Stock Trading Agent with Reinforcement Learning Algorithms ELENE6885 Course Project

Chengyue Li [*1] **Qiaolin Wang** [*1]

Abstract

With the continuous development of the times and the continuous growth of the stock market, more and more enterprises and individual investors are joining the stock investment and trying to obtain considerable returns. However, designing profit strategies in complex and dynamic stock markets is challenging. This project involves developing a trading agent using dual reinforcement learning algorithms. The agent learns from stock market data and recommends actions such as "buy," "sell," or "hold" to maximize cumulative returns. Three cutting-edge framework algorithms have been applied in reinforcement learning environments: Deep Q-Network (DQN), Dominator Critique (A2C), and Deep Deterministic Policy Gradient (DDPG). The model can be trained using specified stock historical data and provide guidance for future investments and decisions. From the experimental test results, all three algorithms have achieved good cumulative returns, and the trading agent we trained has practical application value.

1. Introduction

Investment is the process of obtaining returns by allocating resources to the most efficient places, and portfolio optimization is a decision-making process that maximizes investment utility. Portfolio optimization can be seen as an investment strategy aimed at continuously reallocating funds to different financial assets, assigning different weights to each asset, in order to achieve the goal of maximizing returns or minimizing risks. Since Markowitz proposed the mean variance model in 1952, portfolio optimization theory has undergone a long period of development. The objects of portfolio optimization may include financial assets such as stocks, bonds, foreign exchange, and cryptocurrencies. Among these assets, the stock market has always been a very important investment direction, and investors used to trade stocks based on their views on the stock market. However, this trading method is usually inefficient and can easily cause significant losses due to investors' irrational behavior. With the rapid development of computer hardware and algorithms, traditional stock trading is undergoing significant changes, laying the foundation for the development of quantitative trading. Unlike traditional trading practices, quantitative trading utilizes historical data and mathematical models to search for profitable trading patterns through computer calculations. In both the financial and academic fields, the main challenges faced by research on the stock market are its high complexity and dynamic properties. Stock market data is complex, noisy, nonlinear, and non-stationary, and previous trading strategies often cannot achieve profitable returns under real market conditions. In the related research of quantitative trading, many scholars use deep learning or meta-heuristic methods to predict stock prices. After obtaining a well performing model, they trade the stocks they hold based on the predicted direction. However, as mentioned earlier, real stock market data is complex and unstable, and making decisions through establishing prediction models for the market often fails in practice. To address this issue, scholars have proposed trading strategies based on Deep Reinforcement Learning (DRL) algorithms, hoping that trading agents developed based on DRL can make decisions based on historical data, statistical indicators, or heuristic methods, and take appropriate actions to achieve significant levels of return.

In based stock trading strategies based DRL, there are mainly two categories: high-frequency trading and low-frequency trading. The action frequency of high-frequency trading systems is minutes and seconds, and research in this field is usually conducted by cutting-edge quantitative fund companies. These studies are not publicly published, and there is limited literature available. In addition, high-frequency trading systems have very strict hardware requirements, as they not only require high-performance GPUs, but also require very small latency devices to connect to

[*]Equal contribution [1]Department of Electrical Engineering, Columbia University, New York, USA. Correspondence to: Chengyue Li <cl4578@columbia.edu>, Qiaolin Wang <qw2443@columbia.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

data centers containing real-time stock prices, which is one of the reasons why high-frequency trading research is so scarce. Therefore, the main focus of this study is on low-frequency trading, where agents use the frequency of actions as the minimum unit per day, and can buy and sell stocks on a daily, weekly, or monthly basis like real investors. The use of DRL for stock portfolio optimization is a popular and innovative research direction. Therefore, the significance of this study is to construct a deep reinforcement learning model that conforms to the stock market, which expands the application field of DRL at the theoretical level and provides investors with a novel method for investment decision-making in practical terms.

2. Background

Stock investment, as the most common and popular investment method at present, has attracted more and more people to participate. Research on stock price trends and stock buying and selling strategies has begun to emerge. These studies aim to analyze market information and stock prices in order to identify potential patterns. This article will introduce the current research status in the field of stock investment from three aspects: financial investment based on traditional methods, financial investment based on deep learning, and financial investment based on deep reinforcement learning.

