sensors

MDPI

Article

LSTM-DDPG for Trading with Variable Positions

Zhichao Jia [1], Qiang Gao [1,2,*] and Xiaohong Peng [3]

[1] School of Electronics and Information Engineering, Beihang University, Beijing 100191, China; jia313682644@buaa.edu.cn
[2] Hangzhou Innovation Institute, Beihang University, Hangzhou 310051, China
[3] Faculty of Computing, Engineering and the Built Environment, Birmingham City University, Birmingham B5 5JU, UK; xhpeng100@gmail.com
* Correspondence: gaoqiang@buaa.edu.cn

Abstract: In recent years, machine learning for trading has been widely studied. The direction and size of position should be determined in trading decisions based on market conditions. However, there is no research so far that considers variable position sizes in models developed for trading purposes. In this paper, we propose a deep reinforcement learning model named LSTM-DDPG to make trading decisions with variable positions. Specifically, we consider the trading process as a Partially Observable Markov Decision Process, in which the long short-term memory (LSTM) network is used to extract market state features and the deep deterministic policy gradient (DDPG) framework is used to make trading decisions concerning the direction and variable size of position. We test the LSTM-DDPG model on IF300 (index futures of China stock market) data and the results show that LSTM-DDPG with variable positions performs better in terms of return and risk than models with fixed or few-level positions. In addition, the investment potential of the model can be better tapped by the reward function of the differential Sharpe ratio than that of profit reward function.

Keywords: trading strategy; deep reinforcement learning; variable positions; reward function

check for updates

Citation: Jia, Z.; Gao, Q.; Peng, X. LSTM-DDPG for Trading with Variable Positions. *Sensors* **2021**, *21*, 6571. https://doi.org/10.3390/s21196571

Academic Editor: Friedhelm Schwenker

Received: 1 September 2021
Accepted: 29 September 2021
Published: 30 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Over the past decade, machine learning techniques have driven significant advances across many application areas which inspire investors and financial institutions to develop machine learning-aided investment strategies. More and more studies have applied machine learning to financial investment [1–6]. However, the highly non-stationary nature of financial markets hinders the application of typical data-hungry machine learning methods in financial investment.

Supervised learning is suitable for portfolio strategies because sufficient data can be prepared when investing in a portfolio. Imajo et al. [7] propose a system for constructing portfolios with a spectral decomposition-based method to hedge out common market factors and a distributional prediction method based on deep neural networks incorporating financial inductive biases. Nakagawa et al. [8] propose a principled stock return prediction framework called Ranked Information Coefficient Neural Network (RIC-NN) to alleviate the overfitting. The learning difficulties of initialization, the stopping of training models and the transfer among different markets have been addressed. Pun et al. [9] present a financial thought experiment with the aid of the Generative adversarial network (GAN) framework and adapt it to the portfolio risk minimization problem by adding a regression network to GAN. Tsang et al. [10] investigate a deep-learning solution to high-dimensional multiperiod portfolio optimization problems with bounding constraints on the control.

Reinforcement learning is suitable for trading strategies while supervised learning is suitable for portfolio strategies. The trading process can be considered as a multistep decision process and the reinforcement learning framework is suitable for solving trading problems [11–15]. In reinforcement learning-aided trading, an agent interacts with the

market environment and simultaneously makes trading decisions without any supervised information [16–22].

Neuneier [23] uses Q-learning to make trading decisions of position direction, in which the Q-learning model optimizes the trading policy by sampling state-action pairs and returns while interacting with market conditions. Moody et al. [15] propose a Recurrent Reinforcement Learning (RRL) model to make trading decisions. The RRL model takes the previous action as part of the input to properly take into account the effects of transactions costs. Deng et al. [11] introduce a Recurrent Deep Neural Network (RDNN) for simultaneous environment sensing and recurrent decision making for online financial assert trading. The bulk of the RDNN is composed of two parts of recurrent neural network (RNN) for reinforcement learning. Liu et al. [24] propose imitative recurrent deterministic policy gradients to automatically develop trading strategies by an intelligent trading agent. In addition, imitation learning is introduced to create a balance between the exploration and exploitation of strategy.

The aforementioned studies mainly focus on making decisions of position direction. A fixed position size or the maximum position size is assumed explicitly or implicitly in these studies. Both the position direction and position size should be determined based on the market condition during trading. To achieve high profit with low risk, the position size should vary with market conditions. When the market has a large and steady upward or downward trend and the market condition is easy to identify, we should set a large position size to obtain high profits. When the market is volatile, a small position size or even zero size is preferred to reduce trading risk.

Recently, some scholars considered changeable position sizes during trading. Li et al. [25] proposed a position-controlled action space. The action space is extended to $\{-3, -2, -1, 0, 1, 2, 3\}$, which represents the position held in the next state. Jeong et al. [26] predict the position size by adding a DNN regressor to a deep Q-network. Besides, the position size is limited by a maximum position size of 10. These studies only consider a few levels or limited levels of position size, and there is no work considering variable position sizes as far as we know. If the position size can change arbitrarily with subtle variations of market condition, the efficiency of investment capital can be improved and the trading risk can be reduced further.

