mathematics

MDPI

Article

A Portfolio Model with Risk Control Policy Based on Deep Reinforcement Learning

Caiyu Jiang and Jianhua Wang *

School of Science, Wuhan University of Technology, Wuhan 430070, China
* Correspondence: wangjh@whut.edu.cn

Abstract: It was shown that deep reinforcement learning (DRL) has the potential to solve portfolio management problems in recent years. The Twin Delayed Deep Deterministic policy gradient algorithm (TD3) is an actor-critic method, a typical DRL method in continuous action space. Despite the success of DRL in financial trading, surprisingly, most of the literature ignores the element of risk control. The research is proposed to combine long- and short-term risk (LSTR) control with the TD3 algorithm to build a portfolio model with risk management capabilities. Using Chinese stock data from the Shanghai Stock Exchange, we train and validate the proposed portfolio model. Performances were compared to the TD3 model without risk control. The results indicated that our proposal offers better risk control and investment returns.

Keywords: deep reinforcement learning; portfolio management; long- and short-term risk control

MSC: 68T07

1. Introduction

Portfolio management is the decision-making process of continually reallocating funds to various financial products to maximize returns within a specific range of risks. The portfolio theory based on the mean-variance model initially proposed by Markowitz in 1952 is an essential foundation for portfolio management [1]. However, the model must satisfy the hypothesis that returns follow a normal distribution to measure risk effectively. Therefore, many scholars are committed to finding more reasonable and effective risk measurement methods [2]. Many traditional portfolio algorithms were developed to maximize final returns, such as UP, EG, ONS, and CFR-OGD [3–6]. However, these algorithms may incur significant losses in specific markets. The fundamental reason is that they are not designed to address risk directly [7]. Subsequently, it was suggested that risk should be defined based on variance, half variance, and the likelihood of adverse outcomes [8]. However, any portfolio strategy using these three risk definitions requires the collection of adequate performance observations to make empirical estimates of the likelihood properties associated with risk. These portfolio algorithms can monitor long-term risk but do not have the ability to monitor short-term risk. To achieve effective trading algorithms, it is also helpful to make specific risk enhancements to existing portfolio algorithms. One method involves monitoring the maximum drawdown of each portfolio vector, i.e., the investor gives more cash to the portfolio vector with a smaller maximum drawdown at each trading period [9]. Another approach is to add factors describing risk to the reward function of reinforcement learning. Arguably, the most common variable for risk control is the Sharpe ratio [10,11]. The Sharpe ratio calculation includes the mean and standard deviation of portfolio returns, both used as long-term indicators describing the market with a small amount of short-term variation. Given the above problems, Bai et al. proposed a long- and short-term risk control algorithm to control both long-term and short-term loss risk by controlling the proportion of risk-free assets and achieved good results on six financial market data sets [12].

Citation: Jiang, C.; Wang, J. A Portfolio Model with Risk Control Policy Based on Deep Reinforcement Learning. *Mathematics* **2023**, *11*, 19. https://doi.org/10.3390/math11010019

Academic Editor: Andrea Scozzari

Received: 23 October 2022
Revised: 14 December 2022
Accepted: 19 December 2022
Published: 21 December 2022

Reinforcement learning, an essential branch of machine learning, has been widely used in financial market trading in the past decades to predict price movements using historical market data. The advantage of reinforcement learning is that investors do not need to focus on much expertise beyond the basic trading rules. Moody et al. were the first to propose a framework for cyclic reinforcement learning to trade a single asset, which uses the differential Sharpe ratio as the objective function to maximize risk-adjusted returns by considering transaction cost [13]. Dempster, Deng, Almahdi, and Yang et al. developed RRL frameworks successively for trading assets [14–16]. Bertoluzzo et al. considered Q-learning to construct automatic financial trading systems for artificial and real-time series of daily financial asset prices [17]. Unfortunately, the action spaces of these algorithms are discrete.

Therefore, these reinforcement learning methods with discrete action space cannot be directly applied to portfolio management problems with continuous action space. Deep reinforcement learning has demonstrated the ability to learn complex strategies from many types of environments in recent years due to its great success in gaming. Deep Q-network (DQN) is a popular approach in deep reinforcement learning. Zhang et al. adopted it to design trading strategies for continuous futures contacts [18]. In order to further improve the versatility of the algorithm, Lillicrap et al. proposed the deep deterministic policy gradient (DDPG) algorithm using deterministic policies instead of random policies, which can solve a series of control problems in continuous action space [19]. Jiang et al. and Xiong et al. conducted tests based on DDPG in digital currency and stock markets to demonstrate the effectiveness of the DDPG algorithm in portfolio management [20,21]. Wang et al. combined the classical DDPG and the hierarchical RL structure for portfolio management problems [22]. It was tested on the U.S. stock market, which illustrates this approach can outperform the classical DDPG. Its advantage is that it can effectively use reinforcement learning tasks in large-scale or continuous action spaces. Still, DDPG also suffers from bias and variance problems introduced by function approximation. To solve this problem, Meger et al. proposed a TD3 algorithm based on the DDPG algorithm [23]. However, to my knowledge, few scholars have applied it to the portfolio management problem.

Since deep reinforcement learning shows excellent potential in portfolio problems, inspired by Huang et al., this paper attempts to apply the TD3 algorithm to the portfolio management problem in the Chinese stock market [24]. In addition, considering that existing deep reinforcement learning algorithms assume less risk when constructing the objective function. This paper introduces the long- and short-term risk control algorithm into TD3 to build the LSTR–TD3 algorithm, which can update the resulting actions by updating the risk control parameters to optimize portfolio strategies. Three different performance metrics are used in the experiments to construct three other portfolios to evaluate the algorithm's performance from different perspectives. The experimental results show that the LSTR–TD3 algorithm is effective in the Chinese stock market and possesses a more robust risk control capability than the original TD3 algorithm. The experimental results also show that the method proposed in this paper is adequate for investors who suffer huge losses due to special reasons such as financial crises.

The paper is organized as follows. Section 2 describes the definition of portfolio models. Section 3 introduces the classical TD3 algorithm and LSTR algorithm. Section 4 presents the novel algorithm LSTR–TD3 algorithm proposed in this paper. Section 5 shows the experimental setup and results. The conclusions are shown in Section 6.

2. Portfolio Allocation Problem

Considering the interactive, stochastic nature of the stock trading market, the trading process of the portfolio can be modeled as a Markov decision process. We set the state $S_t = (X_t, W_t)$, where X_t is a tensor consisting of price features and W_t is a weight vector composed of the weight of each asset in the portfolio. Tensor X_t consists of four price features: close price, high price, low price, and open price, where t indicates the tth trading period. The structure of X_t is shown in Figure 1.

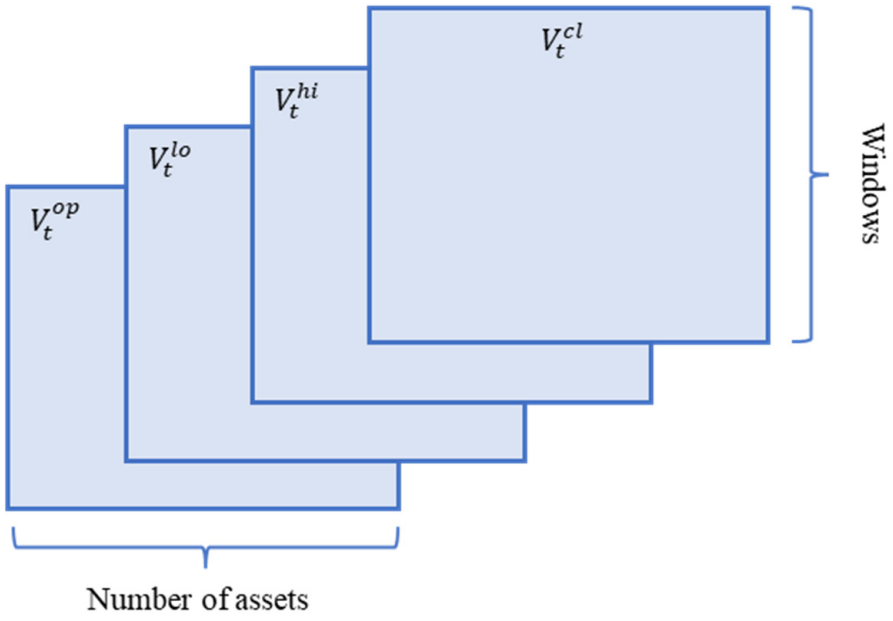

Figure 1. The data structure of the price tensor.

