Hindawi
Scientific Programming
Volume 2022, Article ID 7648810, 9 pages
https://doi.org/10.1155/2022/7648810

Research Article

Ensemble Investment Strategies Based on Reinforcement Learning

Fangyi Li, Zhixing Wang, and Peng Zhou

Chengdu University of Technology, Chengdu, Sichuan, China

Correspondence should be addressed to Zhixing Wang; wang.zhixing@student.zy.cdut.edu.cn

Received 16 June 2022; Revised 5 July 2022; Accepted 8 July 2022; Published 8 September 2022

Academic Editor: Lianhui Li

Due to the rapid development of hardware devices, the analytical processing and algorithmic capabilities of computers are also being enhanced, which makes machine learning play an increasingly important role in the field of quantitative investment. For this reason, the possibility of replacing traditional human traders with automated investment algorithms that have been trained several times has become a hot topic in recent years. The majority of machine algorithms used in today's stock trading market are supervised learning algorithms, which are still unable to objectively analyse the market and find the optimal solution for market trading on their own. To solve the two major challenges of environment awareness and automated decision-making, this study uses three core algorithms, PPO, A2C, and SAC, to build a set of ensemble automated trading strategies in a deep reinforcement learning-based framework. The *ensemble* trading strategy combines the advantages of each of the three algorithms to make the original reinforcement learning algorithm more adaptive, and to avoid consuming a large amount of memory when training the network, the study uses the PCA method to compress the dimension of the stock feature vector. We test our algorithm on 40 A-share stocks with sufficient liquidity and compare it with different trading strategies. The results show that the ensemble strategy proposed in this study outperforms three independent algorithms and two selected baselines, achieving an accumulated return of around 70%.

1. Introduction

Increasing computer processing power has led to the gradual digitization of financial market transactions, and it has become a reality to use computers to access large amounts of financial market data and complete high-frequency transactions within milliseconds [1]. The massive amount of data in the financial markets is often highly dimensional and noisy, and traditional econometrics cannot accurately quantify this multidimensional market data, so forecasting financial market data has always been a challenging problem in finance [2]. Most of the quantitative trading algorithms in the market today use supervised learning methods, which rely on manual design to select features and construct labels [3]. Quantitative trading systems under this approach require external managers with a certain level of financial knowledge to be able to change labels and parameters in a timely manner in response to market conditions, and therefore, the objectivity and independence of the algorithm cannot be guaranteed [4]. In contrast to supervised learning methods, reinforcement learning methods have the ability to learn control strategies from high-dimensional data [5]. Reinforcement learning methods do not require supervision by external managers and can continuously optimize paths to achieve the best cumulative returns through rewards and penalties during the interaction of market transactions [6].

Although reinforcement learning offers a new way of thinking about the analysis and prediction of financial data, there is still room for improvement. The first is that derivative algorithms on reinforcement learning often ignore the dominance of irrational investor sentiment in financial markets in the pursuit of decision objectivity, and researchers often combine sentiment mining with supervised algorithms, but few apply this to reinforcement learning methods [7]. This study addresses this issue by expressing market sentiment as a sentiment indicator and effectively improving the adaptability of algorithms to irrational markets. Secondly, it has been shown that reinforcement learning relies on a large amount of external environment as input data, but existing trading algorithms tend to select fewer technical indicators as the spatial state of the underlying investment to alleviate memory pressure [8], making the benefits of reinforcement learning methods less impressive than supervised learning methods. This study uses the PCA algorithm to compress the spatial vector dimension, incorporating more technical indicators to extend the spatial state of the underlying under the same memory pressure. In addition, the researcher found that different agents are suitable for different conditions of the stock market, which means that the model constructed by a single agent does not have the effectiveness and generalization ability in the face of different stock assets and different market environments [9]. So, this study constructs an ensemble strategy of three algorithms: PPO, A2C, and SAC. The ensemble strategy can choose the appropriate agent to maximize the accumulated return for different market environments and situations, so the ensemble strategy is more stable and reliable than a single strategy.

In the design of the ensemble model, we firstly constructed a deep reinforcement learning framework, setting up the environment, state space, and action space. Secondly, we designed a reward and punishment function for the agent to ensure that it can effectively optimize its own decisions. Thirdly, we connect the three different agents effectively through the Sharpe ratio. Finally, we demonstrate the effectiveness of the ensemble algorithm through ablation experiments and two baselines.

The study is structured as follows: Section 2 presents the relevant literature in the same research area, Section 3 describes the theoretical approach used in this study, Section 4 describes the detailed construction process of the ensemble model, Section 5 presents the data processing and empirical results, and finally we place the summary in Section 6.

2. Literature Review

Reinforcement learning has now evolved from its initial stand-alone application to a multiplicity of applications in combination with deep learning. It has been extensively investigated by many researchers in the field of finance due to its ability to effectively deal with continuous decision-making.