In this paper, we propose an LSTM-DDPG model to make trading decisions with variable positions. To be specific, we describe the trading process as a Partially Observable Markov Decision Process (POMDP) with the acknowledgement that the financial market environment is not completely observable. The LSTM-DDPG model is composed of a long short-term memory (LSTM) network and deep deterministic policy gradient (DDPG) framework. The LSTM is used to extract environmental state features from environmental observations and the DDPG is used to make trading decisions. The DDPG consists of a critic network and an actor network. The critic network estimates the action-value function and the actor network adjusts the deterministic policy which outputs a continuous action at each step. The action is a real number in the range of $[-1, +1]$. The sign of the action (+, −) represents the position direction, and the absolute value represents the position size with variable amounts. In addition, we consider two different reward functions: DSR and profit. The experimental results of our model on IF300 (index futures of the Chinese stock market) data show that the model can achieve well-balanced trading performance between profit and risk factors.

The remaining parts of this paper are organized as follows. Section 2 describes the details of the proposed LSTM-DDPG model. Section 3 is the experimental part where we evaluate the performance of our model. Section 4 concludes this paper.

2. Methodology

We look the trading process as a POMDP, and propose the LSTM-DDPG model to solve it.

2.1. Partially Observable Markov Decision Process

In a financial market, the security price is influenced by macroeconomic policies and microeconomic activities, which contain the information on unpredictable events and trading behaviors of all the market participants. Therefore, it is difficult to model the true financial market from the perspective of an investor or a trading agent and the trading process can be viewed as a Partially Observable Markov Decision Process (POMDP).

A POMDP is a tuple $S, A, T, R, \Omega, O, \gamma$.Here S is a set of states, and $s_t \in S$ is the state at time step t. s_t is not well known by the trading agent. A is a set of actions, and $a_t \in A$ represents the action at step t. We define the trading action as a continuous variable within a range of $[-1, +1]$, which represents the variable positions. The direction of variable positions is represented by the sign of the action, i.e., '+' means long positions, '−' means short positions and '0' means no holding. The size of variable positions is represented by the absolute value of the action and is measured as a percentage of the amount of total capital. For example, $a_t = +0.6$ means we have a long position with the 60% total capital invested in the financial market. T is a state transition matrix, which consists of conditional transition probabilities between states. R is the reward function. $R_t = R\left(s_t, a_t, s'_{t+1}\right)$ represents the instantaneous reward at step t after executing the action a_t. Ω is a set of observations and O is a set of conditional observation probabilities. In our study the closing prices of the past 20 trading days are inputted into a LSMT network as the observation, and the output h_t is the modelling of market state. γ is the discount factor ranging from 0 to 1, which is used to calculate the future discount rewards.

Optimizing investment is essentially a multi-objective optimization problem that requires maximizing profits and minimizing risks. The profit and DSR are taken as reward functions separately in our trading model. The profit reward function considers only profit, while the reward function of DSR takes both profit and risk into account.

Considering the transaction fees and slippage, the profit r_t is defined as

$$r_t = \frac{K}{p_t}\{(p_{t+1} - p_t)a_t - \delta p_t |a_t - a_{t-1}|\}. \tag{1}$$

where K represents the total capital. p_t and a_t are the price and the trading action at time t. δ is a parameter which accounts for transaction fees and slippage. When the reward function is profit, the expected future discount rewards are cumulative return.

When the reward function is DSR, the expected future discount rewards is related to the Sharpe ratio. The Sharpe ratio S_t is defined as

$$S_t = \frac{E[r_i]}{\sigma[r_i]}, i \leq t \tag{2}$$

where $E[r_i]$ is the mean of profits and $\sigma[r_i]$ is the standard deviation of profits which represents the volatility of profits and the trading risk (see [15]).

Expanding the Sharp ratio to the Taylor series in the adaptation rate η, we have

$$S_t \approx S_{t-1} + \eta \frac{dS_t}{d\eta}\bigg|_{\eta=0} + o\left(\eta^2\right). \tag{3}$$

Noting that only the first-order term in this expansion depends upon the return r_t at time t, so the DSR d_t (see [15]) can be defined as

$$d_t \equiv \frac{dS_t}{d\eta} = \frac{B_{t-1}\Delta A_t - \frac{1}{2}A_{t-1}\Delta B_t}{\left(B_{t-1} - A_{t-1}^2\right)^{\frac{3}{2}}}. \tag{4}$$

In this expression, A_t and B_t are the estimations of exponential moving average for the first and second moments of r_t, ΔA_t and ΔB_t are their update quantities. They can be written as

$$A_t = A_{t-1} + \eta \Delta A_t = A_{t-1} + \eta(r_t - A_{t-1}), \tag{5}$$

$$B_t = B_{t-1} + \eta \Delta B_t = B_{t-1} + \eta\left(r_t^2 - B_{t-1}\right), \tag{6}$$

$$\Delta A_t = r_t - A_{t-1}, \tag{7}$$

$$\Delta B_t = r_t^2 - B_{t-1}. \tag{8}$$

2.2. LSTM-DDPG

We propose the LSTM-DDPG model to solve the POMDP for trading, which is shown in Figure 1. The LSTM-DDPG is composed of the LSTM network and the DDPG framework. The LSTM network is used to extract the market features and the DDPG framework is used to make trading decisions.

