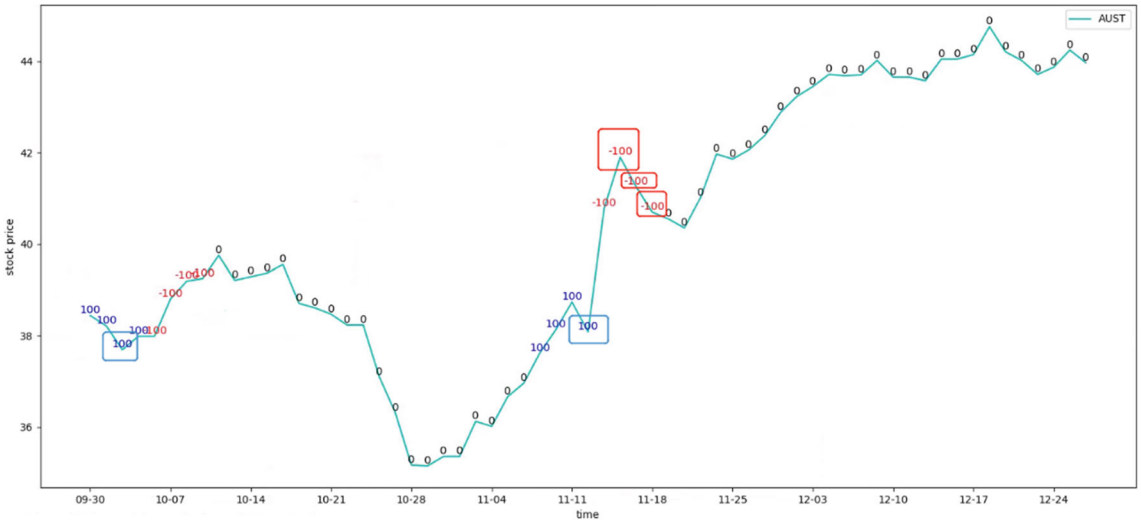

Fig. 11 Decision-making process of AUST on CSCO stock for 2020/09/30-2020/12/30

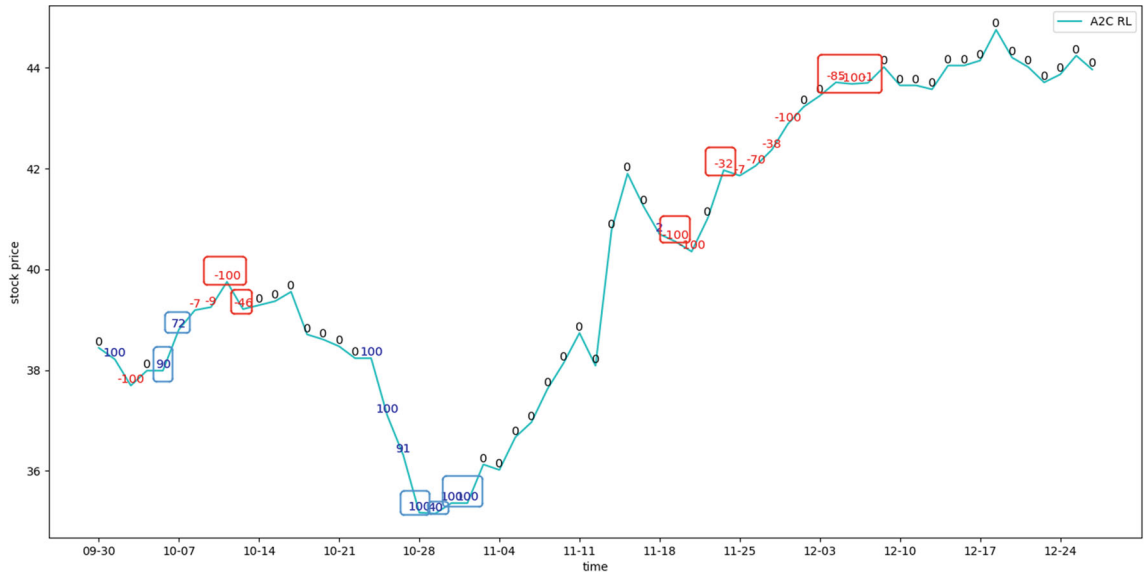

Fig. 12 Decision-making process of nested A2C RL on CSCO stock for 2020/09/30-2020/12/30

number represents the number of stocks bought on a day, and the red number represents the number of stocks sold.