In terms of practical applications of reinforcement learning, Moody and Saffell proposed to construct portfolios and trade stocks with recurrent reinforcement learning and eventually proved that the returns of the reinforcement learning strategy were higher than those of the buy-and-hold strategy [10]. Tan et al. added an adaptive network fuzzy inference system as a supplement to the reinforcement learning framework to design a high-frequency trading strategy [11]. Sun and Bi selected convolutional neural networks and LSTM neural networks to build up and down classification models, respectively, based on which a high-frequency trading strategy was proposed and backtested with the main asphalt futures contract and demonstrated that the high-frequency trading strategy based on convolutional neural networks and long- and short-term memory neural networks had better profitability [12]. Dai and Zhang demonstrated that the reinforcement learning model outperformed the buy-and-hold strategy and MACD strategy in stock selection [13]. Lu and Salem applied the long- and short-term memory model (LSTM) to reinforcement learning and backtested it through forex trading, and the results demonstrated that the improved reinforcement learning model could effectively control the number of trades and maintain stable profitability [14]. Hu et al. constructed a cointegrated pair trading model based on the reinforcement learning SARSA algorithm and conducted simulation trading experiments on the Chinese bond market and demonstrated that the model outperformed the traditional model in all aspects and could significantly improve the profitability of the trading system [15].

In terms of algorithm design for reinforcement learning, Zhang and Wang and other scholars (2015) constructed a stock prediction model based on neural networks, which can solve the problem of high dimensionality of input data through genetic algorithms, and the results showed that genetic algorithms can improve model training efficiency [16]. Yung added news headlines for market opinion mining on the basis of stock time-series price data and in this way effectively improved the correct rate of model decisions [17]. Zhou et al. improved the traditional quantitative trading algorithm using the sentiment indicator ARBR, enabling the improved reinforcement learning algorithm to have richer returns in irrational markets [18]. Li et al. proposed a new trading model for deep reinforcement learning, which used two different reinforcement learning methods, stacked denoising self-coding (SDAEs) and long- and short-term memory (LSTM), and can effectively extract features from the raw data to build a robust trading agent, and experimental results show that the model achieves stable risk-adjusted returns in both the stock and future markets [19]. Gabrielsson and Johansson introduced seven new features based on Japanese candlesticks into the reinforcement learning input, and their HFT system outperformed the S&P 500 index and significantly outperformed the basic RRL algorithm in the test [20].

It is therefore easy to see that trading models built on reinforcement learning are well established and widely used in the financial field, achieving good returns in both the equity and foreign exchange markets. In terms of improvements to reinforcement learning algorithms, researchers have focused on improving different neural network structures and expanding the spatial state of the model.

3. Methodology

3.1. Reinforcement Learning Theory. Reinforcement learning is an important machine learning approach in current quantitative trading research [21]. Unlike common machine learning algorithms, the core of reinforcement learning is to allow an agent in an interactive environment to calculate the reward value for different actions using the current action state at the moment and to continuously optimize the agent's internal policy along the direction of the best reward value until the best policy is found [22]. In summary, the reinforcement learning framework consists

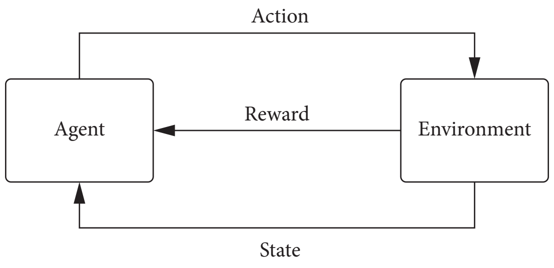

Figure 1: Structure of the reinforcement learning algorithm.

of the interactable state of the agent in the current environment, the different actions resulting from the decision, the policy for the decision made in the current state, the reward function used to calculate the reward value at the end of the action, the value function for the different actions and states, and the environment used to implement the agent interaction process.

The flow of the whole reinforcement learning algorithm is shown in Figure 1. At time t , the agent obtains the current state of the environment S_t and uses the policy function to process S_t to output the action a_t in the current state. After the action is completed, the action a_t will be applied to the environment, causing the environment state to change from S_t to S_{t+1}, and then, the function uses the action transfer state to calculate the reward value τ_t for the action a_t. The agent can use τ_t to continuously optimize future action strategies and ultimately maximize the cumulative reward value. The optimal strategy can be shown in (1), where π is the chosen strategy, γ is the discount rate, T is the total moment of interaction, and a is the state space.

$$\pi^* = \text{argmax}_\pi E_\pi \left\{ \sum_{T=0}^{\infty} \gamma^T r_{t+T} | S_t = S \right\}, \forall S \in S, \forall t \geq 0. \quad (1)$$