We set the weight of each asset in the portfolio as actions, i.e., $A_t = W_t$ and initializing the portfolio weights to $W_0 = (1, 0, \cdots, 0)^T$, which means before the transaction begins, there is no money invested in other assets and only cash assets in the portfolio. It is also assumed that the entire portfolio has one cash asset and m risky assets (in this paper, risky assets refer to CSI500 constituent stocks and CSI300). Then, A_t can be expressed as:

$$A_t = (\omega_{0,t}, \omega_{1,t}, \omega_{2,t}, \cdots, \omega_{m,t}) \tag{1}$$

$\omega_{0,t}$ indicates the percentage of cash held by investors in the trading period and $\omega_{m,t}$ is the percentage of each stock held in the tth trading period.

Using the daily log returns of the portfolio as the reward function R_t, if the portfolio price for the tth trading period is denoted by ρ_t. Then, ignoring transaction costs, the portfolio price can be expressed as:

$$\rho_t = \rho_{t-1} \exp[(\ln Y_t) \cdot W_{t-1}] \tag{2}$$

The symbol $\cdot$ denotes the dot product of the vectors, Y_t represents the relative price vector, i.e., the ratio of the closing price vectors of each asset of the trading period t and the trading period $t-1$ of a portfolio.

The daily log rate of return of a portfolio is:

$$\gamma_t = \ln(\rho_t / \rho_{t-1}) \tag{3}$$

Assume that the portfolio weight vector is W_{t-1} at the beginning of the trading time t. Due to stock price changes, the portfolio weights will shift and become W_t' at the end of trading time t. The weight evolution equation is:

$$W_t' = (Y_t \odot W_{t-1}) / (Y_t \cdot |W_{t-1}|) \tag{4}$$

where the symbol $\odot$ denotes the Hadamard product. The weight vector will evolve from W_t' to W_t before the start of the trading time $t+1$, such that the entire transaction cost rate of the portfolio at this point is:

$$C_t = \mu_t \left(\sum_{i=1}^{m} |\omega_{i,t}' - \omega_{i,t}| \right) \tag{5}$$

Figure 2 demonstrates the dynamic relationship between the portfolio value and weight vector.

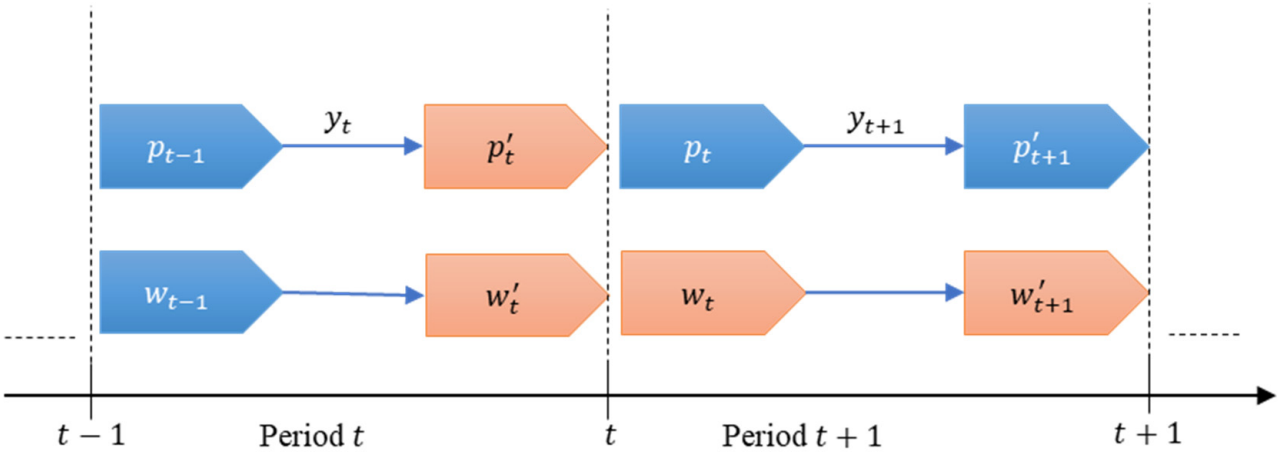

Figure 2. Portfolio value and weight vector evolution.

μ_t is the transaction cost rate of a single asset on the stock exchange, including commissions and other costs in the trading process. Then, the portfolio price after the introduction of transaction costs is:

$$\rho_t = \rho_{t-1}(1 - C_t)\exp[(\ln Y_t) \cdot W_{t-1}] \tag{6}$$

3. Methodology

In this section, we first describe the algorithmic architecture of the classical TD3 algorithm. Then, we describe the LSTR method by directly changing the proportion of risk-free assets. Finally, we describe the algorithm for adding LSTR to the TD3 algorithm.

3.1. TD3 Algorithm

The TD3 algorithm is an improved version of the DDPG algorithm, mainly to solve the problem of overestimation of Q-values due to function approximation error [23]. The DDPG algorithm improves the DPG algorithm based on the AC framework with the empirical playback of DQN and the dual network structure [19]. The DDPG algorithm learns the approximate value function $Q(s, a, w)$ and the deterministic policy $\mu(s, \theta)$. The network model is trained using stochastic gradient descent, where w and θ are the weights of the value network and the policy network, respectively. In addition, the DDPG algorithm solves the problem of inter-data correlation and non-stationary normal distribution by introducing an empirical replay mechanism to reduce the bias arising from the value function estimation. DQN knows that learning using only a single network can be unstable, therefore DDPG introduces target networks $Q'(s, a, w')$ and $\mu'(s, a, \theta')$, for the value network $Q(s, a, w)$, and the policy network $\mu(s, a, \theta)$, respectively [10].

As with the DQN, the DDPG utilizes the TD error-based MSE as a loss function.

$$y = r + \gamma Q'(s', \mu'(s', \theta'), w') \tag{7}$$

$$L(w) = \mathrm{E}[(y - Q(s, a, w))^2] \tag{8}$$

The objective of the value network is to minimize the loss function, therefore the MSGD method is used to obtain N small batches of randomly sampled data from the empirical pool D as a sampled estimate of the expected value.

$$\nabla_w L(w) \approx \frac{1}{N}\sum_{i=1}^{N}(r_i + \gamma Q'(s'_i, \mu'(s'_i, \theta'), w') - Q(s_i, a_i, w)\nabla_w Q(s_i, a_i, w) \tag{9}$$

where θ' and w' denote the weights of the target policy network μ' and the target value network Q', respectively.

To address the problem of overestimating Q-values in the DDPG, TD3 use clipped double Q-learning, delayed policy update, and target policy smoothing to alleviate this problem [23]. The structure of the TD3 algorithm is shown in Figure 3.