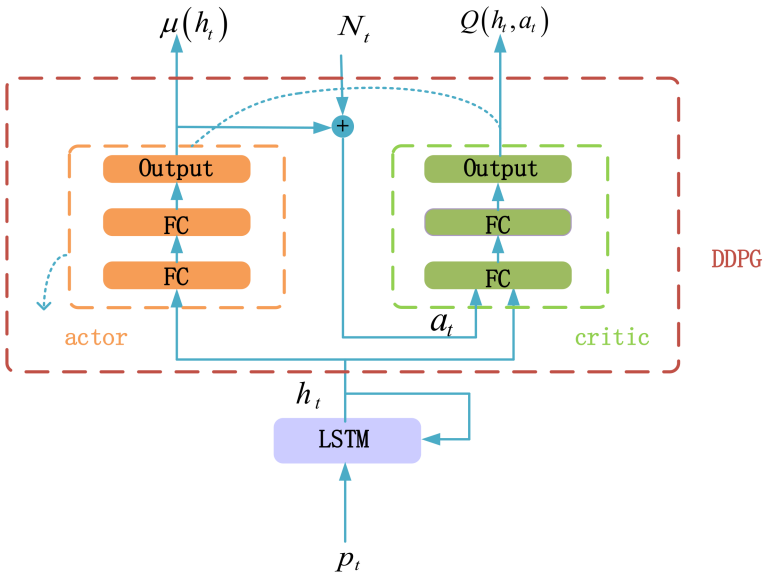

Figure 1. LSTM-DDPG model.

2.2.1. LSTM

LSTM is a special RNN that can learn the long-term dependency within the input data. The past closing prices of T = 20 trading days are looked as the observations of financial market and are input into LSTM, and the output of LSTM represents the market state, hence the LSTM is unrolled for T time steps. LSTM can be formulated as follows

$$f_t = \Lambda\left(W_f \cdot [h_{t-1}, p_t] + b_f\right), \tag{9}$$

$$i_t = \Lambda(W_i \cdot [h_{t-1}, p_t] + b_i), \tag{10}$$

$$\widetilde{c}_t = \tan h(W_c \cdot [h_{t-1}, p_t] + b_c), \tag{11}$$

$$c_t = f_t * c_{t-1} + i_t * \widetilde{c}_t, \tag{12}$$

$$o_t = \Lambda(W_o \cdot [h_{t-1}, p_t] + b_o), \tag{13}$$

$$h_t = o_t * \tan h(c_t), \tag{14}$$

where p_t is the closing price of market at time t. h_t is the output vector, f_t is the forget gate, i_t is the input gate and o_t is the output gate, c_t is the cell state, $\widetilde{c}_t$ is the update value of cell state, W_* are weight matrices, b_* are bias vectors. $\Lambda(x)$ is the sigmoid function and defined as $\Lambda(x) = \frac{1}{1+e^{-x}}$, which guarantees that the values of the gates are in the range of

0 to 1. The input gate i_t gives the information that needs to be stored in the cell state c_t and the forget gate f_t controls the information which needs to be forgot from the last cell state c_{t-1} to the current cell state c_t. The output gate o_t is used to generate the output vector h_t from the cell state c_t. Through the control of forget gate, input gate and output gate in the network, features are extracted according to the timing relationship of the input data.

2.2.2. DDPG

The DDPG framework is used to make trading decisions based on the market features captured by LSTM. The DDPG framework includes both the critic network and the actor network, and both are composed of two fully connected layers (FC) and one output layer, as shown in Figure 1. The critic network fits the action-value function. The actor network adjusts the trading policy by ascending the gradient of the action-value function.

