CORE
Metadata, citation and similar papers at core.ac.uk
Provided by Teeside University's Research Repository

Noname manuscript No.
(will be inserted by the editor)

Application of Deep Reinforcement Learning in Stock Trading Strategies and Stock Forecasting

Yuming Li · Pin Ni · Victor Chang

Received: date / Accepted: date

Abstract The role of the stock market across the overall financial market is indispensable. The way to acquire practical trading signals in the transaction process to maximize the benefits is a problem that has been studied for a long time. This paper put forward a theory of Deep Reinforcement Learning in the stock trading decisions and stock price prediction, the reliability and availability of the model are proved by experimental data, and the model is compared with the traditional model to prove its advantages. From the point of view of stock market forecasting and intelligent decision-making mechanism, **this paper proves the feasibility of Deep Reinforcement Learning in financial markets and the credibility and advantages of strategic decision-making.**

Keywords Reinforcement Learning · Financial Strategy · Deep Q Learning

1 Introduction

1.1 Background

As the current Artificial Intelligence methods have become closer to the way humans think and behave, there is a need to develop something innovative. Deep Reinforcement Learning (DRL), which integrates the perception of Deep Learning with the decision-making ability of Reinforcement Learning, simulates human cognition and learning mode. This method can input vision and other multidimensional and high-dimensional resource information, and then

Yuming Li, Pin Ni
Department of Computer Science, University of Liverpool, Liverpool, UK
E-mail: Y.Li278@liverpool.ac.uk E-mail: P.Ni2@liverpool.ac.uk
Victor Chang
School of Computing Engineering and Digital Technologies, Teesside University, Middlesbrough, UK
E-mail: victorchang.research@gmail.com

directly output actions through the simulation of Deep Neural Network, which can be controlled directly according to the input image without external supervision.

Deep Neural Network (DNN) can automatically find the corresponding representation of the lower dimension by extracting the higher dimension input data. The core of DNN is to integrate the bias of respondent into the hierarchical neural network architecture. Therefore, Deep Learning has a strong perception and feature extraction ability. Its weakness is the lack of possessing decision-making capabilities. Although Reinforcement Learning can be used in decision-making processes, it has problems to express perception fully. This has motivated us to integrate Deep Learning with Reinforcement Learning since each method will be complementary to each other. The integrated approach can provide a scheme for the construction of cognitive decision-making system of the sophisticated system.

The stock market is characterized by rapid change, many interference factors and insufficient periodic data. Stock trading is a game process under incomplete information, and the single-objective supervised learning model is difficult to deal with such serialization decision problems. Reinforcement learning is one of the effective ways to solve such problems. The conventional quantitative investing is often based on technical index, with the relatively poor self-adaptability and short life span. This paper tend to realize the application about introducing Deep Reinforcement Learning model to financial area, which can deal with the huge scale data in financial market, enhance the ability of data processing and extracting features from transaction signals, to improve the ability of transaction. Besides, this study combines Deep Learning with Reinforcement Learning theory in computer science area to the field of financial field and realize the feature of Neural Network to catch and analyze the information of the mass of data from the field of finance. For instance, stock exchanging is a sequential decision-making approach, for Reinforcement Learning, the final task is learning multiple stage behavior strategies. The method can identify the best price in a certain state, to make the transaction cost the lowest. Consequently, it has the best practicability for the investment field.

1.2 Organization

The subsequent sections are arranged as follows: Section 2 introduces and analyzes the methods recorded in the existing literature, and puts forward the gap of the existing research; Section 3 describes the architecture of the proposed method; Section 4 describes the details of the experiment, including the experimental environment, parameter settings and method description. Section 5 illustrates the results of the experiment in the form of charts and tables, and gives a brief description of the results. Section 6 analyzes the above experimental results in detail, and explains the meaning of the results and the reasons leading to the results; Section 7 compares the proposed method with

another alternative approach; Section 8 summarizes the full text and proposes further research plans.

2 Literature Review

The major point of view about the early Deep Reinforcement Learning is through involving neural networks for dimensionality decrease of data from the higher dimension to promote data processing task. Shibata et al. [28][27] irstly integrated the monolayer neural network with Reinforcement Learning to process the visual signal during the construction of the model for the pushing-box task automatically, Lange et al. [18] proposed the application of competent deep auto encoder to visual learning control, and came up with "visual motion learning" to train the agent have human-like perception and decision-making capacity. Abtahi et al. [2] proposed the Deep Belief Networks (DBN) into Reinforcement Learning, in the process of model construction, the DBN is used to replace the original value function approximator, and the model is triumphantly applied to the character segmentation task of license plate image. Then, Lange et al. [19] proposed Deep Q-Learning witch applied Reinforced Learning based on visual issue to automatically control the car. Koutnik et al. [16] made the combination of the Neural Evolution (NE) method and Reinforcement Learning to the popular car race game TORCS [35] and finally realize the automaticly driving of the automobile.

In the stock decision model based on DRL, the DL part automatically perceives the current market environment for feature learning, and the RL part construct the interaction together with deep characterization and makes trading decisions to accumulate the final return of the current updated environment.