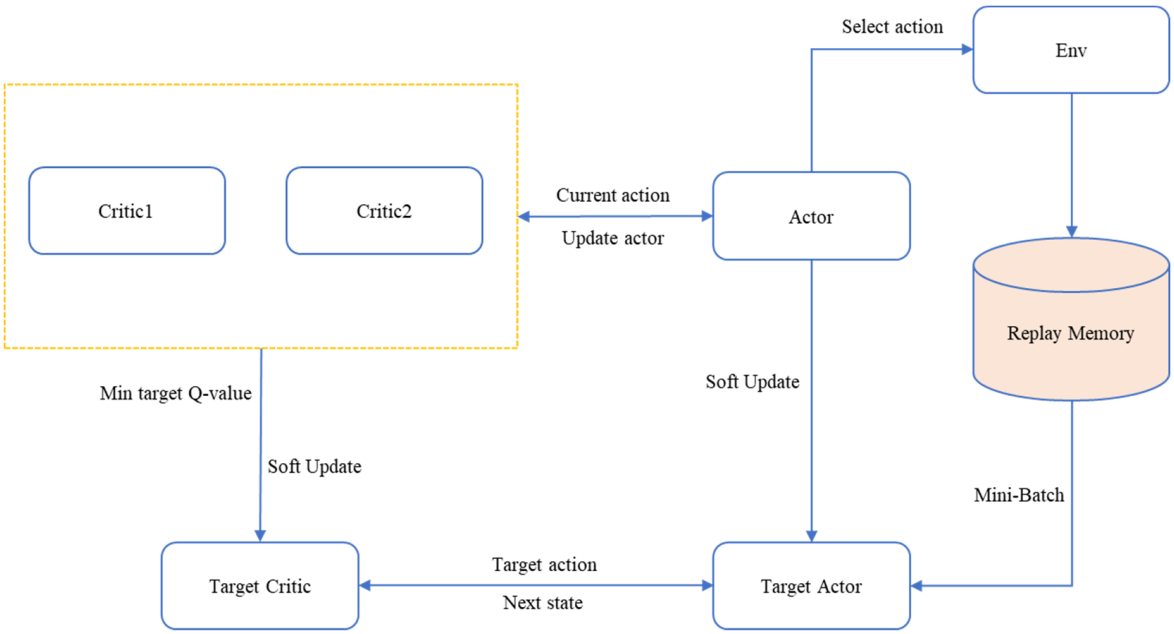

Figure 3. TD3 algorithm flowchart.

Truncated double Q-learning uses two independent critics Q_{w_1} and Q_{w2} and two independent actors μ_{θ_1} and μ_{θ_2}, splitting the original Q-value function into two parts and choosing a minimum of two critic networks Q_{w_1} and Q_{w2} as the value of function estimation.

$$y = r + \gamma \min_{i=1,2} Q(s', \mu(s', \theta), w_i') \tag{10}$$

The above equation is used to update the critic network. The following loss function is shared.

$$L(w_i) = \mathrm{E}_{s,a,r,s' \sim D}[(y - Q(s, a, w_i))^2] \tag{11}$$

The delayed policy update is to make the target network out of sync with the current network update. The current network updates d times before the target network is updated. This reduces the accumulated error and the variance. The same strategy network can also be updated on a delayed basis because the parameters are updated slowly in the AC method. Performing a delayed update can reduce unnecessary repeated updates on the one hand and the accumulated error in multiple updates on the other hand. While reducing the update frequency, soft updates should be used.

$$\theta' \leftarrow \tau\theta + (1 - \tau)\theta' \tag{12}$$

Target policy smoothing is based on the target policy μ to obtain the target action and add truncated noise to the target action so that the value of a small area around the target action is smooth enough to reduce the generation of errors effectively. The target policy smoothing process can be expressed as:

$$a' = clip(\mu(s') + \varepsilon, a_{low}, a_{high}) \tag{13}$$

where $\varepsilon \sim clip(N(0,\sigma), -c, c)$, a_{low} and a_{high} represent the maximum and minimum values of the action a, respectively. $clip(x, -y, y)$ represents the truncation of each element of x in the interval $[-y, y]$.

3.2. LSTR Algorithm

The commonly used risk indicators in portfolio problems are the Sharpe ratio, maximum drawdown, etc. Still, they cannot respond to the market risk situation promptly, therefore Bai et al. proposed a long- and short-term risk control algorithm to redefine risk and improve risk control ability [12]. The basic principle of this algorithm is to use a particular portfolio algorithm to obtain the investment strategy of each period b_t, and then calculate the long- and short-term risk parameters λ and η. After that, we can make the following risk adjustment:

$$\hat{b}_t = \lambda\eta e_1 + (1 - \lambda\eta)(I - e_1) \odot b_t \tag{14}$$

where $\odot$ denotes element-by-element multiplication and $\eta \in [0, 1]$ indicates the proportion of risk-free assets based on the short-term return of the portfolio strategy, which in this paper refers to the ratio of cash assets, I denotes a column vector with all elements being 1, e_1 is a vector with the first element 1 and the rest 0. $\lambda \in [0, 1]$ denotes a sample of a probability distribution that describes information about the long-term trend based on the market.

The algorithm uses the risk indicator random variable $C(s_t)$ to describe the long-term risk and short-term risk. s_t is used to describe the daily return of the portfolio on the tth trading day and ϕ is to denote the desired daily return set by the investor. $z \geq 0$ represents the maximum loss that the investor can sustain on a single trading day, and then the risk indicator random variable is defined as:

$$C(s_t) = \begin{cases} 0, \phi - s_t > z \\ 1, \phi - s_t \leq z \end{cases} \tag{15}$$

Long-term risk control means the historical data of tth trading day are known and the probability of tth trading days $C = 0$ is predicted. q is the probability of $C = 0$ on tth trading day. The q obeys the Beta distribution. After the daily return s_t of the $t - 1$th trading day is known and the tth trading day is $C = 0$, the posterior probability of q is:

$$P(q|v, N - v) = P(\alpha, \beta + 1) \tag{16}$$

Correspondingly, if the tth trading day is $C = 1$, the posterior probability of q is

$$P(q|v, N - v) = P(\alpha + 1, \beta) \tag{17}$$

where N describes the number of consecutive trading days and v indicates the number of $C = 0$ in this successive trading day. α and β are parameters of the Beta distribution. Since the long-term risk control parameter $\lambda \sim P(q|v, N - v)$, as the number of trading days $C = 1$ increases, i.e., the daily return s_t is smaller than the investor's preset loss, i.e., the portfolio strategy performs well. Then, the Beta distribution becomes right skewed, λ becomes larger, and the proportion of risk-free assets increases when substituted into the portfolio update formula. The long-term risk parameter λ can take the value of:

$$\lambda^E = E[q] = \frac{\alpha}{\alpha + \beta} \tag{18}$$

The short-term risk parameter η represents the proportion of risk-free assets in the portfolio. Changes in the algorithm's short-term returns can be reflected by the number of consecutive occurrences of the risk indicator variable $C = 1$, which, as the number of successive events increases, implies that the short-term daily losses are within acceptable

limits for the investor. During this period, the short-term risk parameter η needs to decrease rapidly to near zero.

$$\eta = \frac{1}{1 + \exp\{\kappa + \tau\}} \tag{19}$$

In the above equation, κ indicates the number of consecutive trading days in which $C = 1$ occurs. The τ is a constant that determines the degree of η has declined. In other words, how much money will enter the market. The framework of the algorithm is shown in Figure 4.

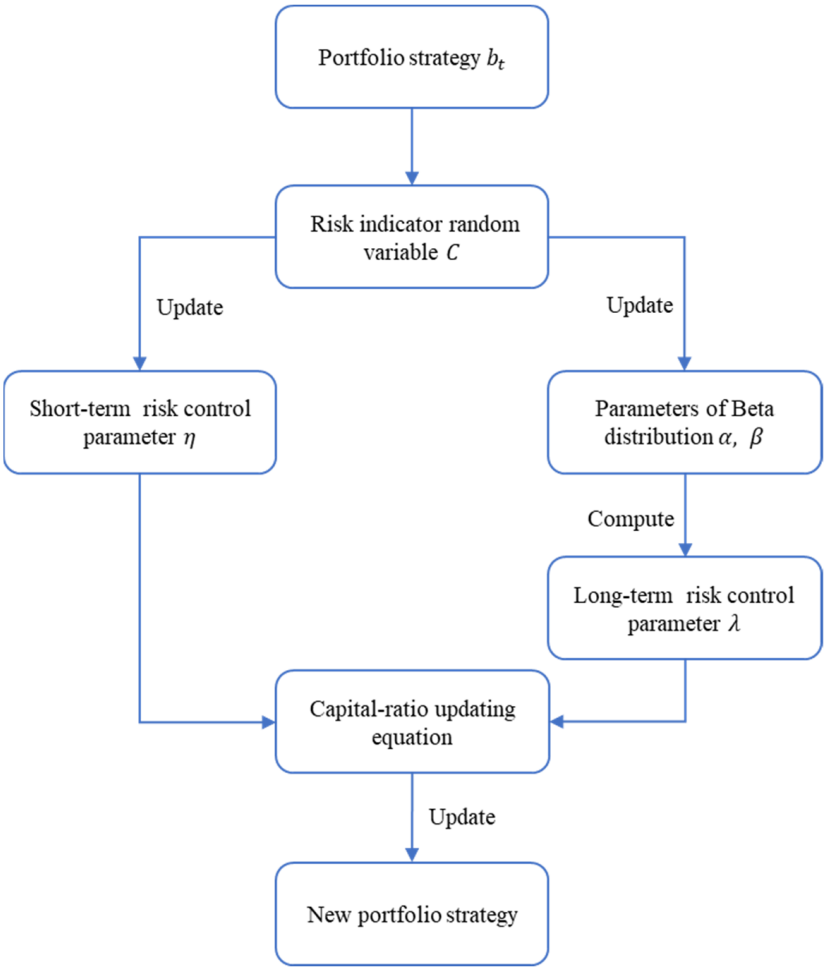

Figure 4. LSTR algorithm flowchart.