The DDPG framework stores transitions (o_t, a_t, R_t, o_{t+1}) in the prioritized replay buffer during model training and then extracts transitions from the buffer to update the model parameters. To improve the exploration efficiency, Gaussian noise N_t is added to the output of actor network $\mu(h_t)$ to construct the action

$$a_t = \mu(h_t) + N_t. \tag{15}$$

The probability V_t of a transition being sampled in the prioritized replay buffer is related to its priority v_t.

$$V_t = \frac{v_t}{\sum_{i=1}^{t} v_i} \tag{16}$$

The transition priority is defined as follows

$$v_t = \left| y_t - Q\left(h_t, a_t \middle| \theta^Q\right) \right| + \varepsilon, \tag{17}$$

where θ^Q is the vector of parameters of the critic network, ε is a small positive constant to prevent v_t from being zero. Q is the action-value function. y_t is the estimation of cumulative return, which can be calculated as follows

$$y_t = R_t + \gamma Q'\left(h_{t+1}, \mu'\left(h_{t+1} \middle| \theta^{\mu'}\right) \middle| \theta^{Q'}\right), \tag{18}$$

where θ^μ is the vector of parameters of the actor network. The vectors of parameters of the actor target network and the critic target network $\theta^{\mu'}$ and $\theta^{Q'}$ are recursively updated as follows

$$\theta^{\mu'} = \tau\theta^\mu + (1-\tau)\theta^{\mu'}, \tag{19}$$

$$\theta^{Q'} = \tau\theta^Q + (1-\tau)\theta^{Q'}. \tag{20}$$

where τ is the renewal factor which affects the update rate of target networks.

The loss function of critic network is

$$L = E\left[w_t * \left(y_t - Q\left(h_t, a_t \middle| \theta^Q\right)\right)^2 \right], \tag{21}$$

where w_t is the importance sampling weights of a transition, which is defined as

$$w_t = (N * V_t)^{-\beta}, \tag{22}$$

where N is the batch size and β is a constant.

The actor network updates its parameters in the direction of the action-value gradient. The gradient $\nabla_{\theta_\mu} J$ is defined as follows

$$\nabla_{\theta_\mu} J = E\left[\nabla_a Q\left(h, a \middle| \theta^Q\right)\Big|_{h=h_i, a=\mu(h_i)} \nabla_{\theta_\mu} \mu(h|\theta^\mu)\Big|_{h=h_i} \right]. \tag{23}$$

2.3. Training Process of LSTM-DDPG

The detailed process to train the LSTM-DDPG model is described in Algorithm 1. The LSTM network and DDPG framework in the model are trained jointly. The parameters of the LSTM network are updated according to the loss passed back from the critic network. In the training process, when the total return of the training set for the latest epochs tends to be stable, the model is considered to be converged and the training is stopped. The set of parameters obtained during training will be used in the test data by the model for trading.

Algorithm 1 Training Process of LSTM-DDPG

Input: Closing prices $p_1, \ldots, p_t, \ldots, p_M$
Initialization: Initialize the parameters of LSTM network θ^L;
Initialize the parameters of actor network and actor target network θ^μ, $\theta^{\mu'}$;
Initialize the parameters of critic network and critic target network θ^Q, $\theta^{Q'}$;
Initialize the batch size N, size of prioritized replay buffer, discount factor γ, renewal factor of target networks τ, parameter for importance sampling weights β, learning rate of actor network, learning rate of critic network, parameter accounting for transaction fees and slippage δ;
Initialize the prioritized replay buffer;
epoch = 0;

repeat:
  for $t = 1 \ldots M$ **do**
    Update o_t;
    Output the feature of market h_t by the LSTM;
    Output the trading action $\mu(h_t)$ by the actor network according to h_t;
    Add Gaussian noise to $\mu(h_t)$ to construct the action a_t;
    Update o_{t+1} and calculate the profit R_t;
    Store transition (o_t, a_t, R_t, o_{t+1}) in the prioritized replay buffer;
    Sample a minibatch of transitions from the prioritized replay buffer;
    Update θ^Q, θ^L according to Equation (21) based on Adam;
    Update θ^μ according to Equation (23) based on Adam;
    Update $\theta^{\mu'}$, $\theta^{Q'}$ according to Equations (19) and (20);
  end
  Calculate the total return of training set;
  epoch = epoch + 1;
until convergence.

3. Experiments

We conducted experiments to test our model. In this section, the experimental setup is represented. Then the performance of the reward functions of DSR and profit in LSTM-DDPG are compared. We also compare the performance of LSTM-DDPG when the fixed, few-levels, and variable position sizes are employed.

The direction and size of position should be determined in trading decisions based on the market conditions. The trading action a_t takes the value from $[-1, +1]$. The sign of a_t represents the position direction and the absolute value represents the position size. For the trading of fixed position sizes a_t can only take values $\{-1, 0, +1\}$, i.e., there are three actions in the trading: investing all the capital in a long position, investing all the capital in a short position, holding no position. This is the case studied by most researches. For the trading of few-levels position sizes a_t can take values $\{-1, -0.5, 0, +0.5, +1\}$, i.e., the trading system can take two more actions, investing half the capital in long or short positions. This is similar to the studies in [25,26]. For the trading of variable position sizes in the proposed LSTM-DDPG model, the position size changes continuously with the variation of market condition and a_t is allowed to take any real value between -1 and $+1$.