Generally, both strategies can achieve the following expected decision behavior: buy at a low point and sell at a high point. Figure 11 shows that AUST only makes a buy decision at low points on 10/03 and 11/12 and makes a sell decision at three high points on 11/16, 11/17 and 11/18. In contrast, nested A2C RL can make decision-making operations at low and high points, as shown in Fig. 12. For example, buy-decisions were made at low points of 10/06, 10/08, and 10/28 to 11/01, and sell-decisions were made at high points of 10/11, 10/13, 11/17, 11/25 and 12/04. This proves that nested A2C RL can capture more trading opportunities and make accurate and timely decisions on whether to buy, sell or hold dynamically in response to changes in stock prices. In this way, nested A2C RL is able to obtain higher profits while avoiding investment risks.

6 Conclusion

In this paper, we analyze the applicability of a deep reinforcement learning model for stock decision-making. To address the complex environment of the stock market, we propose stock trading integration strategies based on deep reinforcement learning. One strategy is that of nested RL, which ensembles multiple deep reinforcement learning agents, including the A2C, DDPG and SAC. The other WRSC approach synthesizes the strategies of the three RL agents by computing their maximal annualized return and confidence. Experimental results for 90 stocks (the U.S., Japanese and British markets) demonstrate that both trading strategies perform better than the RandomWeight-R, Weight-R, Average-W, Multi-DQN and AUST ensemble strategies, which only use a greedy algorithm to select agents. Our methods can obtain more returns while ensuring lower risks and capture more trading decision-maker points in practical cases to adapt to different complex stock markets. And in addition to stock trading, our proposed method is also applicable to those scenarios that require intelligent decision-making. For example, automatic driving and route planning, game decision-making, recommendation system and service composition.

There are some limitations to our work. We only ensemble three DRL models as basic decision-makers and do not consider other factors involved in trading, such as sentiment and politics. Therefore, several types of research can be conducted in the future. First, we may explore more strong decision-makers with good performance and integrate them into nested strategies. Additionally, we could focus on other factors that influence stock trading, such as social news, sentiment [31], and politics. Finally, we may study ways to lower the annual volatility and investment risk of our RL methods under the conditions of high returns.

Acknowledgements This paper is supported by the National Key R&D Program of China (Funding No. 2018YFB1402800) and the State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2020ZX-01).

Author Contributions **Xiaoming Yu:** Conceptualization, Methodology, Validation, Writing - original draft. **Wenjun Wu:** Writing - review editing. **Xingchuang Liao:** Visualization, Investigation, Data curation. **Yong Han:** Proofread content.

Declarations

Conflict of Interests 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.