Mnih et al. [23] was called the pioneer of DRL. In this paper, the pixel points of the game screen are taken as the input data (S), and the front, rear, left and right directions of the game joystick are taken as actions (A) to solve the decision-making problem of atari games. Finally, he proved that the performance of agent of Deep q-network could surpass all existing algorithms in 2015 [24]. Many researchers then improved DQN later on. Van Hasselt et al. [30] proposed Double-DQN, in which one of the Q networks chooses the action and the other Q network evaluates the action. The two networks work together to solve the deviation problem existing in a single DQN. In 2016, Silver et al. [25] added a replay mechanism based on original Double-DQN to speed up the training process and added camouflage samples. Wang et al. [34] brought forward the Dueling Network, which is a DQN-based method divides the original network into an output scalar V(s) and an output action to the dominant value, and integrates two Q values after operation respectively. Silver et al. [29] demonstrated Deterministic Policy Gradient Algorithms (DPG), then DDPG paper by Google [21] combined DQN and DPG together to put DRL into continuous motion space control. To the research from Berkeley University [26], the essential of the method is the credibility of simulating and improving the

stability of the DRL model. Gabriel et al. [10] creatively introduced the concept of action embedding, embedding the discrete action in reality into the continuous space, so that the reinforcement learning method can be applied to large-scale learning problems. The above results can prove that in order to adapt to more realistic situations, the deep reinforcement learning algorithm is continuously improved and perfected. Reinforcement learning can observe the environment without supervision, actively explore and trial and error, and can self-summarize excellent experience. Although the active learning system combining deep learning and reinforcement learning is still in the initial stage, it has achieved excellent results in learning various video games.

In recent years, researchers have become increasingly interested in evolutionary algorithms like genetic algorithm [8] [9] [5], and artificial neural networks [6], to come up with stock trading strategy. Deep reinforcement learning has been applied to such areas as high frequency trading and investment portfolios in financial pair trading. To be more exactely, the Reinforcement Learning (RL) algorithm have been used in quantitative finance [4]. The superiority in applying RL concepts in finance is common, which includes the automated processing real-time data with high-frequency, and conduct transactions efficiently with the use of agent. For example, both Sarsa (On-Policy TD Control) and Q-learning (Off-Policy Temporal Difference Control Algorithm) are used by the optimization algorithm of optimized by JP Morgan Chase trading system [15]. League Champion Algorithm (LCA) [3] for the extraction stock trading rules, the process extracts and holds multiple stock trading rules for diversiform stock market environment.

Krollner et al. [17] review diverse types of stock market forecasting papers based on machine learning, such as neural network based models, evolution and optimization mechanics, multiple and compound methods, ect. The Artificial Neural Network (ANN) is used commonly by scientists to predict the stock market trend [33][20]. For example, Guresen et al.[13] use Dynamic Artificial Neural Network (DANN) and Multi-Layer Perceptron (MLP) Model for NASDAQ Stock Index prediction. Hu et al.[1] combined reinforcement learning algorithm and cointegration paired trading strategy to solve the problem of portfolio selection.Using sotino ratio as the return index, the adaptive dynamic adjustment of model parameters is realized, and the return rate and sotino ratio are greatly improved.The maximum retracement obviously drops, the transaction frequency obviously reduces.However, there are fewer bonds, smaller data sets and fewer status indicators. Vanstone et al. [31] design a MLP-based trading system to detect trading signals for the Australian stock market. Due to the limitations of a single model, hybrid machine learning (HML) models have been used to resolve financial trading points based on time sequence. HML models have become the mainstream for financial analysis as follows. J.Wang et al. [32] propose a hybrid Support Vector Regression model. It can connect Principal Component Analysis with Brainstorm optimization to predict trading prices. Mabu et al.[22] introdused a rule-based evolutionary algorithm combined with MLP to identify the trading points of the stock market.

Due to the mutual influence of a large number of complex factors, financial market data have the characteristics of uncertainty and timing. Data analysis is a complex nonlinear and unsteady problem. Traditional statistical model and mass data mining model are not effective in financial forecasting and sequence decision making. Traditional quantitative investment algorithms are often relyed on technical indicators and evaluation criteria.These strategies usually have a long life span and poor self-adaptation. Subsequent machine learning algorithms can significantly improve strategic data in financial investment.The processing speed of the machine can significantly improve the adaptability of strategies and the ability to extract market characteristics from real-time trading signals.

3 ARCHITECTURE OF OUR DEPLOYMENT

This section describes the architecture of our deployment, since it can help us to get analysis quicker, better and with higher accuracy. As shown in Figure 1. When the raw data comes in, the data analysis phase is performed first to ensure that it is not extreme data. At the same time, the original data is passed into the data processing phase. Data will then be the input for the Deep Q Network part. DQN is a kind of network that uses a neural network to predict Q value and continuously updates the neural network to learn the max Q value. There are two neural networks (NN) in DQN: one is Target-Network with relatively fixed parameters, which is used to obtain the target value; the other is called Current Q-Network, which is used to evaluate the Current Q value. The training data is extracted randomly from Replay Memory, which records the actions (a), rewards (r), and results of the next state $(s, a, r, s^{'})$. As the Environment changes, Networks will update its parameters regularly and Replay Memory will change accordingly. The Loss function is the result of subtracting the value of Q in Target-Network from the value of Q in Current q-network. The values between modules are changed iteratively until the optimal value of Q is achieved and output operation is carried out.