3.1. Experimental Setup

The proposed LSTM-DDPG trading model was tested on the China IF300, which is calculated based on the prices of the top 300 stocks from both the Shanghai and Shenzhen exchange centers. We use daily closing prices over 18 years spanning July 2002 to June 2020, which are shown in Figure 2. The data set is divided into a training set from July 2002 to June 2014, a validation set from July 2014 to June 2017, and a test set from July 2017 to June 2020.

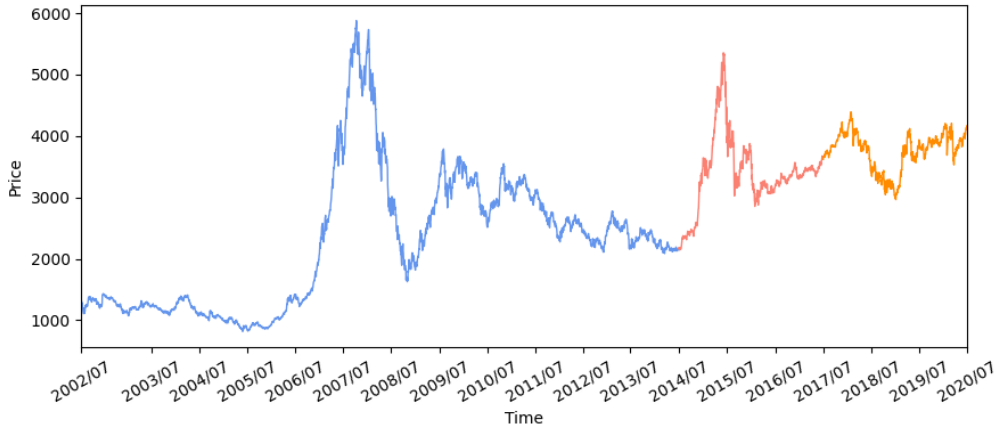

Figure 2. Daily closing price of IF300.

The output of LSTM-DDPG is the trade action and the inputs are the closing prices of the previous 20 trading days. In LSTM-DDPG, the node number of the LSTM layer is set to 64. Both the actor network and the critic network have one output layer and two hidden layers, with 64 and 32 hidden nodes, respectively.

The other hyperparameters used in the experiments are summarized in Table 1. The optimal batch size of 128 is set to balance the gradient oscillation and falling into a local minimum. The values of the prioritized replay buffer size, the discount factor, the renewal factor of target networks and the parameter for importance sampling weights are chosen by pre-training.

The learning rates for the actor network and the critic network are set according to [21]. The Adam optimizer is used for training. The transaction fees set by the futures exchange is 0.0023%, The parameter accounting for transaction fees and slippage is set to 0.01% in the experiments.

Table 1. Hyperparameters of LSTM-DDPG.

Hyperparameter	Value
batch size N	128
size of prioritized replay buffer	1.0×10^5
discount factor γ	0.8
renewal factor of target networks τ	0.01
parameter for importance sampling weights β	0.8
learning rate of actor network	1.0×10^{-4}
learning rate of critic network	1.0×10^{-3}
parameter accounting for transaction fees and slippage δ	0.01%

The proposed model is built and run on TensorFlow 2.3.1, a machine learning platform. The programming language is python 3.6.12. The LSTM-DDPG model is trained and evaluated on a server with two Intel Xeon Gold 6226R CPUs, two NVIDIA RTX 2080 Ti GPUs and 128 GB RAM.

Evaluation metrics in this study are total return rate, Sharpe ratio and maximum drawdown. The risk-return tradeoff is the trading principle, which states that the potential

return rises with an increase in risk. Thus, a trading method can be assessed from the perspectives of profit and risk. The total return rate focuses on profit. The maximum drawdown emphases risk. The Sharpe ratio characterizes how well the return of a trading method compensates the investor for the risk taken.

Total return rate (TR) is the ratio of the return during the trading period, which can be formulated as

$$TR = \frac{\sum_t r_t}{K}, \tag{24}$$

where K represents the initial capital and r_t is the profit during the tth sampling interval of the trading process.

The Sharpe ratio (SR) considers both profit and risk, which reflects the profitability under the unit trading risk and is defined as Equation (2).

The maximum drawdown (MDD) describes the worst case in the process of trading, which reflects the trading risk and is generally related to the volatility of profits, i.e., the standard deviation of return $\sigma[r]$. The MDD is calculated as follows

$$MDD = MAX\left(\frac{K_{t_i} - K_{t_j}}{K_{t_i}}\right), t_i < t_j, \tag{25}$$

where K_{t_i} is the capital at time t_i.