References

1. Wu ME, Syu JH, Lin JCW, Ho JM (2021) Portfolio management system in equity market neutral using reinforcement learning
2. Noonpakdee W (2020) The adoption of artificial intelligence for financial investment service. In: 2020 22nd International Conference on Advanced Communication Technology (ICACT), IEEE. pp. 396– 400
3. Zhang Y, Li X, Guo S (2018) Portfolio selection problems with Markowitz's mean–variance framework: a review of literature[J]. Fuzzy Optimization and Decision Making 17(2):125-158
4. Strumberger I, Bacanin N, Tuba M (2016) Constrained portfolio optimization by hybridized bat algorithm, IEEE, ISMS
5. Zhou J, Li X, Pedrycz W (2016) Mean-semi-entropy models of fuzzy portfolio selection[J]. IEEE Transactions on Fuzzy Systems 24(6):1627-1636
6. Xu Y., Yang C., Peng S., Nojima Y. (2020) A hybrid two-stage financial stock forecasting algorithm based on clustering and ensemble learning. Appl Intell 50:3852–3867
7. Chou JS, Nguyen TK (2018) Forward forecast of stock price using sliding-window metaheuristic-optimized machine-learning regression[J]. IEEE Transactions on Industrial Informatics 14(7):3132-3142
8. Deng Y, Xu H, Wu J (2021) Optimization of blockchain investment portfolio under artificial bee colony algorithm, vol 385
9. Jeon S, Hong B, Chang V (2018) Pattern graph tracking-based stock price prediction using big data[J]. Future Generation Computer Systems 80:171-187
10. Li G, Yang Y, Li S, et al. Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness[J]. Transportation Research Part C: Emerging Technologies, 2021: 103452
11. Kiran BR, Sobh I, Talpaert V, Mannion P, Al Sallab AA, Yogamani S, Pérez P (2021). Deep reinforcement learning for autonomous driving: A survey IEEE Transactions on Intelligent Transportation Systems
12. Gottesman O, Johansson F, Komorowski M, et al. (2019) Guidelines for reinforcement learning in healthcare[J]. Nature medicine 25(1):16-18
13. Johannink T, Bahl S, Nair A, Luo J, Kumar A, Loskyll M, Ojea J. A, Solowjow E, Levine S (2019) Residual reinforcement learning for robot control[C]//2019. In: International Conference on Robotics and Automation (ICRA). IEEE, pp 6023-6029
14. Vinyals O, Babuschkin I, Czarnecki WM, et al. (2019) Grandmaster level in StarCraft II using multi-agent reinforcement learning[J]. Nature 575(7782):350-354
15. Deng Y, Bao F, Kong Y, Ren Z, Dai Q (2016) Deep direct reinforcement learning for financial signal representation and trading. IEEE transactions on neural networks and learning systems 28:653–664
16. Lample G, Chaplot DS (2017) Playing fps games with deep reinforcement learning. In: Thirty-First AAAI Conference on Artificial Intelligence
17. Martinez C, Ramasso E, Perrin G, Rombaut M (2020) Adaptive early classification of temporal sequences using deep reinforcement learning. Knowl-Based Syst 190:105290
18. Song Y, Lee JW, Lee J (2019) A study on novel filtering and relationship between input-features and target-vectors in a deep learning model for stock price prediction. Appl Intell 49:897–911
19. Lee J, Koh H, Choe HJ (2021) Learning to trade in financial time series using high-frequency through wavelet transformation and deep reinforcement learning, Appl Intell, 1–22
20. Liu XY, Yang H, Chen Q, Zhang R, Yang L, Xiao B, Wang CD (2020) FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance. In: Deep Reinforcement Learning Workshop, 34th Conference on Neural Information Processing Systems (NeurIPS)
21. Yang X, Liu W, Zhou D, et al. (2020) Qlib: An AI-oriented Quantitative Investment Platform[J] Papers
22. Wu X, Chen H, Wang J, Troiano L, Loia V, Fujita H (2020) Adaptive stock trading strategies with deep reinforcement learning methods. Inf Sci 538:142–158
23. Li X, Li Y, Zhan Y, Liu XY (2019) Optimistic bull or pessimistic bear: Adaptive deep reinforcement learning for stock portfolio allocation. In: Proceedings of the 36th International Conference on Machine Learning (ICML)
24. Saphal R, Ravindran B, Mudigere D, Avancha S, Kaul B (2021) Seerl: Sample efficient ensemble reinforcement learning. In: Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)
25. Carta S, Corriga A, Ferreira A, Podda AS, Recupero DR (2021) Amulti-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. Appl Intell 51:889–905
26. Yang H, Liu XY, Zhong S, Walid A (2020) Deep reinforcement learning for automated stock trading: An ensemble strategy. Available at SSRN
27. Fischer TG (2018) Reinforcement learning in financial markets-a survey. Technical Report. FAU Discussion Papers in Economics
28. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning. In: International conference on machine learning, PMLR. pp. 1928–1937
29. Lillicrap T, Hunt J, Pritzel A et al (2016) Continuous control with deep reinforcement learning. In: International conference on learning representations
30. Haarnoja T, Zhou A, Abbeel P, Levine S (2018) Soft actor critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International conference on machine learning, PMLR. pp. 1861–1870
31. Vanstone BJ, Gepp A, Harris G (2019) Do news and sentiment play a role in stock price prediction? Appl Intell 49:3815–3820
32. Qi J, Rekkas M, Wong A (2018) Highly accurate inference on the sharpe ratio for autocorrelated return data[J]. Journal of Statistical and Econometric Methods 7(1):21-50
33. Hatami-Marbini A, Kangi F (2017) An extension of fuzzy TOPSIS for a group decision making with an application to Tehran stock exchange[J]. Applied Soft Computing 52:1084-1097
34. Rather AM, Sastry VN, Agarwal A (2017) Stock market prediction and Portfolio selection models: a survey[J]. Opsearch 54(3):558-579
35. Liu C. Legal Risks and the Countermeasures of Developing Intelligent Investment Advisor in China[C]//International Conference on Intelligent Human Systems Integration. Springer, Cham, 2018: 76-82
36. Li L, Wang J, Li X (2020) Efficiency analysis of machine learning intelligent investment based on K-means algorithm[J]. Ieee Access 8:147463-147470
37. Daldaban II (2020) Artificially intelligent investment advisers and the fiduciary duty problem: risks, challenges and regulatory solutions[J]