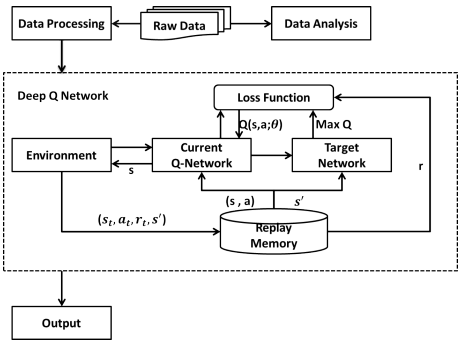

Fig. 1 The architecture of our deployment for experiments and data analysis

4 DESCRIPTION OF THE EXPERIMENT

To verify the feasibility of Deep Reinforcement Learning in stock market investment decisions, ten stocks in the 'Historical daily prices and volumes of all US stocks' dataset were selected by randomization for experiments. The dataset is available from kaggle, which provides the full historical daily price and volume data for all US-based stocks and ETFs (Exchange-Trade Funds) trading on the NYSE (New York Stock Exchange), NASDAQ (National Association of Securities Dealers Automated Quotation), and NYSE MKT (New York Stock Exchange Market). The three classic models in DRL, Deep Q-Network (DQN), Double Deep Q-Network (DDQN), and Dueling Double Deep Q-Network (Dueling DDQN) were selected for computational analysis and then performance comparison. Each stock was split into a training set and testing set and then fed into three Deep Reinforcement Learning models. The effects of the models were visually examined by simulating trading, and the benefits of the three models were compared horizontally. The training results and test results could then be compared and analyzed simultaneously.

4.1 Experimental Environment

The experimental environment and its requirement were dependent on python 3.5 version environment based on Tensorflow deep-learning-framework, with Windows 10 64-bit system.

4.2 Methods

We illustrate the three traditional Deep Reinforcement Learning algorithms as follows:

4.2.1 Deep Q Network

Deep Q Network is one of the most classical and excellent algorithms in DRL. Before DQN appear in the deep reinforcement learning research found that using the nonlinear mapping deep network to represent the value function is not stable or even no convergence, and deep network training samples requirements are independent of each other but before and after the intensive study of a correlation between state data, these problems make directly from the high-dimensional data learning control strategy can be difficult. However, deep Q network introducing experience replay technology, target network and other methods, to overcome the above problems, use DQN build the Agent on the Atari 2600 game[28] directly before four frames of the graphics for the input, output control instruction for end-to-end training. In the test, DQN shown that it can be comparable to those of human players, and even outperforming experienced human experts in less difficult, non-strategic games.

The algorithm blends the Q-learning algorithm and neural network benefits. This is achieved by 1) increasing the experience reply function, from the previous state transition (experience) in the random sample training and 2) overcoming the correlated data and non - stationary distribution problems. In DQN, the Q value represents the current learned experience. The key of DQN model is to learn the q-value function, and finally be able to converge and accurately predict the Q value of each action in various states. The Q value calculated according to the formula is a score obtained by the agent through interaction with the environment and its own experience (namely, the target Q value). Finally, update the old Q value $Q^{'}(s_t, a_t)$ with the target Q value $r_{t+1} + \gamma max_{a'} Q(s_t^{'}, a_t^{'}; \theta)$. The corresponding relationship between the target Q value and the old Q value is exactly the corresponding the correlation between the result value and the output value in the supervised learning neural network. The experience pool can save the transfer samples (s_t, a_t, r_t, s_{t+1}) fetched and stored by each time step agent and the current environment into a memory unit, some are randomly fetched for training at the time needed. The loss function of DQN is presented below:

$$L(\theta) = E[(TargetQ - Q(s_t, a_t; \theta))^2]$$

(a_t represents the action situation of the Agent, s_t refer to the current state of this Agent, r_t is a real number that means the reward of selected action, θ indexes the mean square error of the network parameter, and $Q^{'}$, $s_t^{'}$, $a_t^{'}$ signals the renovated value of Q, s_t and a_t.)

Under certain conditions, Q learning algorithm only needs to use greedy strategy to ensure convergence, so Q learning is a more commonly used model independent reinforcement learning algorithm.

4.2.2 Double DQN

In Q learning and deep Q learning, optimal Q value will be used to select and measure an action. Choosing an overestimated value will lead to overestimation of the real value of Q. Van Hasselt et al. [30] found and proved that the traditional DQN method had the problem of overestimating the Q value, and the error would accumulate with the increase of the number of iterations. The proposal of Double Deep Q learning is to solve the problem caused by overestimation.

Deep Q learning can be regarded as a new neural network plus an old neural network. They have the same structure, but their internal parameters are updated with time difference. Since the optimal Q value predicted by the neural network is inherently wrong, and the error will become larger and larger with the iteration, Double DQN introduces another neural network to optimize the influence of error. Target network and main network can effectively reduce the number of participants. The specific operation is to modify the generating of the Target Q value is:

$$TargetDQ = r_{t+1} + \gamma Q(s_t, argmax_{a'} Q(s_t^{'}, a_t^{'}; \theta); \theta^{'})$$

4.2.3 Dueling DQN

In many DRL tasks, the value functions of the state action pairs are different under the influence of different actions. However, in some states, the size of the value function is independent of the action. Based on this situation, Wang et al.[34] proposed Dueling DQN, and add it into the DQN network pattern.