3.2. Markov Decision Process. The Markov decision process is the classical algorithm for reinforcement learning modelling, and its main idea is to perform dynamic planning with finding the maximum cumulative payoff on the MDP [23]. If the current state, which contains all relevant historical information, can be used to determine the future cumulative payoff, then we can consider the state to have Markovianity, and this property can be described as follows:

$$P(s_{t+1}|s_t) = P(s_{t+1}|s_t, \ldots, s_2, s_1). \quad (2)$$

We can describe the MDP using a set, as in (3), where S is the state space, which stores all states in the environment, A is the action space, which represents all actions that the agent can interact with, and P is the transfer probability, which represents the probability that an action taken by the agent in a state will result in a state transfer, which we usually identify as needing to satisfy $0 \leq p(s|s_t, a_t) \leq 1$. R is the reward generated by the action.

$$M = \{S, A, P, R, \gamma\}. \quad (3)$$

3.3. A2C. The actor-critic approach is one of the mainstream approaches in reinforcement learning, combining the advantages of both value-based and policy-based classical algorithms [24]. The core idea is to use the value of the state actions predicted by the critic model to optimize the decision-making behaviour of the actor model, and by alternating training, the generated actions can be made to better match the current environment and state. The structure of the model is shown in Figure 2.

Since the original AC model was slow to converge, the A2C algorithm was proposed to reduce the variance of the strategy gradient by adding a baseline to the strategy gradient while keeping the expectation of the strategy gradient random variable constant, allowing for faster convergence.

The gradient formula of the AC algorithm in its original state can be expressed as follows:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)(Q_\pi(s,a) - B)]. \quad (4)$$

Since the baseline function B is only related to the state S and not to the action A, the above equation can be derived as follows:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log\ \pi_\theta Q_\pi(s,a)]. \quad (5)$$

Since the baseline function B is a function independent of the action A, we can further optimize the formula using the state value function $V(s)$ as the baseline $B(s)$ and let the dominance function be $A(s,a) = Q(s,a) - V(s)$. Eventually, we can obtain the optimized gradient as follows:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log\ \pi_\theta(s,a) A(s,a)]. \quad (6)$$

In this study, the MLP neural network is chosen to build the A2C algorithm. The algorithm is updated step by step during training, and its training process is seen in Table 1.

3.4. PPO. The principle of the PPO algorithm is to represent the policy parametrically as $\pi_\theta(a|s)$, using a parametrized linear function or neural network to represent the policy [25].

The PPO algorithm strategy gradient is implemented by calculating an estimator combined with a stochastic gradient ascent algorithm, and the updated formula can be seen as follows:

$$\theta_r = \theta_b + \alpha \nabla_\theta J, \quad (7)$$

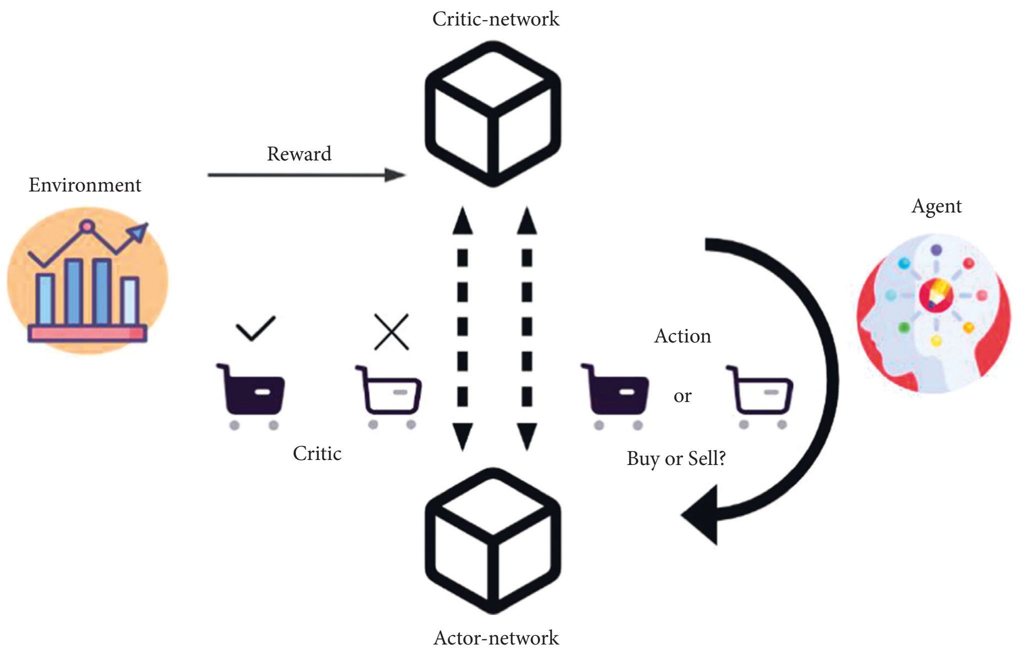

Figure 2: Structure of the actor-critic model.