People often make appropriate trading decisions by predicting stock prices. In traditional stock prediction research, most methods use regression models and machine learning models. For example, (Sharma et al, 2017) investigated widely used effective regression methods and applied stock market data to predict stock market prices. (Yang et al, 2022) selected the methods of least squares ridge regression, Bayesian ridge regression, Lasso method, and polynomial regression in multiple linear regression to fit and predict stock data through the discovery of stock linkage and analysis of prediction methods. (Ingle et al, 2016) used various word frequency inverse file frequency (TF-IDF) features based on data collected from different news channels to predict the price of stocks the next day. They calculated word scores based on the obtained TF-IDF weights and finally generated a hidden Markov model to calculate the probability of sequences and switching values.

With the development and maturity of deep learning (LeCun Y, 2015), some researchers are considering applying it to the field of stock investment. Specifically, (Hiransha et al, 2018) used four different types of deep learning structures, namely Multilayer Perceptron (MLP), Recurrent Neural Networks (RNN), Long Short Term Memory (LSTM), and Convolutional Neural Network (CNN), to predict the historical prices of stock companies. (Hossain et al, 2018) proposed a novel stock price prediction model based on deep learning, which combines LSTM and Gated Recurrent Unit (GRU) to solve the regression based problem of predicting stock prices. (Liu et al, 2020) proposed using a deep learning network predictor with LSTM network as the main part of the mixed frame, and jointly optimizing the stock prediction framework with dropout strategy and particle swarm optimization algorithm. The prediction ability of this framework is higher than that of traditional models.

The reinforcement learning method aims to maximize investment returns, allowing agents to interact with financial markets, generate investment decisions, and obtain returns. In the process of interacting with the environment, agents continuously adjust their decisions with the goal of obtaining maximum returns. With the development of reinforcement learning, some researchers are considering applying reinforcement learning algorithms to stock trading. For example, (Lee et al, 2020) proposed a method of applying reinforcement learning to stock price prediction problems, which is suitable for modeling and learning various interactions in real situations. Adopting a reinforcement learning algorithm that only learns from experience, and approximating functions through artificial neural networks to learn the values of each state, each state corresponds to the stock price trend at a given time. (Nevmyvaka et al,2006) provided a large-scale empirical analysis of reinforcement learning methods applied to optimize execution problems. By using an improved Q-learning algorithm to select the price level for limit trading, there is a significant improvement in trading costs compared to simpler optimization forms. (Ning et al, 2021) adopted a model free approach and developed a variant of Deep Q-Learning (DQN) to estimate the optimal behavior of traders. This model is trained using experience replay and Double Deep Q-Network (DDQN). The input features are represented by the current state of the limit order book, other trading signals, and available execution actions, while the output is a Q-value function that estimates future returns under any action. By applying the model to nine different stocks, it was found that it performed better than the benchmark method on most stocks. (Carta et al, 2021) proposed a multi-level and multi set stock trader to make stock trading decisions. This method first processes the time data of stocks, converts it into meta features, then uses a meta learner to generate trading decisions, and finally uses multiple trading agents to vote and make the final decision.

3. Algorithm Development

In the experiment, we need to obtain daily stock trading price data from the market and use the trading price data to calculate relevant financial indicators. In stock trading tasks, cumulative return and other indicator data are used together to construct the environmental state. Input the environmental state into the reinforcement learning algorithm to obtain task related actions, and calculate the stock returns

Algorithm 1 DQN

Initialize the network $Q_\omega(s,a)$ and target network $Q_{\omega'}$ with random network parameters ω;
Initialize the experience replay pool R;
for $e = 1$ **to** T **do**
  Obtain the initial state of the environment s_1;
  for $t = 1$ **to** T **do**
    Select action a_t based on the current network $Q_\omega(s,a)$ using ϵ-greedy strategy
    Execute a_t and obtain reward r_t, $s_t \rightarrow s_{t+1}$
    Save (s_t, a_t, r_t, s_{t+1}) into the replay pool R
    for each data,use target network compute:

$$y_i = r_i + \gamma \max_a Q_{\omega'}(s_{i+1}, a)$$

    Minimize target loss to update the current network Q_ω

$$L = \frac{1}{N}\sum_i (y_i - Q_\omega(s_i, a_i))$$

    update the target network
  end for
end for

Algorithm 2 A2C

Initialize actor network π_θ and critic network Q_ω, as well as their learning rates a_θ, a_ω;
for $e = 1$ **to** M **do**
  for $t = 1$ **to** T **do**
    Select and execute action a_t at state s_t based on the current policy π_θ, and obtain reward r_{t+1}, s_{t+1}
    Select action a_{t+1} at state s_{t+1} based on the current policy π_θ
    Update actor parameters using policy gradients:

$$\theta = \theta + a_\theta Q_\omega(s_t, a_t) \bigtriangledown_\theta log(\pi_\theta(a_t|s_t))$$

    calculate TD error,

$$\delta_t = r_t + \gamma Q_\omega(s_{t+1}, a_{t+1}) - Q_\omega(s_t, a_t)$$

    By minimizing the TD error to update the critic parameter:

$$\omega = \omega + \alpha_\omega \delta_t \bigtriangledown_\omega Q_\omega(s_t, a_t)$$

  end for
end for

that can be obtained based on the actions. Repeat this process to maximize investment return. The investment process of this article is shown in Figure 1. This article uses three of the most commonly used reinforcement learning algorithms, namely DQN, A2C, and DDPG. DQN belongs to the value function based deep reinforcement learning algorithm, while A2C and DDPG belong to the policy gradient based deep reinforcement learning algorithm. The following will briefly introduce these three algorithms:

3.1. Deep Q-network(DQN)

The DQN algorithm is the most representative value function algorithm, consisting of a Q-network and a target Q-network. The DQN algorithm indirectly learns the best action through the state action value function, which maximizes the expected reward return in the future when the current state has been determined. The algorithm includes an experience pool, where the data obtained from the interaction between the agent and the environment can be placed. After a period of time, a portion of the data is randomly extracted and the current Q network is used to calculate the current Q value, while the target Q network calculates the target Q value. The algorithm steps are shown in the algorithm1.

3.2. Actor-Critic(A2C)

The A2C algorithm combines both strategy and value function, and updates the value function by introducing an evaluation mechanism to solve the problem of high variance. The task of an Actor is to generate corresponding actions based on the current state. The task of Critic is to score the performance of the Actor and guide their actions based on the score. Actors or strategies are used to select actions; The direction of strategy updates guided by the Critic. It should be noted that the actor-Critic algorithm is different from the policy gradient algorithm. It not only needs to update the policy, but also needs to update the value function. The specific steps of the algorithm are shown in Algorithm 2

The algorithm consists of two networks: the actor network and the critic network (used to evaluate actions). The actor still uses the policy gradient update method, while the critic uses the time difference method (TD) to update. Its update target values are the current reward and the Q-value estimation of the next time step, with an error of MSE, which is consistent with the original DQN.

3.3. Deep Deterministic Policy Gradient(DDPG)

The action taken by the DDPG algorithm in a certain state is unique, that is, it takes the action with the highest probability. This algorithm has four networks: Actor Network, Critic Network, Target Actor Network, and Target Critic Network.

The Actor network obtains actions, rewards, and the next state based on the current input state. The Critic network calculates the current Q value and updates the network parameter $Q^{'}$. The target Critic network calculates $Q^{'}(S^{'}, A^{'}, W^{'})$ in the target Q value and periodically copies the $Q^{'}$ parame-

ter as the current parameter.The target Q-value formula is as follows:

$$y_i = R + \gamma Q^{'}(S^{'}, A^{'}, W^{'})$$

Among them, R is the reward and γ is the decay factor.

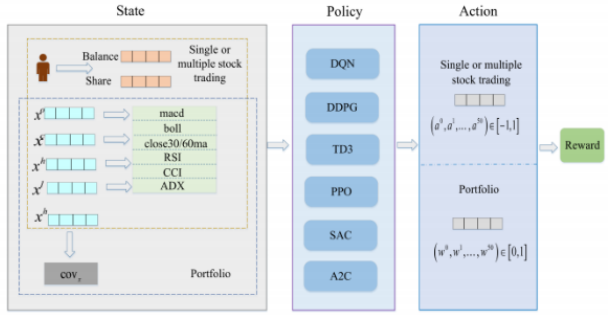

Figure 1. Stock Investment Diagram Based on Reinforcement Learning

3.4. Evaluating indicator

This article uses total reward to evaluate, each action will result in a portfolio return calculation, using the accumulated total return to evaluate the returns brought by the proxy model to the current stage of investment.

$$R_c = v - pre_v$$

v is the portfolio value, pre_v is the previous portfolio value.