Dueling DQN combines Dueling Network with DQN.Dueling net assigns its eigenvalues extracted from the convolutional network layer to its two branches. The first part is the state value function $V(s_t; \theta, \beta)$, which stands for the value of the current state environment itself; the second part is the action advantage function $A(s_t, a_t; \theta, \alpha)$ of the dependent state, which refer to the extra value of an Action (A). Finally, the final Q value $Q(s_t, a_t; \theta, \alpha, \beta)$ can be obtained by re-aggregating the two paths, $V(s_t; \theta, \beta)$ and $A(s_t, a_t; \theta, \alpha)$ together. In the above function, a_t stands for an action of the Agent, θ is the convolutional layer parameter, s_t represents a state of the Agent, and α and β are the two-way fully connected layer parameters.

In real cases, the action dominant flow is commonly set as the individual action advantage function $Q(s_t, a_t; \theta, \alpha, \beta)$ minus the average of all action advantage functions $\frac{1}{|A|} \sum_{a'} A(s_t, a_t^{'}; \theta, \alpha)$ in a certain state.

The advantage of this method is that, when there is no sample to a, a can also be updated, data can be used more efficiently and training can be accelerated. In this way, the relative order of the main functions of each action in this state can be guaranteed to remain unchanged, and the range of Q value can be reduced to reduce the redundancy, so as to improve the overall stability of the algorithm.

4.3 Experimental Steps

The experimental steps can be presented in the following sequences:

- The experimental data were imported and preprocessed. Invalid data were cleaned first, and then the data were divided into training set and test set according to the ratio of 4:6, and appropriate experimental parameters were set.
- Ten stocks were randomly selected, three reinforcement learning algorithms were used to simulate trading, and their closing prices were obtained and compared.
- Further analysis was made on the nature of the single stock and the above experimental results were combined
- The present results are analyzed and discussed

4.4 Problems

Due to the large quantity of data in the dataset, there are certain differences in the size of individual data. For example, some stocks have been recorded for decades, while some newly listed stocks are only a few months. The 10 stocks tested are not consistent in time dimension. In the whole process of the experiment, even though time is not taken as the input parameter that influences the experimental results, some external influences will still be generated in the actual market.

5 RESULTS

The profit of training set and test set about the ten stocks in the three Deep Reinforcement Learning models are shown in Table 1.

Table 1 Profit Statement Table

	DQN		DDQN		Dueling DDQN	
Name	Train-Profit	Test-Profit	Train-Profit	Test-Profit	Train-Profit	Test-Profit
nbh.us	48	56	4	9	16	24
intx.us	88	184	39	149	91	334
rlje.us	37	38	138	147	47	349
eght.us	57	198	18	15	13	38
ibkco.us	20	9	11	12	10	5
rbcaa.us	418	519	312	425	353	418
int.us	958	1029	100	166	10	22
pool.us	179	336	42	66	7	27
a.us	490	916	296	551	173	285
kdmn.us	-9	-7	1	15	1	8

We went on to pick one of the 10 randomly selected stocks for experiments. Figure 2 shows an overview of the time-market value of the three models. The grey line indicates that 'stay' is selected. The cyan line indicates that the 'buy' operation is selected. While the purple line indicates that the sell operation is selected at this time."stay" means the decision-maker adopts a stay-and-wait attitude towards the current situation, neither buying nor selling; "buy" means the trading method selected by investors with a bullish attitude towards the future stock trend; and "sell" means the trading method selected by investors with a bearish attitude towards the future stock trend. Through the matplotlib package in Python, the decision-making process can be visualized, making the distribution of three kinds of decisions clearer and easier to analyze.

Figure 3 shows the Reward and Loss functions of the selected stock after the training by three models. Loss function is used to estimate the degree of inconsistency between the predicted value of the model and the real value,

Fig. 2 time-market value profile

which is mapped by an event (An element in a sample dimension). The experiment compares the loss functions of the three DRL models in order to measure and compare the economic benefits of these models. Reward defines the goal in the reinforcement learning problem, Reward function refers to the total reward caused by the change of environmental state influenced by the sequence of actions selected at each time point. It is an instant reward that can measure the pros and cons of the actions.

Figure 4 shows the close price forecasting line chart of the above randomly selected ten stocks, where the green line indicates the original close price, the red line indicates the test close price, and the blue line indicates the predicted close price. In every sub figure, x-coordinate represent the time, while the y-coordinate represents the corresponding closing price.

Next, extract a single stock for comparison between methods, randomly select the stock named "a.us", first evaluate the following indicators of the stock: MACD, KAMA, Aroon Oscillator, AccelerationBands, stochastic oscillator, Chaikin Money Flow, PSAR, ROC, Momentum and VWAP. As shown in Figure 5.

MACD (Moving Average Convergence and Divergence) is characterized by the dispersion and aggregation of slow and fast moving averages to characterize the current long-short status and the potential trend of growth of stock prices.