Table 1: Pseudo-code for the A2C model.

```
Input: environment of the stock market
Output: estimated optimal strategy $\pi(\theta)$
Initial setup of actor and critic networks
Repeat
  For episodes = 0, 1, 2, . . ., N do:
    Get state $S$ and calculate $\pi(AS;\theta|)$ to get action $A$
    IF the episode does not end there:
      Get $S'$ with reward $\tau$
      Using critic networks to obtain return values to estimate $Q$
      Calculating the gradient using Q values and updating the actor network
      Updating the critic network to reduce the difference
      Update status $S$
    End
  End
To convergence
```

where θ_b is the policy parameter before the update, θ_τ is the updated policy parameter, α is the learning rate, and ∇_θ is the importance weight. J is the optimization objective, which is the expected value of the future reward in state S.

3.5. SAC. SAC is a heteroskedastic AC algorithm developed for maximum entropy reinforcement learning [26]. Unlike other methodological theories, the SAC algorithm changes the goal of reinforcement learning by the introduction of the concept of entropy, which actually improves the exploratory and robustness of the algorithm [27].

The entropy [28–35] of the distribution x can be expressed as follows:

$$H(p) = E_{X\sim p}[-\ln\ p(x)]. \tag{8}$$

In order not to miss any valid actions and trajectories and to promote strategy randomization for greater robustness and exploration, the maximum entropy reinforcement learning algorithm requires the strategy function to output the action expressed as follows:

$$\pi^* = \operatorname{argmax}_\pi E_{s_t,a_t\sim\rho_\pi}\left[\sum_t r_t + \alpha H(\pi(\cdot|s_t))\right]. \tag{9}$$

4. Ensemble Model Development

The ensemble model can be divided into two parts: the first part is to select a suitable algorithm from A2C, PPO, and SAC as agent, and the second part is to build the state space of the stock through stock price, technical indicators, and sentiment indicators to describe the stock trading market environment [36–39]. When these two parts are completed, the action space and reward function will link the agent with the environment so that the agent

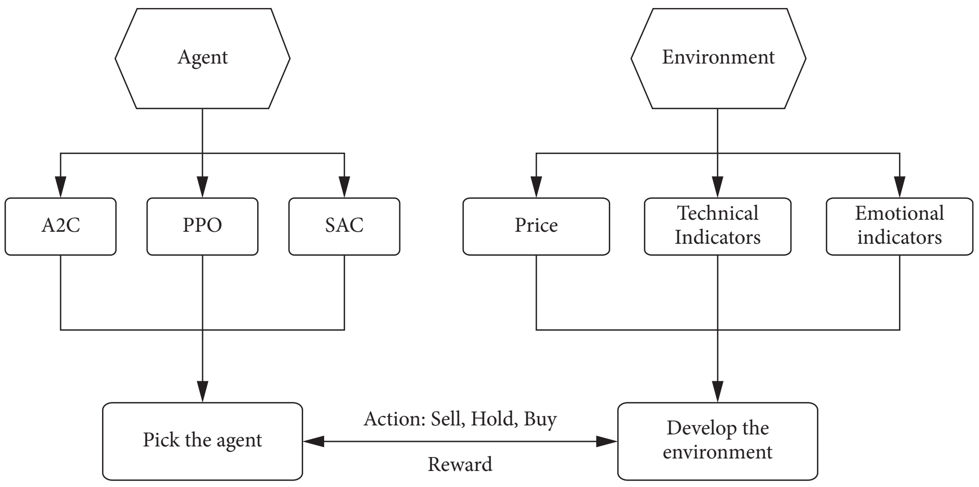

FIGURE 3: Structure of the ensemble strategy.

can make continuous decisions to maximize the cumulative return. The structure of the ensemble model is shown in Figure 3.

4.1. Select the Agent. In this study, we use a several months' long window to train all three agents simultaneously, and every three months, we retrain our three agents. At the same time, we use the latter three months of the training window to validate the performance of the three different agents and select the agent with the highest Sharpe ratio as the agent of the ensemble strategy for the underlying investment. The following equation shows how the Sharpe ratio is calculated:

$$\text{Sharpe ratio} = \frac{\overline{r}_p - r_f}{\sigma_p}. \tag{10}$$

4.2. Setting Up the Environment. We incorporate the daily opening and closing prices of the stock: the technical indicators of the stock and the sentiment indicator ARBR, which describes market sentiment, as the spatial state of the stock.

It is worth mentioning that to incorporate more indicators as the spatial state of the stock while relieving the memory pressure on the algorithm, we used the PCA algorithm to compress the original 24-dimensional feature vector to 20 dimensions. Figure 4 shows the correlation hotspots of the technical indicators selected by the ensemble model. It is easy to see that there is a large positive correlation between vol10 and vol20 and a large negative correlation between the deviation rate BIAS and MA, so the PCA algorithm can be used to reduce the dimensionality.