3.2. Experimental Results

3.2.1. Profit vs. DSR as Reward Function in LSTM-DDPG

We conducted experiments to compare the performance of the LSTM-DDPG with variable positions, in which the profit and DSR are taken as reward functions separately. Figure 3 illustrates the profit curves of Buy and Hold, the LSTM-DDPG with the reward functions of DSR, and profit for the test period from 2017 to 2020. Specifically, buy and hold refers to the trading method whereby we take a long position at the beginning and hold the position until the end of the test period. The profit curve of buy and hold also represents the IF300 itself.

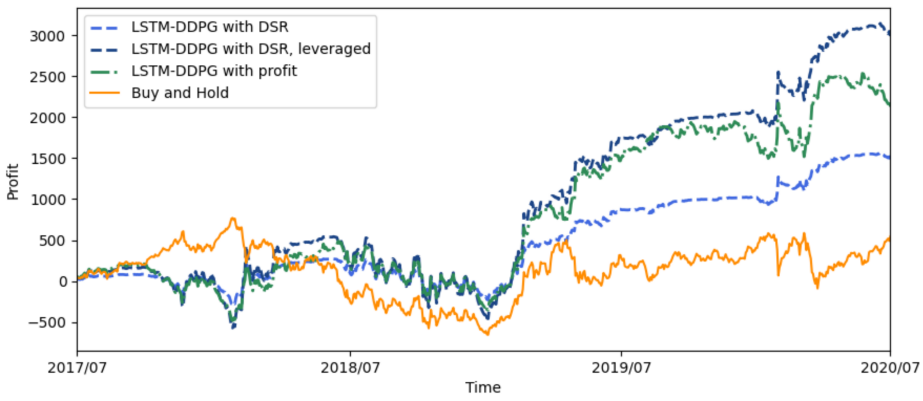

Figure 3. The profit curves of LSTM-DDPG with different reward functions.

As shown in Figure 3, the profit curves of LSTM-DDPG for both DSR and profit as reward functions are significantly higher than that of Buy and Hold most of the time, which means that the LSTM-DDPG is effective and can make a profit sustainably. Note that the price moves in a normal way from July 2017 to June 2020 while there is a "high peak" from July 2002 to June 2014. The LSTM-DDPG performs well in the test set although the market behavior in the test set is quite different from that in the training set since the agent in the reinforcement learning system can refine its responses and predictions and adapt to new environments by exploration. For the LSTM-DDPG, the profit curve of profit reward function is higher than that of DSR reward function in general, however the latter is much smoother than the former. It seems to be that the profit reward function can achieve better

profit performance while the DSR reward function can achieve better risk performance in the proposed LSTM-DDPG model.

Table 2 shows the performance results of buy and hold, and the LSTM-DDPG with different reward functions quantificationally. For the LSTM-DDPG, the profit reward function has the higher total return rate (42.5%) and larger maximum drawdown (15.3%) than the DSR reward function without leverage (29.8% and 9.5%). The Sharpe ratio is an indicator that considers both profit and trading risk. It can be seen from Table 2 that the DSR reward function has a higher Sharpe ratio (0.328) than the profit reward function (0.248). This means that the DSR reward function achieves better overall performance considering both profit and trading risk compared with the profit reward function.

Table 2. Performance of buy and hold, LSTM-DDPG with different reward functions.

Method	Reward Function		Total Return Rate (TR)	Sharpe Ratio (SR)	Maximum Drawdown (MDD)
Buy and Hold	\		10.9%	0.040	24.7%
LSTM-DDPG	profit		42.5%	0.248	15.3%
	DSR	without leverage	29.8%	0.328	9.5%
		with leverage	59.8%	0.328	18.1%

Leverage is an investment mechanism using borrowed money to increase buying power in a margin account. The result is to multiply the potential returns and the potential downside risk will be multiplied at the same time. In China's futures market you can borrow up to 90% of the purchase price of a security. You don't have to margin all the way up to 90%. You can borrow less, say 50%. If you use $5000 cash in your margin account to purchase $10,000 worth of securities you would have a 2× leverage. IF300 is a financial instrument trading on leverage. To fairly compare the profitability of two reward functions in LSTM-DDPG, the profit curve of the DSR reward function is adjusted, keeping the volatility of profits the same as the profit reward function through leverage, which is shown in Figure 3.

It can be seen that the leveraged profit curve of the DSR reward function is significantly higher than that of the profit reward function most of the time, which means that in the LSTM-DDPG the DSR reward function can obtain higher profits than the profit reward function under the same trading risk. As shown in Table 2, the total return rate of the DSR reward function with leverage is 59.8%, which is higher than that of the profit reward function (29.8%), as expected. The DSR reward function can tap the investment potential of LSTM-DDPG with variable positions and can achieve well-balanced trading performance between profit and risk factors.