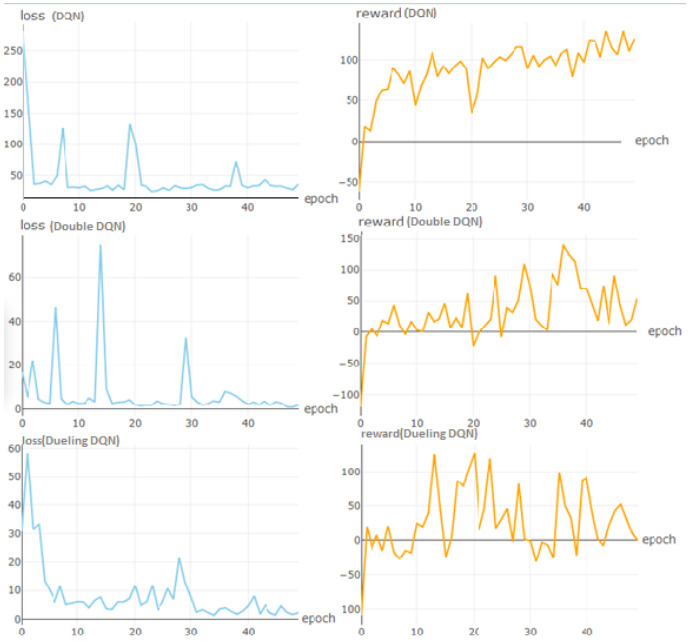

Fig. 3 The Loss Function and the Reward Function of the stock

KAMA (Kaufman Adaptive Moving Average) uses an efficiency ratio to adjust the moving average to accommodate trend and ranging price trends, and also allows the user to control the up and down smoothing limits.

Aron Oscillator can indicate the beginning of a new trend and measure the strength of a trend. The indicator consists of three lines: Aroon-Up, Aroon-Down, and the Aron Oscillator that reacts to the difference.

Acceleration Bands is a statistical indicator. Calculated using the highest and lowest prices. Returns two time series, consisting of the fluctuation moving average of 20 time periods multiplied by the median price plus or minus twice the price.

The Stochastic Oscillator, also known as the KD indicator, consists of a %K line and a %D line. The "%K" line represents the difference between the latest price and the recent lowest price, compared to the recent highest and lowest price difference. The "%D" line algorithm is the same, but the "recent" time range is three times that of the former.

Chaikin Money Flow, referred to as CMF, CMF is based on the assumption that a strong market (in an uptrend market) is usually accompanied by a closing price in the upper half of the daily high and low prices and an enlarged volume. In contrast, a weak market (a market that is in a downtrend) is usually accompanied by a closing price in the lower half of the daily high and low prices and an enlarged volume.

The PSAR indicator is a technical indicator designed to monitor market kinetic energy.The ROC is the speed of the day when the stock price is com-

Fig. 4 The Prediction Broken Line Graph of the stock

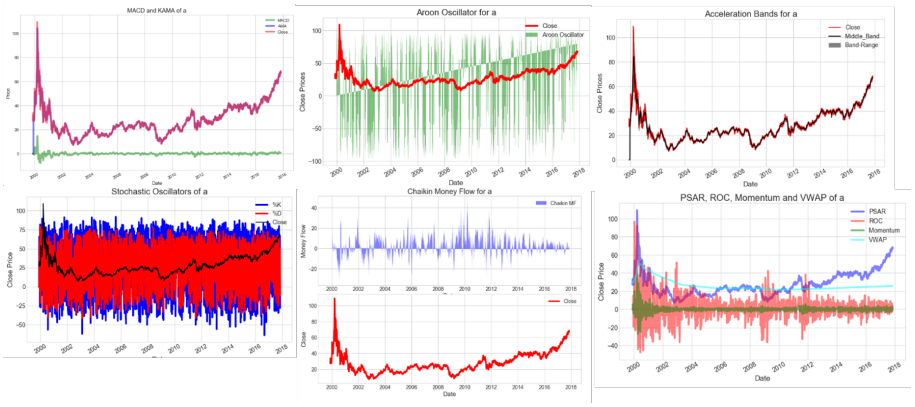

Fig. 5 Technical Indicators Visualization

pared with the stock price of a certain day before a certain number of days, reflecting the degree of change in the stock market.Momentum refers to the ability of stocks (or economic indicators) to continue to grow. VWAP is an average price weighted based on the number of trades traded.

The three strategies of Deep Reinforcement Learning strategy, buy-andhold strategy and KD technical indicator are respectively used in the 10 stocks selected by the empirical data, and the historical strategy net worth of 10 stocks is aggregated and averaged, which is equivalent to the funds for each stock purchased are the same, and each fund is invested in equal capital. The average net value of each strategy is averaged to obtain the total net profit rate of each strategy, and the sent buy signal is imported into the backtesting model. The following backtest results can be obtained in Figure 6, while x-coordinate represents the time, and the y-coordinate represents the corresponding closing price:

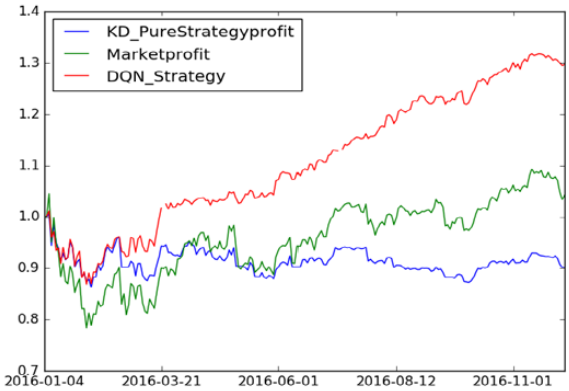

Fig. 6 Back test capital trend chart

6 DISCUSSION

From the perspective of a single stock, as shown in Table 1, the deep reinforcement learning strategy can be fully applied, and some stocks will still show a negative profit, but this method is effective for most stocks and has certain feasibility. For example, the stock kdmn.us has a negative return when using DQN for decision-making, but a positive return through the other two improved reinforcement learning models. Most of the stocks' test profit is higher than the train profit, because the test set uses the optimal Q value of the training set in the training process. In unsupervised mode, the effect of using the optimal Q value is better than that in the process of exploring the optimal Q value. At the same time, by comparing three kinds of depth of reinforcement learning model, it can be seen that DQN in stock decisions are general is greater than the benefits of Double DQN and Dueling DQN. Although Double DQN and Dueling DQN are based on the improved version of DQN, due to the difference between the application field of this paper and that of the above algorithm, double Q network and dueling network are better than traditional

dqn in game competition, but they are better in stock market not applicable for market decision-making.

The shortcomings of traditional human decision-making: 1) insufficient information and it cannot be accurately valued; 2) one-sided basis based on an indicator, since the effect is very poor; 3) the summarized indicators and fixed operational strategies, which cannot dynamically adapt to environmental changes, have weak abilities to counter the effects of risks. As a result, we propose our approaches with distinctive features as follows. First of all, this method has been adopted in the financial field, and the application has achieved good results, which can play a role in assisting manual decision-making, thus saving manpower to some extent. Second, the algorithm can use a large amount of historical data as learning materials. Third, in the handling of emergencies by algorithms, most of the cases are more 'experienced' than human decision-making. It is an improved and hybrid method of using deep learning intelligence to improve stock returns. All these have been evident by the analysis of Figure 2 and 3, the experimental results demonstrate that deep reinforcement learning (represented by three algorithms) plays an effective role in stock timing trading decisions, and can guarantee returns under most conditions.

The experiment provides a paradigm to prove the application of Deep Reinforcement Learning in the area of decision-making mechanism of stock market , which can be used by subsequent practitioners to apply in the real market. In addition, as shown in Table 1, DQN is the Deep Reinforcement Learning model with the best result, not the improved Double DQN and Dueling DQN. This illustrates that we can not stuck in empirical mistakes in practical applications. Improvements based on the original method are not always better than the original. Moreover, each method has its own specific field of application. The method that works well in autopilot may not be applicable to the financial field. Divide and conquer is the best method.Therefore, the conclusion must be made through feasible scientific experiments to prove the point of view.

It can be analyzed from the above experimental results that, except for the possible invalid data, the deep reinforcement learning model is applicable to most stocks with sufficient information, but it cannot be fully applicable to all stocks. Therefore, in the actual investment of practitioners, the dependence on the model should be reduced as far as possible to reduce the risk. The model only plays an auxiliary role in judgment, rather than a theorem. The empirical results prove the differentiated application of deep learning in the financial field to some extent.

As shown in Figure 4, the prediction data is a little bit of delay to test data. This shows that the decision-making result of Deep Reinforcement Learning is to imitate and learn the past data trend. The decision-making choice for a distinct node is based on the data before the node, but not the future data, which is slightly insufficient at the global level.

Figure 5 shows that the stock we randomly selected fluctuated greatly in the initial stage, and then gradually became stable, without obvious specificity, and could be used as a universal experimental sample.

As shown in Figure 6, it can be seen from the net trend graphs of the three strategies that **the Deep Reinforcement Learning strategy is superior to the buy-hold strategy and the KD technical indicator strategy.** And as the number of learning samples and the length of learning increase, its advantages become more and more obvious.

With the continuous upgrading of hardware platforms, the computational resources and computing power have been greatly improved, so that the algorithm that requires a lot of training time can be reduced to a shorter time period. The contribution of the Deep Reinforcement Learning algorithm is selfevident, but the algorithm cannot ignore the powerful computing resources needed behind it. In order to improve the training efficiency of the algorithm faster, we can't rely on the support of hardware resources, and we need to carry out more in-depth research on the efficiency of data utilization training. Nowadays, most Deep Reinforcement Learning algorithms It is based on the research done on the premise of a single agent behavior control task, and the decision-making tasks (such as real-time strategy games, multiplayer online confrontation games, multi-agent information interaction, etc.) are completed in collaboration with multi-agents that require different attributes. The performance is still unsatisfactory, and the current work has been carried out[11][12], and has aroused widespread concern from all walks of life. It can be expected that Deep Reinforcement Learning algorithms based on multi-agent collaboration will become one of the focuses of future research.