3.3. LSTR–TD3 Algorithm

To construct a deep reinforcement learning algorithm with risk control capability and apply it to solve the portfolio management problem, this paper tries to introduce the long- and short-term risk control algorithm into the TD3 algorithm in DRL so that the agent can control risk effectively and obtain better returns. The detailed process of the LSTR–TD3 algorithm is shown in Algorithm 1. Its input hyperparameters include the number of iterations M, discount factor γ, etc. The output is an action vector.

We start by using classic TD3 to obtain the portfolio strategy for each period. Second, we calculate the long-term and short-term risk parameters. Finally, the strategy update formula is used to obtain the final portfolio optimization strategy.

In the process, the LSTR algorithm proposes a new risk definition based on the probability of adverse results and defines the random variables of the risk indicator. Different statistical methods for this variable can represent long-term and short-term risk, respectively. η is a non-linear, rapidly changing short-term risk control parameter. Its local

changes are rapid and can reflect the short-term trading risk of the portfolio strategy. Therefore, the parameter η is used to capture short-term and nonlinear controls to achieve local fast trend reversals. λ is a linear, slow-changing, long-term risk parameter control quantity. For the Beta posterior distribution which λ obeys, depending on the results of the previous trading day, the parameters are $\alpha + 1$ or $\beta + 1$ on the basis of the prior distribution and its likelihood function assumes that the market fits the mean reversion theory, which is more useful in long-term investment. In the strategy update formula, when the product of the two parameters is large, that is, the current risk is larger, the proportion of funds of risk-free assets will be higher, and vice versa. After allocating the proportion of funds to risk-free assets, the remaining funds are allocated to other assets to achieve the purpose of controlling risk.

Algorithm 1 LSTR–TD3 Algorithm

Initialize predictive value networks Q_{w_1} and Q_{w_2}, parameters are w_1 and w_2
Initialize target value networks $Q_{w_1'}$ and $Q_{w_2'}$, parameters are w_1' and w_2'
Initialize predictive policy networks
μ_θ and target policy networks $\mu_{\theta'}$, parameters are θ and θ'
Synchronized parameters $w_1' \to w_1$, $w_2' \to w_2$, $\theta' \to \theta$
The capacity of the experience pool D is N
The number of iterations is M, the discount factor is γ, $\tau = 0.0001$, the number of random mini-batch sampling samples is n
For $e = 1$ to M **do**:
Receive initial state S_0
Repeat each time step in episode $t = 0, 1, 2, \cdots$:
Select actions $A_t = \mu(S_t, \theta) + \varepsilon_t$ based on the current prediction strategy network and exploration noise, where $\varepsilon_t \sim N_t(0, \sigma)$
Calculate the risk control parameters λ and η
Perform risk adjustment according to $\hat{A}_t = \lambda\eta e_1 + (1 - \lambda\eta)(1 - e_1) \odot A_t$
Execute the action $\hat{A}_t$, obtain returns R_{t+1} and the next state S_{t+1}
Convert experience $(S_t, \hat{A}_t, R_{t+1}, S_{t+1})$ to experience pool D
Randomly sampling small batches of n^{th} experience transfer samples from the experience pool D
Actions after perturbation $\widetilde{a}_{i+1} \leftarrow \mu(S_{i+1}, \theta') + \varepsilon_i$, where $\varepsilon_i \sim clip(N_t(0, \widetilde{\sigma}), -c, c)$
Update the target $y_i = R_{i+1} + \gamma \min_{i=1,2} Q(S_{i+1}, \widetilde{a}_{i+1}, w_i')$
Use MBGD method, parameters of policy network w are updated according to the maximization objective function
$\nabla_w L(w) \approx \frac{1}{N} \sum_{i=1}^{N} (y_i - Q(S_i, \widetilde{A}_i, w)) \nabla_w Q(S_i, \widetilde{A}_i, w)$
If t mod d **then**
Use MBGA method, parameters of value network θ are updated according to the minimization loss function
$\nabla_\theta \widetilde{J}_\beta(\theta) \approx \frac{1}{N} \sum_{i=1,2}^{N} \nabla_a Q(S_i, a, w)\Big|_{a=\mu(S_i,\theta)}$
Soft update target network $\begin{cases} w' \leftarrow \tau w + (1-\tau) w' \\ \theta' \leftarrow \tau\theta + (1-\tau)\theta' \end{cases}$
Until $t = T - 1$
End for

4. Experiment Study

4.1. Experiment Setup

This paper selects the CSI500 index with a strong market representation from the Chinese stock market. In order to guarantee the integrity of the training data set, we selected stocks listed before 1 January 2012 from the constituent stocks of the CSI500 index to generate a stock pool. After that, four stocks were randomly selected from the stock pool and used together with the CSI300 as a trading asset portfolio for empirical research in the model. Random selection can ensure that the model setter selects stocks based on subjective factors and has an uncertain impact on the experiment. In order to

avoid accidents, three portfolios were generated and experimented with separately to analyze the experimental effects of the model on different portfolios. This demonstrates the effectiveness of the LSTR–TD3 model constructed in Section 3.3 in solving the problem of stock portfolio management. Additionally, the performance is compared with the TD3 algorithm without risk control, where the transaction cost is 0.0025 for every single asset being traded. The daily closing price changes of stocks in the three experiments are shown in Figures 5–7.