3.2.2. Comparisons among Fixed, Few-Levels and Variable Position Sizes

When the position size changes arbitrarily with market condition in the process of trading, the efficiency of investment capital can be improved and the trading risk can be reduced. Table 3 shows the performance of buy and hold and the LSTM-DDPG with fixed, few-levels, and variable position sizes for the test period, in which DSR is taken as the reward function. When the variable position mechanism is employed in the LSTM-DDPG, a variable position size in the range from 0 to 1 will be determined according to market conditions. The few-levels position mechanism says that the position size can only be chosen from several values and here three values: 0, 0.5 and 1, are set. In the fixed position mechanism, no position or maximum position is used during trading.

Table 3. Performance of buy and hold, LSTM-DDPG with fixed, few-levels and variable position sizes.

Method	Position Size		Total Return Rate (TR)	Sharpe Ratio (SR)	Maximum Drawdown (MDD)
Buy and Hold	\		10.9%	0.040	24.7%
LSTM-DDPG	fixed		28.0%	0.215	12.9%
	few-levels		33.2%	0.265	12.4%
	variable	without leverage	29.8%	0.328	9.5%
		with leverage	46.3%	0.328	14.3%

It can be seen from Table 3 that in the LSTM-DDPG model the variable position mechanism has a higher Sharpe ratio (0.328) than the few-levels position mechanism (0.265), and that the few-levels position mechanism has a higher Sharpe ratio than the fixed position mechanism (0.215). The LSTM-DDPG, with variable position sizes, achieves the smallest maximum drawdown (9.5%) when there is no leverage in the trading process. We can also see from Table 3 that the LSTM-DDPG with variable position sizes can obtain the highest total return rate (46.3%) when leverage is used. These demonstrate that the variable position mechanism can achieve better overall performance considering both profit and trading risk compared with the fixed and few-levels position mechanisms.

Figure 4 illustrates the profit curves and position curves of the LSTM-DDPG with fixed and variable position sizes, in which the DSR is taken as a reward function. We can see from the purple shadowed areas in the figure that the LSTM-DDPG with variable position sizes prefers a much larger position size in order to make as much money as possible when the market (as represented by the profit curve of buy and hold) presents a large upward or downward trend and the market direction is easily judged. We can also see from the yellow shadowed areas that the LSTM-DDPG with variable position sizes adjusts its position size to near zero to avoid trading risk when the market is volatile and it is difficult to determine the market direction or when there is no definite direction in the market. In this way the variable position mechanism in the LSTM-DDPG can adjust the position size according to the market conditions and thus can achieve high profit with low risk in the process of trading. Therefore, a good performance in term of the Sharpe ratio is expected for the LSTM-DDPG with variable position sizes.

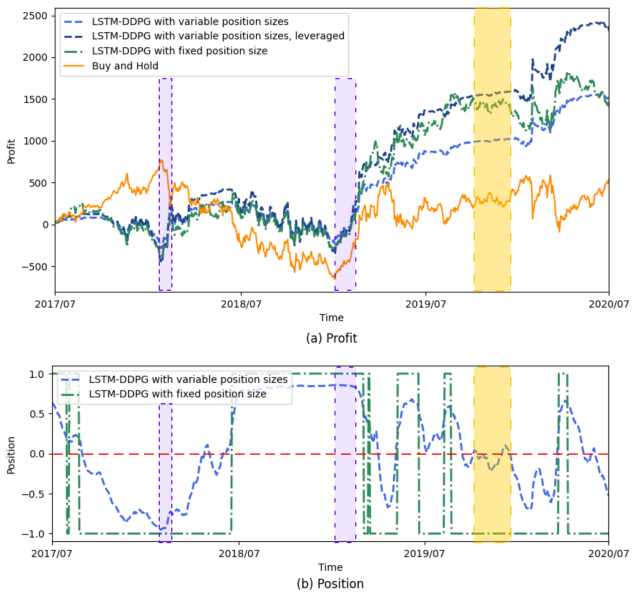

Figure 4. The profit curves and the position curves of the LSTM-DDPG with fixed and variable position sizes.

4. Conclusions

In this paper, we have considered the trading process as a POMDP and proposed the LSTM-DDPG model to make trading decisions regarding the direction and variable size of position. The different reward functions of DST and profit have been considered as reward functions in the LSTM-DDPG. Our model has been trained and tested on China IF300 data. The experimental results show that the LSTM-DDPG model can achieve good trading performance with well-balanced profit and risk. The variable positions mechanism in LSTM-DDPG can adjust the position size according to the market conditions to try to increase the trading profitability and avoid trading risk. The LSTM-DDPG with variable positions and the DST reward function can achieve a higher total return rate for the testing period than other trading methods when leverage is used.

There are some investigations that can be pursued in the future. First, the high price, low price, trading volume etc. besides the closing price can be taken as the input of our model. Second, the trading action at the previous time step can affect the transaction fees of the current action. A better trading decision may be made if the previous action is considered. Therefore, the action at the previous step will be taken as the input of the actor network in our model in our future study. Moreover, to further demonstrate the effectiveness of our model, we will extend the experiments to other markets such as stocks, commodity futures, foreign exchange futures, etc.