4.3. Transaction Cost. Since every transaction in the stock market incurs transaction costs and the rules for transaction costs in the stock exchange vary from country to country, we have set a uniform transaction cost of 0.1% of the value of each transaction.

4.4. Reward Function. We define the reward value as the maximum profit that each group of stocks can take in a given period of time, expressed as follows:

$$r_t = p_t - p_{t-1}, \tag{11}$$

where r_t is the value of the reward currently received, p_t is the price of the stock at time t, and p_{t-1} is the price of the stock at the previous time. The reward value is therefore the difference between the two momentary prices. When the current price is greater than the past price, a positive reward value is obtained; when the current price is lower than the past price, a negative reward value is obtained. The final cumulative return is as follows:

$$R_t = \sum_{t=1}^{T} r_t. \tag{12}$$

4.5. Action Space. To make it easier to calculate the profitability of the ensemble model in the stock market, we do not consider shorting trades in the stock and simply use buy, sell, hold, and wait and see as the action states of the stock. This can be expressed as follows:

$$a = \begin{cases} 1, & \text{Buy}, \\ 0, & \text{Wait and see}, \\ -1, & \text{Sell}. \end{cases} \tag{13}$$

5. Empirical Results

5.1. Data Preprocessing. We select the constituent stocks of the CSI 100 index as the pool of stocks to be traded in the pooled strategy. We use historical daily data from 1 January 2010 to 12 February 2021 for the evaluation of model returns. The stock data used in this study are downloaded via the wind terminal. As mentioned above, we split the historical stock data into two parts: one for the training of the three agents, PPO, A2C, and SAC, and the other for the validation of the three agents, including the adjustment of the learning

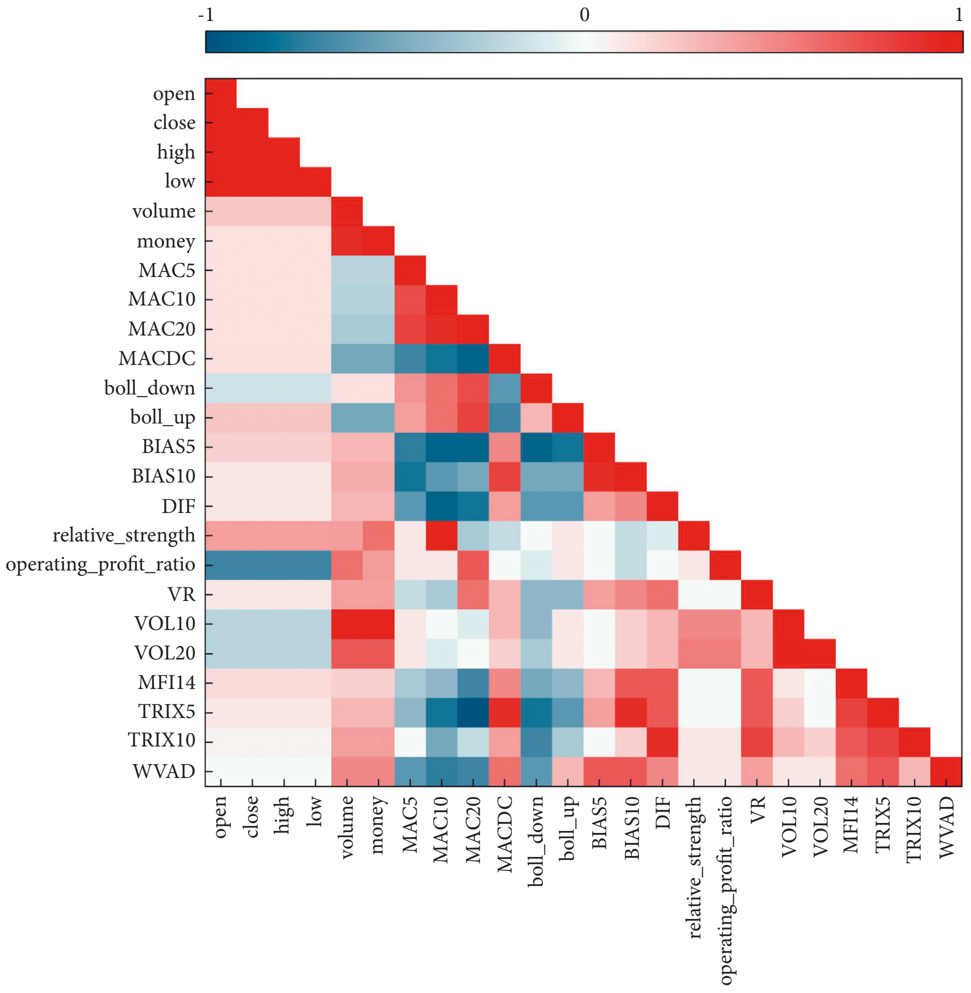

Figure 4: Relevance of selected technical indicators.