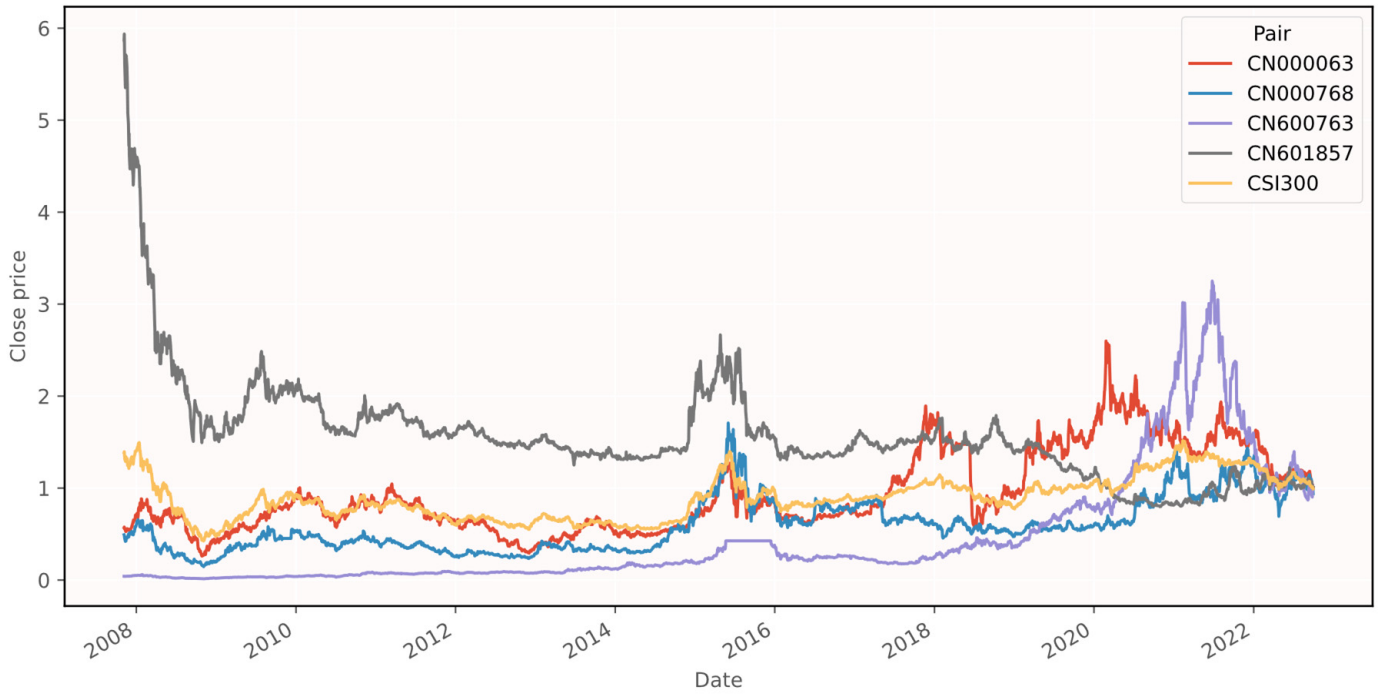

Figure 5. The closing price changes of Portfolio 1.

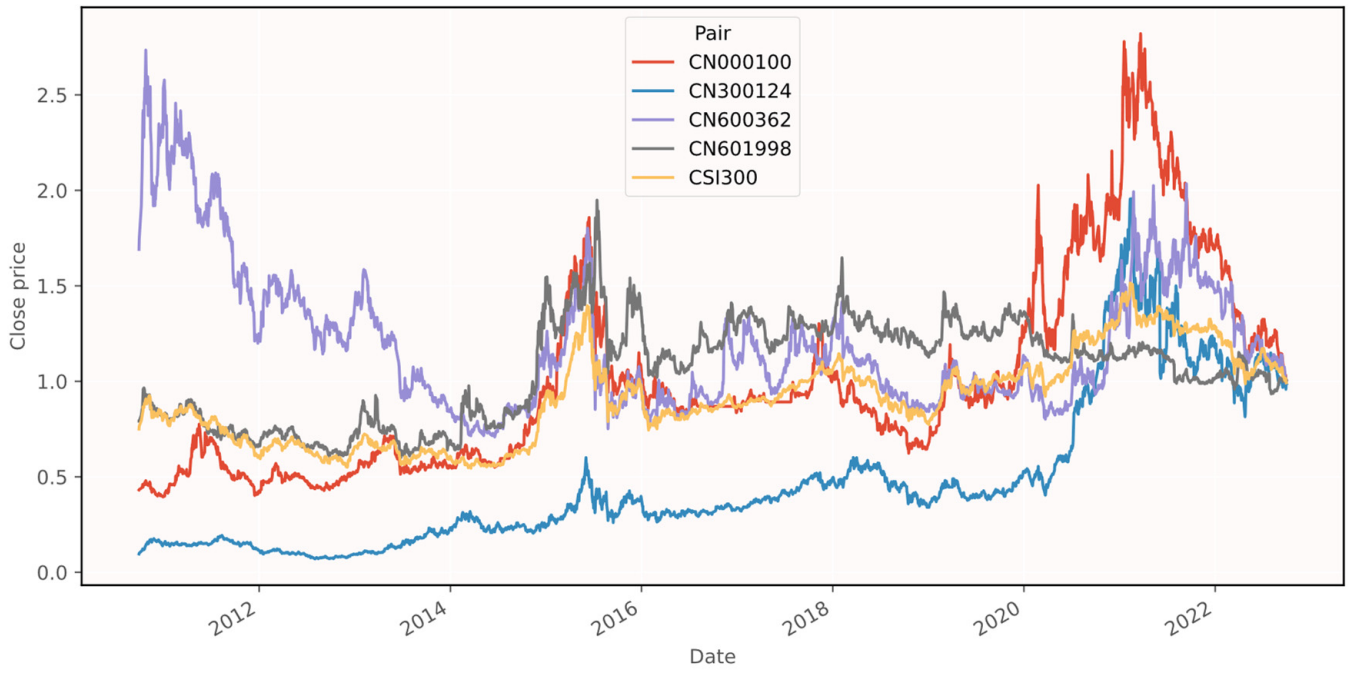

Figure 6. The closing price changes of Portfolio 2.

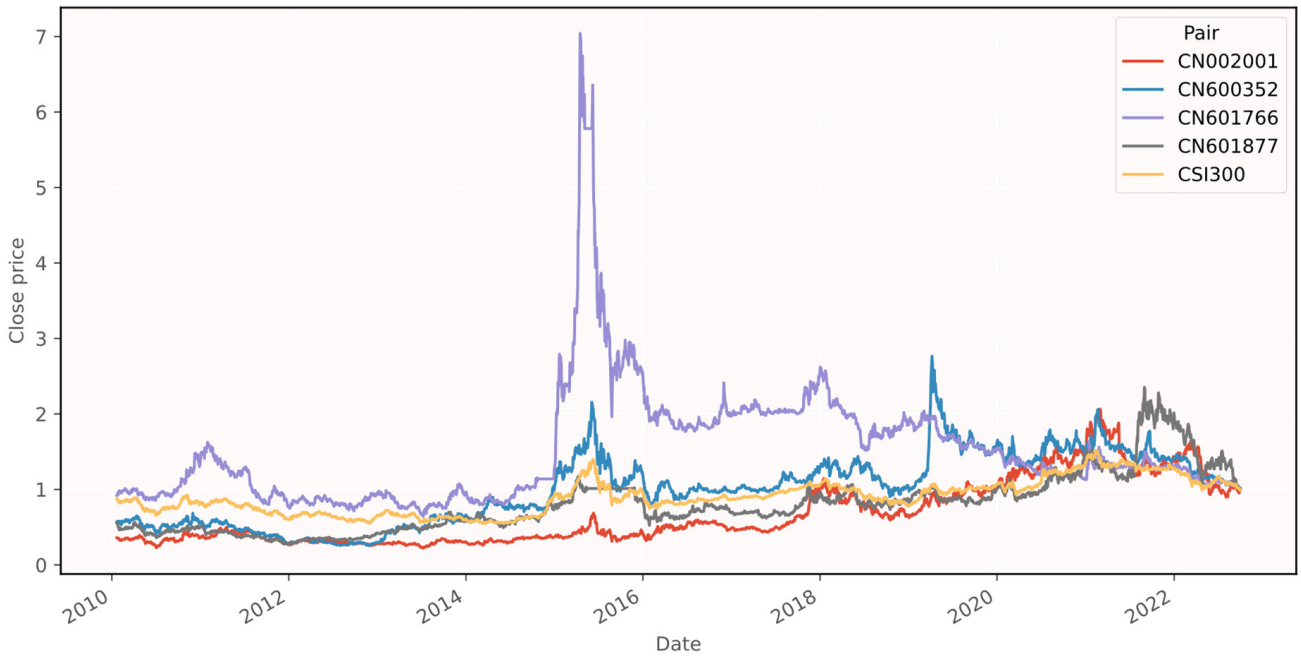

Figure 7. The closing price changes of Portfolio 3.

The data selected for the experiments are from the JoinQuant database. The experimental data in this paper are chosen from daily trading data, including four price features: the closing price, the open price, the high price, and the low price. In addition, it is assumed that all risky assets are sufficiently liquid, each trading is executed immediately, and the transaction does not impact the market. The stock codes selected in each experiment are shown in Table 1.

Table 1. The stock code of different experiments.

Portfolio 1	Portfolio 2	Portfolio 3
600763	000100	601766
601857	600362	601877
000768	300124	002001
000063	601998	300122
CSI300	CSI300	CSI300

4.2. Performance Metrics

Three different metrics are used to evaluate the performance of trading strategies, the first of which is the Accumulated Portfolio Value (APV), which is defined as:

$$\text{APV} = \frac{P_t}{P_0} \tag{20}$$

Here, P_t represents the portfolio value at the end of the test period and P_0 represents the portfolio value at the beginning of the test period.