4. Experiment results

In this section, we will introduce our proposed scheme for experimentation and performance evaluation. We tested and compared three reinforcement learning strategies. This experiment is based on the reinforcement learning toolkit GYM to develop a proxy learning environment, defining each trading day in the dataset as a time step. At the beginning of training, the agent calls the reset function to obtain observations from the environment and make action decisions. Afterwards, the environment will receive action parameters and call the step function to transfer to the next trading day, returning the reward, current state, and termination signal of the agent's previous state action. Repeat this cycle until the last day of the trading day, at which point the agent will receive a termination signal and stop the cycle. After receiving the termination signal, the agent will reset the environment and start collecting data for training again from the first trading day of the dataset. The reinforcement learning environment used in this study has a fixed number of steps, and the length of an event is equal to the number of trading days in the dataset, with the initial state always being the first trading day of the dataset. The figure2 is a schematic diagram of the structure of the reinforcement learning environment used in this study:

Table 1. Total rewards of three algorithms.

ALGORITHM	TOTAL REWARD
DQN	1671.67
A2C	4251.98
DDPG	1513.82

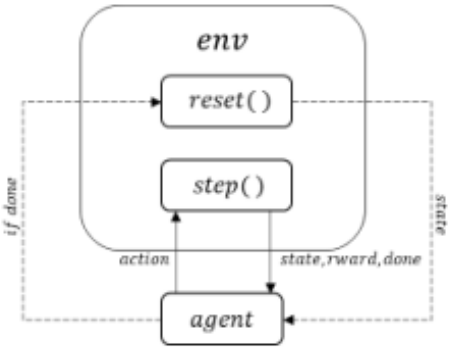

Figure 2. Portfolio Environment Structure

The program we designed allows users to freely input the stock code and date they want to explore, and use historical stock data from a specified time period as training data for proxy training. We chose AAPL stock as the test, using data from 2010 to 2020 as training data, and then testing with data from 2020 to 2021. The degree uses DQN, A2C, and DDPG as proxy strategies. In order to compare, except for different selection strategies, all other parameters are set to 'always', with an iteration count of 500 and an epsilon of 0.1 Finally, we will evaluate the profitability of each algorithm, and Figure3 shows the cumulative returns of the three strategies From the graph, it can be seen that the agents of the three strategies have all achieved certain returns, with the A2C algorithm receiving the highest total return.

The performance comparison of three algorithms in single stock trading tasks is shown in table1. It can be seen that all three proxy strategies have performed well in terms of cumulative returns, among which the A2C strategy has the best effect.

Due to the poor learning ability of value function algorithms in continuous or high-dimensional action spaces. Therefore, three basic reinforcement learning algorithms were used for comparative analysis in stock trading tasks and investment

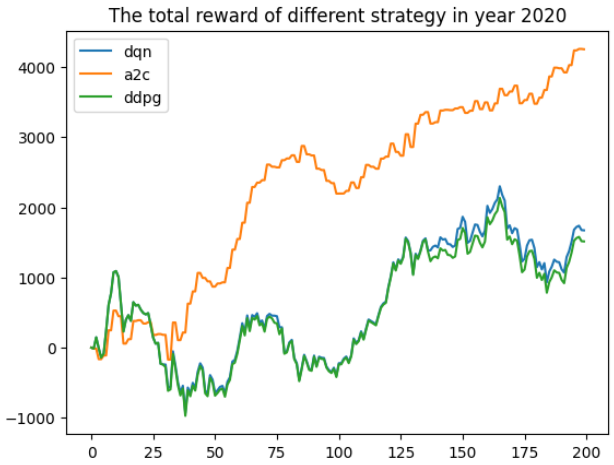

Figure 3. Cumulative return curves of three algorithms

portfolio tasks. It can be seen that A2C is significantly better than DQN, but DDPG seems to have not achieved the desired effect, which requires further analysis.