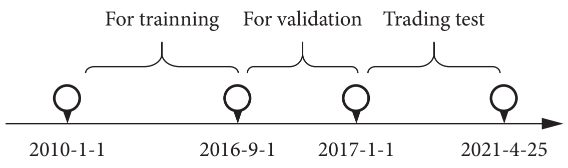

Figure 5: Classification of empirical data.

rate and key parameters. After selecting a suitable agent by comparing the Sharpe ratio, we start the real trading test and compare the results. The three agent's algorithms are MACD, and Min-Variance two baselines. A breakdown of the training data used is seen in Figure 5.

5.2. Test Result. The backtest results of the ensemble model and the comparison model are shown in Figure 6. It is easy to see that the ensemble model achieved a cumulative return of 71.92% and an annual return of 17.98%, which is higher than the remaining two individual agent models with the baseline in terms of return results. More detailed backtesting data are shown in Table 2. The ensemble model has the lowest annual volatility, which proves that the model is more stable and reliable than the other models, while the min-variance model has the highest annual volatility. In terms of the Sharpe ratio, the ensemble model achieved the highest Sharpe ratio, while it had the lowest maximum capital withdrawal rate. Overall, the A2C, PPO, and SAC models all achieved above-baseline returns, demonstrating that all three models have some portfolio management capability. In contrast, the ensemble model achieved a cumulative return of 70%, while its stability, Sharpe ratio, and maximum return were better than the other models, demonstrating the effectiveness of the model in the equity market.

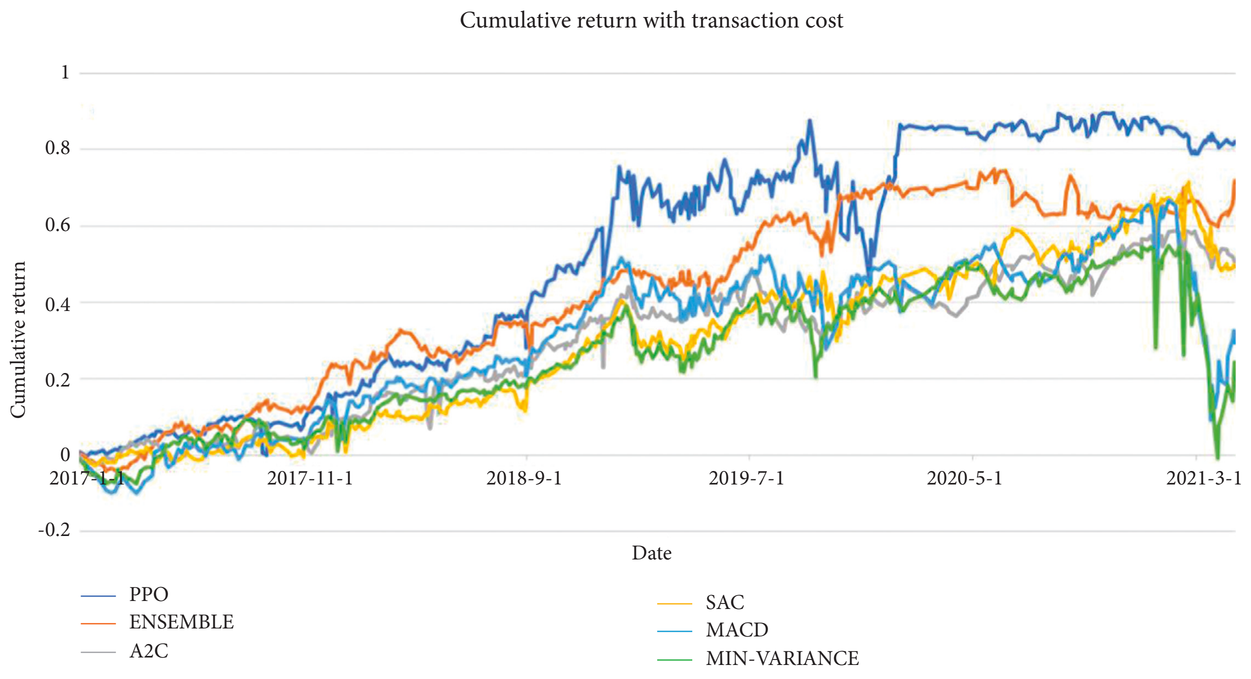

Figure 6: Cumulative return with transaction cost.

Table 2: Backtest results under different strategies.