38. Coleman B, Merkley KJ, Pacelli J. (2021) Human versus Machine: A comparison of robo-analyst and traditional research analyst investment recommendations[J] Available at Social Science Research Network electronic journal
39. Deng Y, Bao F, Kong Y, et al. (2016) Deep direct reinforcement learning for financial signal representation and trading[J]. IEEE transactions on neural networks and learning systems 28(3):653-664
40. Leung MF, Wang J (2021) Minimax and Biobjective Portfolio Selection Based on Collaborative Neurodynamic Optimization. IEEE Transactions on Neural Networks and Learning Systems 32(7):2825–2836
41. Cao B, Zhao J, Lv Z, Gu Y, Yang P, Halgamuge SK (2020) Multiobjective Evolution of Fuzzy Rough Neural Network via Distributed Parallelism for Stock Prediction. IEEE Transactions on Fuzzy Systems 28(5):939–952
42. Wang L (2017) Modeling Stock Price Dynamics With Fuzzy Opinion Networks. IEEE Transactions on Fuzzy Systems 25(2):277–301
43. Hsu YL, Tsai YC, Li CT FinGAT: Financial Graph Attention Networks for Recommending Top-K Profitable Stocks. IEEE Transactions on Knowledge and Data Engineering, https://doi.org/10.1109/TKDE.2021.3079496
44. https://openai.com/blog/baselines-acktr-a2c/
45. Fortune (2021). Warren Buffett: Why Stocks Beat Gold and Bonds Accessed Dec., 20
46. Carta S, Ferreira A, Podda AS, et al. (2021) Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting[J]. Expert systems with applications 164:113820

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Xiaoming Yu received a master's degree in Tianjin University of Technology in 2018. She is currently a Ph.D. candidate at Beihang University and is majoring in software engineering at the State Key Laboratory of Software Development Environment. Her research interests include reinforcement learning, stock decision-making and workflow recommendation.

Wenjun Wu received his Ph.D. degree in computer science from Beihang University in 2001. He was employed at Argonne National Laboratory as a research scientist working on grid computing, cloud computing, media collaboration, etc., until 2012. He is currently employed at Beihang University as a professor. His research interests include crowdsourcing, machine learning, cloud computing, eScience and cyber infrastructure.

Xingchuang Liao is a graduate student at Beihang university majoring in software engineering at State Key Laboratory of Software Development Environment. His research interests include reinforcement learning based on multi-agent and stock, knowledge mapping and dialogue system.

Yong Han received a Master's degree in school of computer and communication engineering from university of science and technology beijing in 2015, he is now a Ph.D. candidate at Beihang university majoring in software engineering at State Key Laboratory of Software Development Environment. His research interest is Education Data mining.