The second metric is the Sharpe Ratio (SR), which primarily represents the risk-adjusted return of the strategy and is defined as:

$$SR = \frac{\mathrm{E}_t|\rho_t - \rho_{RF}|}{\sqrt{Var(\rho_t - \rho_{RF})}} \tag{21}$$

where ρ_{RF} denotes the risk-free asset return.

The third metric is the maximum drawdown (MDD), which is used to assess the risk resistance of an investment strategy and is defined as:

$$MDD = \max_{t_2 \succ t_1} \frac{V_{t_2} - V_{t_1}}{V_{t_2}} \tag{22}$$

where t_1 and t_2 denote different time steps; this metric represents the maximum loss of portfolio value from peak to trough.

4.3. Algorithm Parameters

The key hyperparameter settings for the DRL training process are listed in Table 2.

Table 2. Primary hyperparameters of LSTR–TD3 algorithm.

Hyperparameters	Value
Critic learning rate	0.001
Critic regularization	None
Actor learning rate	0.0001
Optimizer	Adam
Discount factor	0.99
Exploration noise	0.2
Policy update frequency	2
Batch size	64
Reward scaling	1000

5. Experiment Results

The agents were trained in the training environment and tested separately in the test environment. The actor and critic networks are designed by using the Alexnet convolutional neural network, which is based on the results of Huang et al. [24]. Figure 8 shows the neural network structure of the actor. Figure 9 shows the neural network structure of the critic.

Figures 10–12 show the portfolio value and CSI300 value curves for TD3 and LSTR–TD3 in the three experiments during the testing period, respectively. In the Portfolio 1 data set, LSTR–TD3 shows excellent performance compared with the benchmark CSI300, while TD3 is slightly inferior. In addition, its three performance indicators are better than the TD3 algorithm. In the Portfolio 2 data set, the Sharpe ratio and maximum drawdown are both significantly improved, except for the cumulative portfolio value, which is slightly lower than the original TD3 algorithm. In the Portfolio 3 data set, LSTR–TD3 still outperforms TD3 in three performance indicators, which indicates that the LSTR–TD3 model proposed in this paper can achieve better returns while controlling risk.

Table 3 compares the LSTR–TD3 algorithm proposed in this paper with the TD3 algorithm and the benchmark in the three experiments. Three performance indicators are computed to compare the performance of the LSTR–TD3 algorithm.

Table 3. Experimental results for three different portfolios.

Portfolio	Metrics	TD3	LSTR–TD3	CSI300
Portfolio 1	APV	1.0848	1.4573	1.0069
	SR	0.5679	1.8225	0.0773
	MDD	12.7039	8.5175	8.8823
Portfolio 2	APV	1.1030	1.0859	0.9951
	SR	0.8053	0.8830	0.1151
	MDD	8.3310	5.1889	8.8823
Portfolio 3	APV	0.9493	1.0739	0.9966
	SR	−0.1644	0.5378	0.1287
	MDD	16.5333	15.8147	8.8823

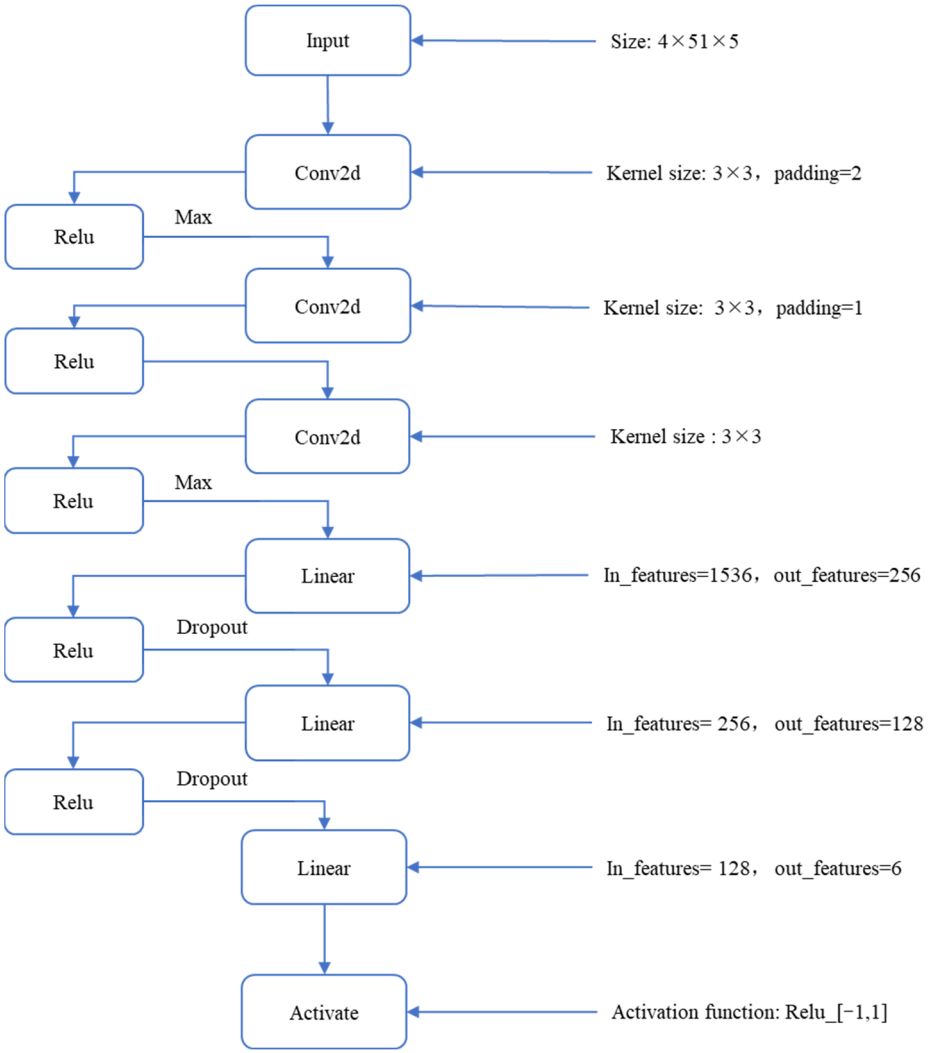

Figure 8. The neural network structure of actor.

Compared with the benchmark, the cumulative portfolio value and Sharpe ratio of both the classical TD3 and the LSTR–TD3 model are higher than the CSI300 index, indicating that classical TD3 and the methodology proposed in this article can be effectively applied to the China A-share market. In addition, the maximum drawdown of the LSTR–TD3 model in Portfolios 1 and 2 can even be lower than the CSI300 index. This shows that the portfolio strategy generated by the model in this paper can be less risky than the CSI300 index. Compared to the classic TD3 algorithm, the LSTR–TD3 model has a better maximum drawdown. At the same time, its cumulative portfolio values are higher than or equal to the classical TD3. Therefore, from the perspective of comprehensive return and risk, the model proposed in this paper with LSTR outperforms the classic TD3 in balancing risk and return. In conclusion, the results show that the LSTR–TD3 is effective in China's A-share market and has better risk management capabilities than the classic TD3.

However, Figure 11 shows that the cumulative portfolio value of the CSI300 index rose sharply at the end of December 2021, while the cumulative portfolio growth of the LSTR–TD3 model was not as fast as the CSI300 Index. This suggests that when the model has a rapid upward signal in the market, the agent fails to seize the opportunity in time to increase the return by buying stocks with large income increases. Therefore, in addition to price information, technical indicators should be added in future work so we can further improve the return of portfolio strategy, and seize the opportunity of the bull market.