2017/01/01–2021/04/25	Ensemble	PPO	A2C	SAC	MACD	Min-variance
Cumulative return	71.92%	81.99%	50.73%	49.23%	29.32%	24.27%
Annual return	17.98%	20.5%	12.68%	12.31%	7.33%	6.07%
Annual volatility	9.7%	14.8%	11.3%	14.9%	18.3%	20.1%
Sharpe ratio	1.41	1.12	1.15	0.78	0.42	0.45
Max drawdown	−9.5%	−25.3%	−18.4%	−14.8%	−38.8%	−34.6%

6. Conclusion

In this study, we propose an ensemble trading strategy in a reinforcement learning framework, which selects the appropriate strategy as agent from three strategies, PPO, A2C, and SAC, through the Sharpe ratio, and incorporates more stock indicators and data as the state space of the stock using the PCA method. Through backtesting on the CSI 100, the results show that the proposed model outperforms the two agent models A2C and SAC in terms of return and outperforms all three independent agent models and the two baselines in terms of Sharpe ratio, annual volatility, and maximum retracement, so the ensemble model is innovative and superior and has research and application value.

Data Availability

The dataset can be accessed upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Y. Liu, Q. Liu, H. Zhao, Z. Pan, and C. Liu, “Adaptive quantitative trading: an imitative deep reinforcement learning approach,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 2, pp. 2128–2135, 2020, April.

[2] X. Guo, T. L. Lai, H. Shek, and S. P. S. Wong, *Quantitative Trading: Algorithms, Analytics, Data, Models, Optimization*, Chapman and Hall/CRC, Boca Raton, FL, USA, 2017.

[3] R. Caruana and A. Niculescu-Mizil, “An empirical comparison of supervised learning algorithms,” in *Proceedings of the 23rd international conference on Machine learning*, pp. 161–168, Pittsburgh, PA, USA, June, 2006.

[4] D. H. Lee, “Pseudo-label: the simple and efficient semi-supervised learning method for deep neural networks,” *Workshop on challenges in representation learning ICML*, vol. 3, no. 2, p. 896, 2013.

[5] M. Eppe, C. Gumbsch, M. Kerzel, P. D. H. Nguyen, M. V. Butz, and S. Wermter, “Intelligent problem-solving as integrated hierarchical reinforcement learning,” *Nature Machine Intelligence*, vol. 4, no. 1, pp. 11–20, 2022.

[6] L. Chen, K. Lu, A. Rajeswaran et al., “Decision transformer: reinforcement learning via sequence modeling,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[7] F. M. J. Mehedi Shamrat, S. Chakraborty, M. M. Imran et al., "Sentiment analysis on twitter tweets about COVID-19 vaccines using NLP and supervised KNN classification algorithm," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 1, p. 463, 2021.

[8] E. L. Roscow, R. Chua, R. P. Costa, M. W. Jones, and N. Lepora, "Learning offline: memory replay in biological and artificial reinforcement learning," *Trends in Neurosciences*, vol. 44, no. 10, pp. 808–821, 2021.

[9] K. Xia, C. Sacco, M. Kirkpatrick et al., "A digital twin to train deep reinforcement learning agent for smart manufacturing plants: environment, interfaces and intelligence," *Journal of Manufacturing Systems*, vol. 58, pp. 210–230, 2021.

[10] J. Moody and M. Saffell, "Learning to trade via direct reinforcement," *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 875–889, 2001.

[11] Z. Tan, C. Quek, and P. Y. Cheng, "Stock trading with cycles: a financial application of ANFIS and reinforcement learning," *Expert Systems with Applications*, vol. 38, no. 5, pp. 4741–4755, 2011.

[12] D. C. Sun and X. C. Bi, "High-frequency trading strategies based on deep learning algorithms and their profitability," *Journal of University of Science and Technology of China*, vol. 11, pp. 923–932, 2018.

[13] S. X. Dai and S. L. Zhang, "An application of reinforcement learning based approach to stock trading," *Business Management*, vol. 3, pp. 23–27, 2021.

[14] Y. Lu and F. M. Salem, "Simplified gating in long short-term memory (lstm) recurrent neural networks," in *Proceedings of the 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1601–1604, IEEE, Boston, MA, USA, August 2017.

[15] B. Hu, J. Li, J. Yang et al., "Reinforcement learning approach to design practical adaptive control for a small-scale intelligent vehicle," *Symmetry*, vol. 11, no. 9, p. 1139, 2019.

[16] W. Zhang and W. Wang, "Deep learning-based stock market forecasting," *Journal of Wuhan University (Natural Science Edition)*, vol. 201439, no. 1, pp. 1–7.

[17] Z. H. Yung, "Deep reinforcement learning stock algorithmic trading system application," *Computer Knowledge and Technology*, vol. 23, pp. 75-76, 2020.

[18] P. Zhou, J. Tang, and Y. Li, "Research on investment strategies of stock market based on sentiment indicators and deep reinforcement learning," in *Proceedings of the International Conference on Statistics, Applied Mathematics, and Computing Science (CSAMCS 2021)*, vol. 12163, pp. 1151–1156, SPIE, April 2022.