7 COMPARISON WITH AN ALTERNATIVE APPROACH

Chang et al. [7] proposed an alternative approach was based on the development of the Adaboost algorithm[14]. The purpose was to enable computational financial modeling to be conducted and completed with both accuracy and performance achieved. The algorithm was a different sub-division of AI. The approach was to study the historical data and fully understand the trends of the data movements. Once the trends have been captured, the algorithms can better predict the movement. As a result, predictive modeling can be achieved. The predicted stock index can be adjusted when there are changes in the market, so that the accuracy for predictions can be higher. Similarly, optimization algorithms have been developed to ensure better performances can be achieved. Our approach in this paper is to use three DRL algorithms, which can calculate the best strategies and the ideal prices every second. When there are changes, DRL algorithms adapt changes in the next available time unit and then optimize the performance at the same time. In other words, accuracy and performance can be maintained, but more likely to get better outcomes in the next time unit, rather than the same time unit. Both approaches for com-

Strategies	DRL algorithms	Adaboost algorithms
Simulations	All three algorithms can be simulated at the same time	Despite concurrent simulations are possible, "one-at-a-time" approach works better
Accuracy	DRL algorithms can adjust the input based on market changes, and better accuracy can be achieved in the next time unit.	Adaboost algorithms capture data movement and predict based on the trend. A better accuracy can be achieved in the same time unit. It can also be adapted for getting a better accuracy in the next time phase.
Performance	Excellent performance. Optimization can improve performance significantly.	Excellent performance. Optimization can improve performance significantly.
Optimization	A better optimization tends to happen in the next time phase. It is possible to adapt to concurrent optimization with an advanced HPC approach.	Optimization can happen at the same time, but a different algorithm is required to run together with Adaboost algorithms.
Popularity	Some quantitative developers and trading firms have used and improved DRL algorithms.	Some quantitative developers and trading firms have used and improved DRL algorithms.

Table 2 A detailed comparison between DRL and Adaboost algorithms

putational analysis are popular. However, a detailed comparison is presented in Table 2 as follows.

It is also possible to blend both approaches as a hybrid solution. For example, if we are familiar with certain stocks and have a better understanding on them, we can use Adaboost algorithms, since we can capture the more reliable trend of data movements. This can save more time and resources to acquire accurate results computationally, which tend to consume much more energy and require more high-end computing resources. If we are less familiar with new stocks, we can use improved DRL algorithms. This can allow us to get a better understanding of our new invested stocks, and we can compare between predicted and actual values. Once a high accuracy is achieved, we can decide to retain using DRL algorithms or switch to Adaboost algorithms to capture its trends of data movement.

8 CONCLUSION and FUTURE WORK

The paper implemented a novel Deep Reinforcement Learning for stock transaction strategy, and proved the practicality of DRL in dealing with financial strategy issues, and made the comparison of three classical DRL models. The outcomes demonstrated these three learning algorithms we developed were effective, particularly the DQN model with the best performance in dealing with decision-making problems of stock market strategies. The advantages of using our proposed algorithms are as follows. First, these three algorithms have better intelligence than traditional transactions, as they could respond

to the market quickly and adapted for changes. Second, other single-objective supervised learning model is difficult to deal with such serialization decision problems. The reinforcement learning algorithm is a deep learning algorithm that is most similar to the learning process of human beings. The human exploration and development process is similar to the continuous trial and error of reinforcement learning, obtaining the environmental reward label and the alternating process of learning with empirical data. As a result, our three algorithms can be more suitable for tasks involved with humans such as trading and other human-based operations. However, the weakness is that there was a problem with the data set itself. The data size difference was large, which could bring instability to the experiment to a certain extent. This was the most complete data set in the field commonly used for financial analysis.

This research proved the feasibility of the Deep Reinforcement Learning algorithm in the financial field and its practicability as an auxiliary tool for financial investment decision-making. Reinforcement learning was a recently focused algorithm that was more innovative in business applications. Our contribution was mainly on making the application of deep reinforcement learning (a new reinforcement learning variant) to financial transactions, as an innovative approach to the application level of this model. The decision results of our three algorithms under the same set of data were compared. Finally, it was concluded that DQN maximization of decision benefits among the three models, which could be used as a reference for future research. We also compared between improved DRL and Adaboost algorithms in detail in terms of simulations, accuracy, performance, optimization and popularity. Each approach had its specific ways of focuses and apparent strengths and suitability for different cases. We proposed a hybrid solution. For stocks that investors were familiar with, they could use Adaboost. For new invested stocks, they could use DRL algorithms, and then switched to Adaboost algorithms to capture trends of data movement, and would decide the next investment strategy. In summary of our contribution, we developed novel three algorithms that could fit the research direction of interdisciplinary research. We could broaden the scope of use of tools for quantitative investment, and also extended the scope of application of deep learning applications.

The theory of Deep Reinforcement Learning is now widely accepted. However, there are still many challenges to be overcome, such as the exploration and utilization of balance problem, slow convergence rate, space disaster and so on. The following research will gradually increase the latest technology of deep reinforcement learning, continuously enhance the ability of model learning strategy, search for the method of high-level abstract logic memory and control intelligent agent. We plan to improve the main shortcomings in the following research, especially in the field of financial application, and try to introduce the data characteristics and strategy templates of the field, and propose a finance-DRL algorithm applicable to the characteristics of the financial field.