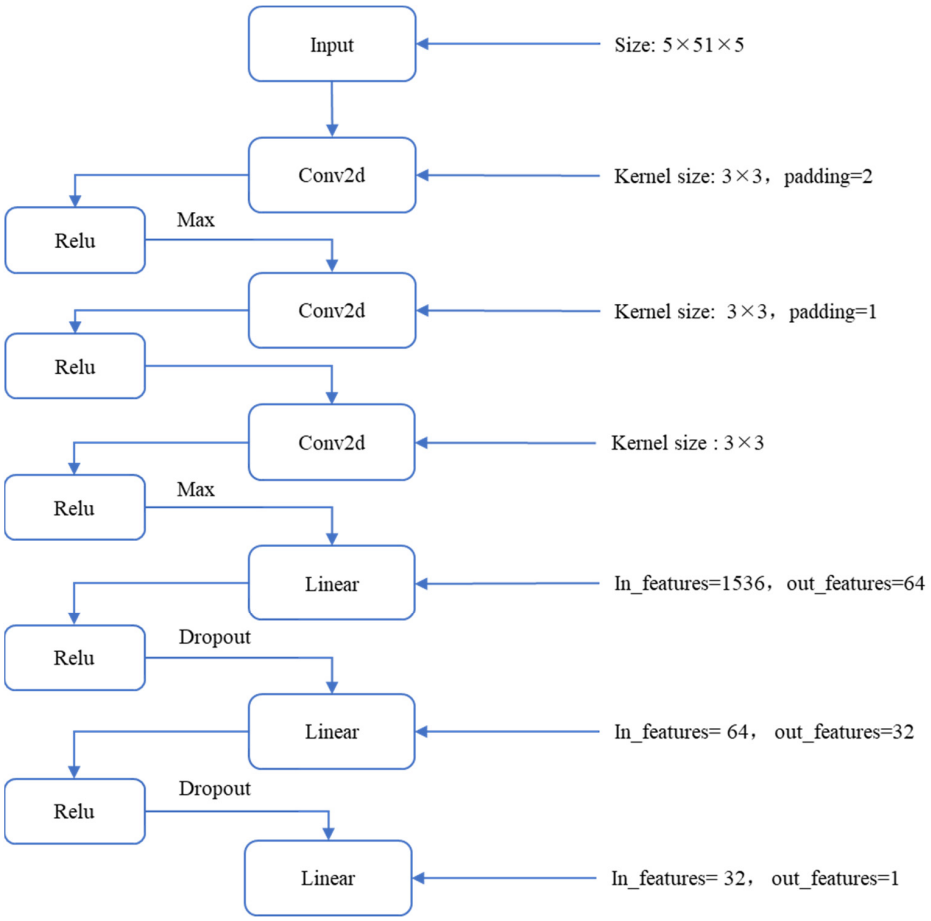

Figure 9. The neural network structure of critic.

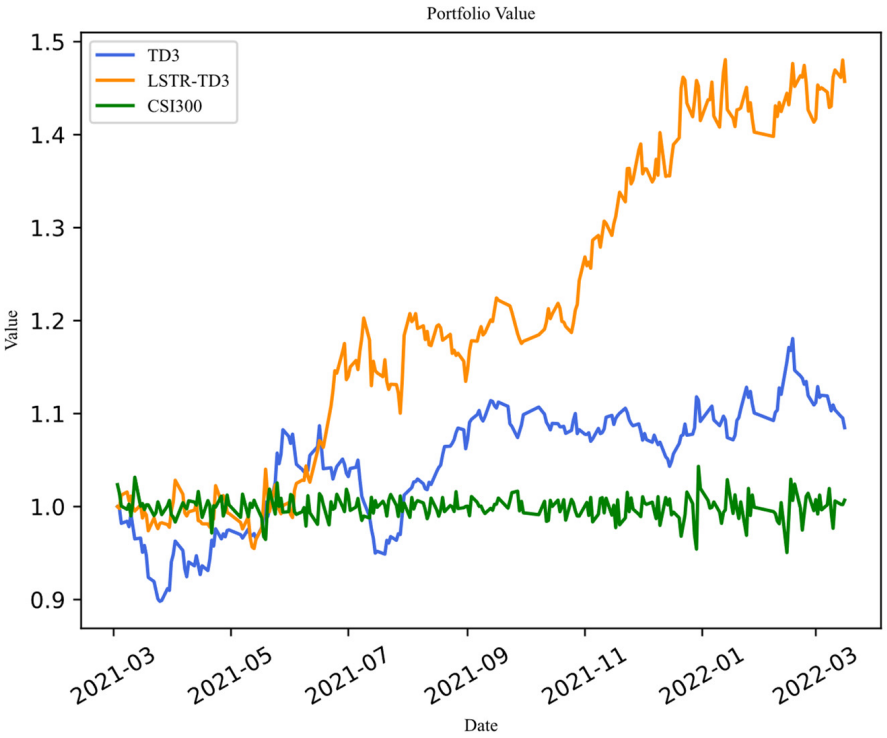

Figure 10. Portfolio value of Portfolio 1.

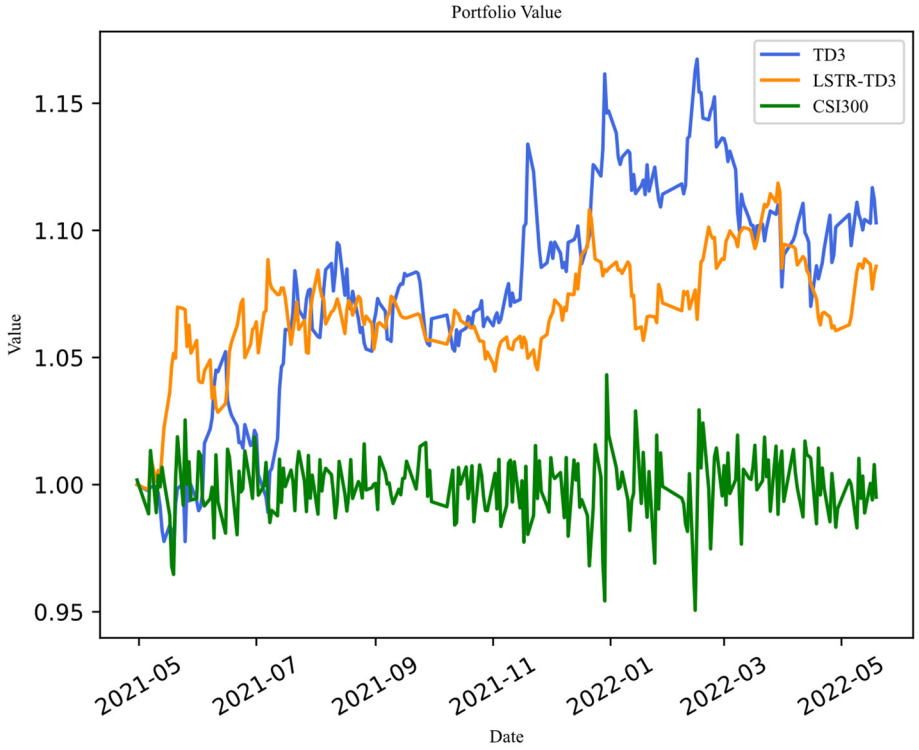

Figure 11. Portfolio value of Portfolio 2.

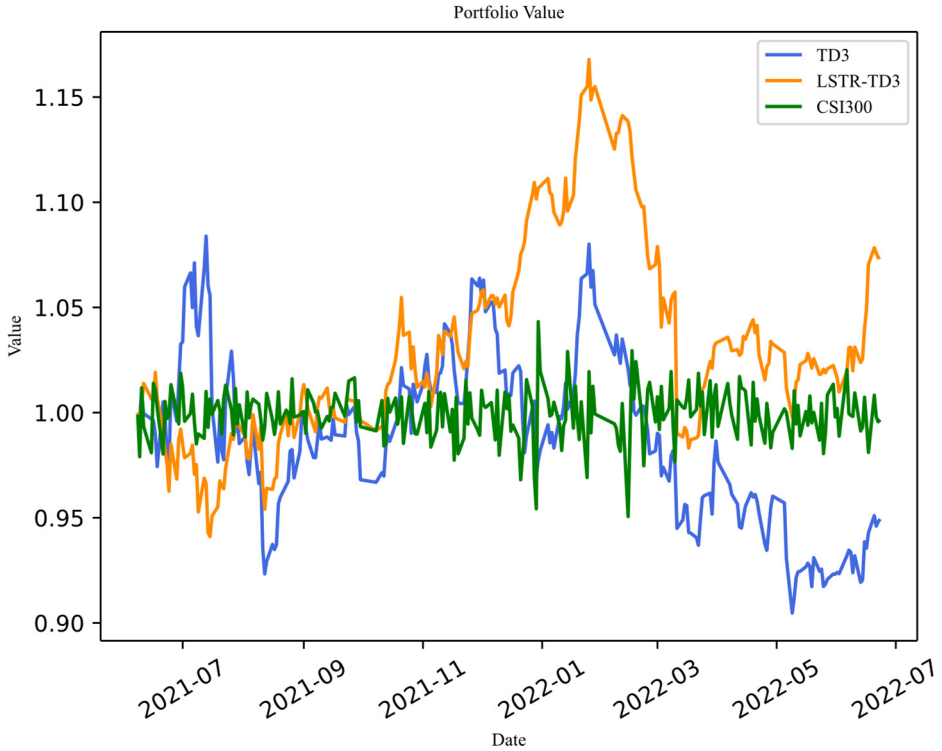

Figure 12. Portfolio value of Portfolio 3.