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

Funding: This research was funded by the National Nature Science Foundation of China under Grant Nos. U2033215 and 91638301, and the Key R&D Program of Zhejiang Province under Grant No. 2020C05005.

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. Saad, E.W.; Prokhorov, D.V.; Wunsch, D.C. Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Trans. Neural Netw.* **1998**, *9*, 1456–1470. [CrossRef] [PubMed]
2. Jiang, Z.; Xu, D.; Liang, J. A deep reinforcement learning framework for the financial portfolio man-agement problem. *arXiv* **2017**, arXiv:1706.10059.
3. Feng, F.; He, X.; Wang, X.; Luo, C.; Liu, Y.; Chua, T.-S. Temporal Relational Ranking for Stock Prediction. *ACM Trans. Inf. Syst.* **2019**, *37*, 1–30. [CrossRef]
4. Hu, Y.; Feng, B.; Zhang, X.; Ngai, E.; Liu, M. Stock trading rule discovery with an evolutionary trend following model. *Expert Syst. Appl.* **2015**, *42*, 212–222. [CrossRef]
5. Hirchoua, B.; Ouhbi, B.; Frikh, B. Deep reinforcement learning based trading agents: Risk curiosity driven learning for financial rules-based policy. *Expert Syst. Appl.* **2021**, *170*, 114553. [CrossRef]
6. Li, Q.; Jiang, L.; Li, P.; Chen, H. Tensor-based learning for predicting stock movements. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; Volume 29.
7. Imajo, K.; Minami, K.; Ito, K.; Nakagawa, K. Deep Portfolio Optimization via Distributional Prediction of Residual Factors. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 2–9 February 2021; Volume 35, pp. 213–222.
8. Nakagawa, K.; Abe, M.; Komiyama, J. Ric-nn: A robust transferable deep learning framework for cross-sectional in-vestment strategy. In Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), Sydney, Australia, 6–9 October 2020; pp. 370–379.
9. Pun, C.S.; Wang, L.; Wong, H.Y. Financial thought experiment: A GAN-based approach to VAST Robust portfolio se-lection. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, Yokohama, Japan, 7–15 January 2021.

10. Tsang, K.H.; Wong, H.Y. Deep-Learning Solution to Portfolio Selection with Serially Dependent Returns. *SIAM J. Financ. Math.* **2020**, *11*, 593–619. [CrossRef]
11. 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.* **2017**, *28*, 653–664. [CrossRef] [PubMed]
12. Carapuço, J.; Neves, R.; Horta, N. Reinforcement learning applied to Forex trading. *Appl. Soft Comput.* **2018**, *73*, 783–794. [CrossRef]
13. Huang, C.-Y. Financial Trading as a Game: A Deep Reinforcement Learning Approach. *arXiv* **2018**, arXiv:1807.02787.
14. Lei, K.; Zhang, B.; Li, Y.; Yang, M.; Shen, Y. Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. *Expert Syst. Appl.* **2020**, *140*, 112872. [CrossRef]
15. Moody, J.; Wu, L.; Liao, Y.; Saffell, M. Performance functions and reinforcement learning for trading systems and portfolios. *J. Forecast.* **1998**, *17*, 441–470. [CrossRef]
16. Sutton, R.S. *Introduction to Reinforcement Learning*; MIT Press: Cambridge, MA, USA, 1998.
17. Puterman, M.L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*; John Wiley & Sons: New York, NY, USA, 1994.
18. Spaan, M.T. Partially observable Markov decision processes. In *Reinforcement Learning*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 387–414.
19. Sharpe, W.F. Mutual fund performance. *J. Bus.* **1966**, *39*, 119–138. [CrossRef]
20. Kim, H.Y.; Won, C.H. Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Syst. Appl.* **2018**, *103*, 25–37. [CrossRef]
21. 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.
22. Luo, S.; Lin, X.; Zheng, Z. A novel CNN-DDPG based AI-trader: Performance and roles in business operations. *Transp. Res. Part E Logist. Transp. Rev.* **2019**, *131*, 68–79. [CrossRef]
23. Neuneier, R. Optimal asset allocation using adaptive dynamic programming. *Adv. Neural Inf. Process. Syst.* **1996**, *8*, 952–958.
24. Liu, Y.; Liu, Q.; Zhao, H.; Pan, Z.; Liu, C. Adaptive Quantitative Trading: An Imitative Deep Reinforcement Learning Approach. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 2128–2135. [CrossRef]
25. Li, Y.; Zheng, W.; Zheng, Z. Deep Robust Reinforcement Learning for Practical Algorithmic Trading. *IEEE Access* **2019**, *7*, 108014–108022. [CrossRef]
26. Jeong, G.; Kim, H.Y. Improving financial trading decisions using deep Q-learning: Predicting the number of shares, action strategies, and transfer learning. *Expert Syst. Appl.* **2019**, *117*, 125–138. [CrossRef]