[19] Y. Li, W. Zheng, and Z. Zheng, "Deep robust reinforcement learning for practical algorithmic trading," *IEEE Access*, vol. 7, Article ID 108014, 2019.

[20] P. Gabrielsson and U. Johansson, "High-frequency equity index futures trading using recurrent reinforcement learning with candlesticks," in *Proceedings of the 2015 IEEE Symposium Series on Computational Intelligence*, pp. 734–741, IEEE, Cape Town, South Africa, December 2015.

[21] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing Atari with Deep Reinforcement Learning," 2013, https://arxiv.org/abs/1312.5602.

[22] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of Real-World Reinforcement Learning," 2019, https://arxiv.org/abs/1904.12901.

[23] F. Garcia and E. Rachelson, "Markov decision processes," *Markov Decision Processes in Artificial Intelligence*, pp. 1–38, Wiley, Hoboken, NJ, USA, 2013.

[24] H. Chen, Y. Liu, Z. Zhou, and M. Zhang, "A2C: attention-augmented contrastive learning for state representation extraction," *Applied Sciences*, vol. 10, no. 17, p. 5902, 2020.

[25] L. Engstrom, A. Ilyas, S. Santurkar et al., "Implementation matters in deep rl: a case study on ppo and trpo," in *Proceedings of the International conference on learning representations*, New Orleans, LA, USA, September 2019.

[26] C. Yang and J. Collins, "Deep learning for pollen sac detection and measurement on honeybee monitoring video," in *Proceedings of the 2019 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pp. 1–6, IEEE, Dunedin, New Zealand, December 2019.

[27] J. Zhang, Q. Wang, and W. Shen, "Message-passing neural network based multi-task deep-learning framework for COSMO-SAC based σ-profile and VCOSMO prediction," *Chemical Engineering Science*, vol. 254, Article ID 117624, 2022.

[28] L. Li and C. Mao, "Big data supported PSS evaluation decision in service-oriented manufacturing," *IEEE Access*, vol. 8, Article ID 154663, 2020.

[29] X. Lu, "Research on biological population evolutionary algorithm and individual adaptive method based on quantum computing," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 5188335, 9 pages, 2022.

[30] L. Li, C. Mao, H. Sun, Y. Yuan, and B. Lei, "Digital twin driven green performance evaluation methodology of intelligent manufacturing: hybrid model based on fuzzy rough-sets AHP, multistage weight synthesis, and PROMETHEE II," *Complexity*, vol. 2020, no. 6, 24 pages, Article ID 3853925, 2020.

[31] L. Zhang and L. Qiu, "Aerobic exercise fatigue detection based on spatiotemporal entropy and label technology," , p. 2022, Scientific Programming, 2022.

[32] L. Li, T. Qu, Y. Liu et al., "Sustainability assessment of intelligent manufacturing supported by digital twin," *IEEE Access*, vol. 8, Article ID 174988, 2020.

[33] J. Zhang, "A study on mental health assessments of college students based on triangular fuzzy function and entropy weight method," *Mathematical Problems in Engineering*, vol. 2021, Article ID 6659990, 8 pages, 2021.

[34] L. Li, B. Lei, and C. Mao, "Digital twin in smart manufacturing," *Journal of Industrial Information Integration*, vol. 26, no. 9, Article ID 100289, 2022.

[35] Y. Li, F. Ning, X. Jiang, and Y. Yingmin, "Feature extraction of ship radiation signals based on wavelet packet decomposition and energy entropy," *Mathematical Problems in Engineering*, vol. 2022, Article ID 8092706, 12 pages, 2022.

[36] J. C. . Alves, D. M. da Silva, and G. R. Mateus, "Applying and comparing policy gradient methods to multi-echelon supply chains with uncertain demands and lead times. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), v 12855 LNAI," in *Proceedings of the Artificial Intelligence and Soft Computing - 20th International Conference, ICAISC 2021*, pp. 229–239, June 2021.

[37] L. Sun, M. K Siddique, L. Wang, and S. J. Li, "Mixing characteristics of a bubble mixing microfluidic chip for genomic DNA extraction based on magnetophoresis: CFD simulation and experiment," *Electrophoresis*, vol. 42, no. 21-22, pp. 2365–2374, 2021.

[38] J. Zheng, M. N. Kurt, and W. Xiaodong, “Integrated actor-critic for deep reinforcement learning. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), v 12894 LNCS,” in *Proceedings of the artificial neural networks and machine learning – ICANN 2021 - 30th international conference on artificial neural networks*, pp. 505–518, Bratislava, Slovakia, September 2021.

[39] X. Wenwen, “Application Research of end to end behavior decision based on deep reinforcement learning,” in *Proceedings of the 2021 5th International Conference on Electronic Information Technology and Computer Engineering, EITCE 2021*, pp. 889–894, Xiamen, China, October 2021.