6. Conclusions

Because of the low signal-to-noise ratio of financial data, investors based on traditional econometric models often miss the opportunity to gain excess returns. In order to beat the market for excess returns, many scholars have used deep reinforcement learning methods to obtain better trading strategies. Despite the success of deep reinforcement

learning in portfolio management problems, the Sharpe ratio and maximum drawdown are excellent measures of risk. Still, they are not applicable for optimizing them as objective functions, which describe long-term risk and do not respond to market conditions promptly. Therefore, this paper explores a deep reinforcement learning algorithm, namely TD3, to find the optimal portfolio strategy in the complex and volatile Chinese stock market. The LSTR algorithm is introduced to optimize the portfolio strategy. While changing the weight of risk-free assets by the LSTR algorithm, the portfolio strategy is adjusted. Compared to the TD3 strategy studied in previous studies, the improved strategy can increase returns while reducing the risk of the strategy. In summary, our approach is more robust than others in balancing risk and reward. In addition, our deep reinforcement learning approach considers not only long-term risks but also short-term risks.

To verify the effectiveness of the proposed method in this paper, an empirical analysis is conducted in the Chinese stock market and a comparative study is performed with the original TD3 algorithm. Three portfolios were randomly selected for the experiment. Their historical price data were collected to express the state of the market. The DRL algorithm proposed in this paper is trained and tested based on these historical data. The experiment results show that the LSTR–TD3 algorithm proposed in this paper has better risk management performance than the TD3 algorithm.

In a real trading system, the investor would be able to set a target return for a single trading day and the maximum loss they can tolerate in a single trading day. Thus, the portfolio risk-averse policy constructed by our proposed model can control risk according to the preferences of different investors. However, this model may be difficult to implement in the practical world. An important reason for this is that asset turnover can be high and the operation of determining portfolio weights can lead to frequent asset switching.

Of course, the algorithm proposed in this paper also has limitations. First, some parameters in the LSTR algorithm need to be set artificially, which seems to go against the original intention of artificial intelligence. In addition, only price features are used to describe the environment during the experiment, which cannot sufficiently represent the state of the market.

Author Contributions: Conceptualization, C.J. and J.W.; methodology, C.J.; software, C.J.; validation; formal analysis, C.J.; investigation, C.J.; resources, J.W.; data curation, C.J.; writing—original draft preparation, C.J.; writing—review and editing, C.J.; visualization, C.J.; supervision, J.W.; project administration, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Markowitz, H. Portfolio Selection. *J. Finance* **1952**, *7*, 77–91. [CrossRef]
2. Elavia, T.; Kothari, S.P.; Li, X.; You, H. Gains from Markowitz Optimization: Evidence from Reoptimization of Mutual Fund Holdings. *J. Portf. Manag.* **2022**, *48*, 199–218. [CrossRef]
3. Agarwal, A.; Hazan, E.; Kale, S.; Schapire, R.E. Algorithms for portfolio management based on the Newton method. In Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, USA, 25–29 June 2006; pp. 9–16.
4. Cover, T.M. Universal Portfolios. *Math. Finance* **1991**, *1*, 1–29. [CrossRef]
5. Helmbold, D.P.; Schapire, R.E.; Singer, Y.; Warmuth, M.K. On-Line Portfolio Selection Using Multiplicative Updates. *Math. Finance* **1998**, *8*, 325–347. [CrossRef]
6. Huang, D.; Yu, S.; Li, B.; Hoi, S.C.H.; Zhou, S. Combination Forecasting Reversion Strategy for Online Portfolio Selection. *ACM Trans. Intell. Syst. Technol.* **2018**, *9*, 1–22. [CrossRef]
7. Uziel, G.; El-Yaniv, R. Online Learning of Portfolio Ensembles with Sector Exposure Regularization. *arXiv* **2016**, arXiv:1604.03266.
8. Huang, X. Portfolio selection with a new definition of risk. *Eur. J. Oper. Res.* **2008**, *186*, 351–357. [CrossRef]
9. Mohr, E.; Dochow, R. Risk management strategies for finding universal portfolios. *Ann. Oper. Res.* **2017**, *256*, 129–147. [CrossRef]

10. Shen, W.W.; Wang, J.; Jian, Y.G.; Zha, H. Portfolio Choices with Orthogonal Bandit Learning. In Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, 25–31 July 2015; pp. 974–980.
11. Yue, H.; Liu, J.; Tian, D.; Zhang, Q. A Novel Anti-Risk Method for Portfolio Trading Using Deep Reinforcement Learning. *Electronics* **2022**, *11*, 1506. [CrossRef]
12. Bai, Y.; Yin, J.; Ju, S.; Chen, Z.; Huang, J.Z. Long and Short Term Risk Control for Online Portfolio Selection. In *International Conference on Knowledge Science, Engineering and Management*; Springer: Cham, Switzerland, 2020; pp. 472–480.
13. Moody, J.; Saffell, M.; Liao, Y.; Wu, L. Reinforcement Learning for Trading Systems and Portfolios: Immediate vs Future Rewards. In *Decision Technologies for Computational Finance: Proceedings of The Fifth International Conference Computational Finance*; Refenes, A.-P.N., Burgess, A.N., Moody, J.E., Eds.; Springer: Boston, MA, USA, 1998; pp. 129–140.
14. Dempster, M.A.H.; Leemans, V. An automated FX trading system using adaptive reinforcement learning. *Expert Syst. Appl.* **2006**, *30*, 543–552. [CrossRef]
15. Deng, Y.; Bao, F.; Kong, Y.; Ren, Z.; Dai, Q. Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 653–664. [CrossRef] [PubMed]
16. Almahdi, S.; Yang, S.Y. An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent rein-forcement learning with expected maximum drawdown. *Expert Syst. Appl.* **2017**, *87*, 267–279. [CrossRef]
17. Bertoluzzo, F.; Corazza, M. Reinforcement Learning for Automatic Financial Trading: Introduction and Some Applications. *Univ. Ca'Foscari Venice Dept. Econ. Res. Pap. Ser. No* **2012**, *33*. [CrossRef]
18. Zhang, Z.; Zohren, S.; Roberts, S.J. Deep Reinforcement Learning for Trading. *J. Financ. Data Sci.* **2020**, *2*, 25–40. [CrossRef]
19. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
20. Jiang, Z.; Liang, J. Cryptocurrency portfolio management with deep reinforcement learning. In Proceedings of the 2017 Intelligent Systems Conference (IntelliSys), London, UK, 7–8 September 2017; pp. 905–913. [CrossRef]
21. Liu, X.-Y.; Xiong, Z.; Zhong, S.; Yang, H.; Walid, A. Practical Deep Reinforcement Learning Approach for Stock Trading. *arXiv* **2018**, arXiv:1811.07522.
22. Wang, M.F.; Ku, H. Risk-sensitive policies for portfolio management. *Expert Syst. Appl.* **2022**, *198*, 116807. [CrossRef]
23. Fujimoto, S.; Hoof, H.; Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
24. Huang, G.; Zhou, X.; Song, Q. Deep reinforcement learning for portfolio management based on the empirical study of chinese stock market. *arXiv* **2020**, arXiv:2012.13773. [CrossRef]