5. Conclusion

Stock investment portfolio, as a more direct decision-making process, has become an active research topic in recent years. Existing investment portfolio methods ignore the fact that stock assets in the portfolio may be widely interrelated and influenced by each other, while stock relationship information has high value in improving portfolio management. On the other hand, in today's financial markets, using reinforcement algorithms for automatic stock trading is a hot research topic. However, existing reinforcement learning research typically focuses on a single stock trading task or on analyzing and improving the returns of individual reinforcement learning algorithms in stock trading.

This article systematically verifies the effectiveness of various representative deep reinforcement learning algorithms of different types in stock trading and portfolio tasks. Firstly, this article constructs a system that can select historical data of different stocks for training, and uses various reinforcement learning algorithms as trading strategies for comparison. Then you can select stock data from a certain stage for testing. Comparing the performance of different algorithms in cumulative returns, it was found that the A2C algorithm has a good effect in guiding stock trading, but the strategy based DDPG algorithm did not perform well, which may be related to the parameters we set or require more comprehensive testing and comparison.

Based on our existing work, we will further search for reward functions and environmental states that are more suitable for financial investment tasks in the future, so that deep reinforcement learning algorithms can achieve better results. In the following work, we will consider adding more investment products to the portfolio model and verify the effectiveness of reinforcement learning algorithms in various investment products. In addition, we will attempt to optimize our investment portfolio strategy and incorporate more stock relationships into the model, such as the impact of investor sentiment on stock prices and financial news headlines. At the same time, we are considering adding more financial technology indicators to enhance the characteristic information of stocks and optimize investment portfolio strategies.

6. Author Contributions

6.1. Conceptual Work

Chengyue Li: Developed the initial research idea of applying multiple DRL algorithms (DQN, A2C, DDPG) to stock trading, and defined the reward function and state representation.
Qiaolin Wang: Conducted literature reviews to refine the methodology, proposed integrating both discrete and continuous action spaces for different algorithms, and suggested comparing performance across multiple market periods.

6.2. Code Implementation

Chengyue Li: Implemented the DQN and DDPG agents, set up the Q-network/target network structure and continuous action noise. Handled data preprocessing via *yfinance*, normalized input features.
Qiaolin Wang: Implemented the A2C agent with actor-critic architecture, Integrated these agents with the custom *TradingEnv* and performed initial debugging and hyperparameter tuning, and added performance evaluation scripts to compare algorithms.

References

Carta S., Corriga A., F. A. A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. In *Applied Intelligence*, pp. 889–905, 2021.

G.Y., Y. Comparative analysis of linear regression methods for predicting five types of stocks based on stock correlation. In *Modern Business*, pp. 4245, DOI:10.14097/j.cnki.5392/2022.29.037., 2022.

Hiransha M., Gopalakrishnan E. A., M. V. K. Nse stock market prediction using deep-learning models. In *Procedia computer science*, pp. 1351–1362, 2018.

Hossain M. A., Karim R., T. R. Hybrid deep learning model for stock price prediction. In *symposium series on*

computational intelligence (ssci). IEEE,, pp. 1837–1844, 2018.

Ingle V., D. S. Hidden markov model implementation for prediction of stock prices with tf-idf features. In *Proceedings of the International Conference on Advances in Information Communication Technology Computing*, pp. 1–6, 2016.

LeCun Y., Bengio Y., H. G. Deep learning. In *nature*, pp. 436–444, 2015.

Liu H., L. Z. An improved deep learning model for predicting stock market price time series. In *Digital Signal Processing*, pp. 102741, 2020.

Nevmyvaka Y., Feng Y., K. M. Reinforcement learning for optimized trade execution. In *Proceedings of the 23rd international conference on Machine learning.*, pp. 673–680, 2006.

Ning B., Lin F. H. T., J. S. Double deep q-learning for optimal execution. In *Applied Mathematical Finance.*, pp. 361–380, 2021.

Sharma A., Bhuriya D., S. U. Survey of stock market prediction using machine learning approach. In *Proceedings of the 2017 International conference of electronics, communication and aerospace technology (ICECA). IEEE*, pp. 506–509, 2017.

W., L. J. Stock price prediction using reinforcement learning. In *2001 IEEE International Symposium on Industrial Electronics Proceedings*, pp. 690–695, 2001.