References

1.
2. Abtahi, F., Zhu, Z., Burry, A.M.: A deep reinforcement learning approach to character segmentation of license plate images. In: Machine Vision Applications (MVA), 2015 14th IAPR International Conference on, pp. 539–542. IEEE (2015)
3. Alimoradi, M.R., Kashan, A.H.: A league championship algorithm equipped with network structure and backward q-learning for extracting stock trading rules. Applied Soft Computing **68**, 478–493 (2018)
4. Almahdi, S., Yang, S.Y.: An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. Expert Systems with Applications **87**, 267–279 (2017)
5. Berutich, J.M., López, F., Luna, F., Quintana, D.: Robust technical trading strategies using gp for algorithmic portfolio selection. Expert Systems with Applications **46**, 307–315 (2016)
6. Chang, P.C., Liao, T.W., Lin, J.J., Fan, C.Y.: A dynamic threshold decision system for stock trading signal detection. Applied Soft Computing **11**(5), 3998–4010 (2011)
7. Chang, V., Li, T., Zeng, Z.: Towards an improved adaboost algorithmic method for computational financial analysis. Journal of Parallel and Distributed Computing **134**, 219–232 (2019)
8. Cheng, C.H., Chen, T.L., Wei, L.Y.: A hybrid model based on rough sets theory and genetic algorithms for stock price forecasting. Information Sciences **180**(9), 1610–1629 (2010)
9. Chien, Y.W.C., Chen, Y.L.: Mining associative classification rules with stock trading data–a ga-based method. Knowledge-Based Systems **23**(6), 605–614 (2010)
10. Dulac-Arnold, G., Evans, R., van Hasselt, H., Sunehag, P., Lillicrap, T., Hunt, J., Mann, T., Weber, T., Degris, T., Coppin, B.: Deep reinforcement learning in large discrete action spaces. arXiv preprint arXiv:1512.07679 (2015)
11. Foerster, J., Assael, I.A., de Freitas, N., Whiteson, S.: Learning to communicate with deep multi-agent reinforcement learning. In: Advances in Neural Information Processing Systems, pp. 2137–2145 (2016)
12. Foerster, J.N., Farquhar, G., Afouras, T., Nardelli, N., Whiteson, S.: Counterfactual multi-agent policy gradients. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
13. Guresen, E., Kayakutlu, G., Daim, T.U.: Using artificial neural network models in stock market index prediction. Expert Systems with Applications **38**(8), 10389–10397 (2011)
14. Hastie, T., Rosset, S., Zhu, J., Zou, H.: Multi-class adaboost. Statistics and its Interface **2**(3), 349–360 (2009)
15. jpmorgan. https://www.businessinsider.com/jpmorgan-takes-ai-use-to-the-next-level-2017-8
16. Koutník, J., Schmidhuber, J., Gomez, F.: Online evolution of deep convolutional network for vision-based reinforcement learning. In: International Conference on Simulation of Adaptive Behavior, pp. 260–269. Springer (2014)
17. Krollner, B., Vanstone, B., Finnie, G.: Financial time series forecasting with machine learning techniques: A survey (2010)
18. Lange, S., Riedmiller, M.: Deep auto-encoder neural networks in reinforcement learning. In: The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2010)
19. Lange, S., Riedmiller, M., Voigtlander, A.: Autonomous reinforcement learning on raw visual input data in a real world application. In: Neural Networks (IJCNN), The 2012 International Joint Conference on, pp. 1–8. IEEE (2012)
20. Liao, Z., Wang, J.: Forecasting model of global stock index by stochastic time effective neural network. Expert Systems with Applications **37**(1), 834–841 (2010)
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 preprint arXiv:1509.02971 (2015)
22. Mabu, S., Obayashi, M., Kuremoto, T.: Ensemble learning of rule-based evolutionary algorithm using multi-layer perceptron for supporting decisions in stock trading problems. Applied soft computing **36**, 357–367 (2015)

23. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
24. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529 (2015)
25. Schaul, T., Quan, J., Antonoglou, I., Silver, D.: Prioritized experience replay. arXiv preprint arXiv:1511.05952 (2015)
26. Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P.: Trust region policy optimization. In: International Conference on Machine Learning, pp. 1889–1897 (2015)
27. Shibata, K., Iida, M.: Acquisition of box pushing by direct-vision-based reinforcement learning. In: SICE 2003 Annual Conference, vol. 3, pp. 2322–2327. IEEE (2003)
28. Shibata, K., Okabe, Y.: Reinforcement learning when visual sensory signals are directly given as inputs. In: Neural Networks, 1997., International Conference on, vol. 3, pp. 1716–1720. IEEE (1997)
29. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic policy gradient algorithms. In: ICML (2014)
30. Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI, vol. 2, p. 5. Phoenix, AZ (2016)
31. Vanstone, B., Finnie, G., Hahn, T.: Creating trading systems with fundamental variables and neural networks: The aby case study. Mathematics and computers in simulation **86**, 78–91 (2012)
32. Wang, J., Hou, R., Wang, C., Shen, L.: Improved v-support vector regression model based on variable selection and brain storm optimization for stock price forecasting. Applied Soft Computing **49**, 164–178 (2016)
33. Wang, J.Z., Wang, J.J., Zhang, Z.G., Guo, S.P.: Forecasting stock indices with back propagation neural network. Expert Systems with Applications **38**(11), 14346–14355 (2011)
34. Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., De Freitas, N.: Dueling network architectures for deep reinforcement learning. arXiv preprint arXiv:1511.06581 (2015)
35. Wymann, B., Espié, E., Guionneau, C., Dimitrakakis, C., Coulom, R., Sumner, A.: Torcs, the open racing car simulator. Software available at http://torcs. sourceforge. net **4**, 6 (2000)