Exploration and Agent Improvement under Near-Real Market Environments Simulation Using DRL

Yichen Li
EECS
University of Michigan
Ann Arbor, US
liyichen@umich.edu

Yilin Li
EECS
University of Michigan
Ann Arbor, US
yilinliz@umich.edu

Jiaming Zeng
EECS
University of Michigan
Ann Arbor, US
zjiaming@umich.edu

I. Introduction

Deep reinforcement learning (DRL) has shown great possibilities in simulating real-market environments with well-designed agents competing each other [1], [2]. However, due to the high complexity and dynamic nature of real-world markets, current models trained using historical data might not be able to faithfully simulate real-world trading scenarios. We shall focus on the adversarial behavior among market investors to better reflect the situation that the users may encounter in the real-world market and better help them understand or simulate an optimal option they may take for different situations.

In this paper, we will do an investigation into some of the real-world common scenarios and extreme situations that may occur in the market. Then we will perform simplified simulations for these situations to see how different agents would develop their behaviors and strategies. After that, we will improve some of the logic or models that agents may use to achieve a better result under specific circumstances. Finally, we will give our conclusion regarding which agent is a better choice under specific circumstances.

II. Datasets and Data Preprocessing

A. Datasets

There are many different stock markets in the world. Among these, the 30 stocks included in the Dow Jones Industrial Index (DJIA) are very representative and have a large amount of data. So, in this paper, we choose a dataset of DJIA accessed from Yahoo Finance [7]. We set the time period to be from 2009-04-01 to 2022-11-01. For each stock, we use date, open value, high value, low value, close value, volume, tic, and day as features. The size of the total dataset we use is $100122 * 8$.

B. Data Preprocessing

After we have accessed all data required, we need to deal with missing data in the data cleaning part and add more features in the feature engineering part to convert the data into a model-ready state.

1) Data Cleaning: The cleaning processes of missing data are usually different for various time frequencies. In the low-frequency case, we directly delete the rows with missing values, reflecting suspension in simulated trading environments. While in the high-frequency case, we fill the open, high, low, and close columns with the last valid value of close price and the volume column with 0, which is a standard method in practice.

2) Feature Engineering: In order to better understand the characteristics of the data and facilitate subsequent training, we add additional features including statistical features (technical indicators, fundamental indicators), sentiment features (headlines, lexicon-based news, sentiment), and embedding features (transformer-based embeddings). Currently, we have successfully add several common technical indicators including Moving Average Convergence Divergence (MACD) [11], Relative Strength Index (RSI), Average Directional Index (ADX), and Commodity Channel Index (CCI).

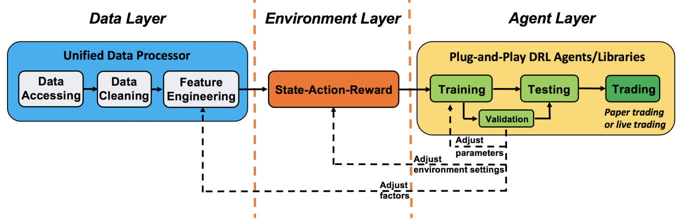

Fig. 1. An image of Financial Real-Market Simulation Pipeline, from [1].

III. Methodology

To acquire an optimal option in some of the extreme situations, we divide our approach into two parts. Firstly, we try to build some of the extreme environments, like high fluctuation, and high transaction fees that probably occur in real market environments. Then, we observe the performance of those popular agents, like A2C, PPO, SAC, and DDPG, and try to find out the logic or their advantages and disadvantages when facing different situations. After that, focus on the pros and cons of different agents (Here, the "different agents" means different Deep Reinforcement Learning models (DRL) they use), we will try to develop or modify some agents to acquire a more robust performance.

A. System Structure

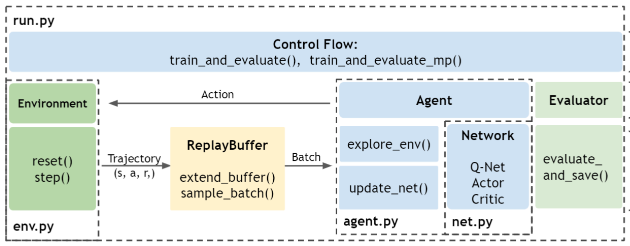

Fig. 2. Structure of market simulation with environments and agents

B. Environment

The environment in the system we are using defines the parameters of a market, it can be influenced by two kinds of factors. The first part of the environment is influenced by the chosen stocks, some indicators like turbulence/fluctuation, the Relative Strength Index (RSI), and Moving Average Convergence Divergence (MACD), etc. Those indicators are either calculated from the prices of the chosen stocks or set to demonstrate information about them. These indicators simulate the "reference" that an investor refers to when making an investment in a real-world market. The second part of the environment is the parameters of the market rules. They can be set by changing them directly, like the number of stocks, transaction cost/fees, state space, and action space. These values simulate the rules set by market managers. So the environment combines the two factors to simulate a more comprehensive market scenario.

C. Agent Building/Modification

We will first try to modify some of the existing agents, to see whether those settings would make a significant difference or improvement. And after that, if time allows, we will also try to implement some other DRL models that exist, but have not yet been implemented here in the ElegentRL repository [4], like Every-visit Monte-Carlo, A3C, which is an A2C model plus an asynchronous part, TRPO, QR-DQN, SARSA, etc.

We hereby introduce a fundamental model A2C and its advancing version A3C we are trying to implement.

1) A2C and A3C: As we can see from Fig., A2C is optimized (Q-actor critic) mainly based on two parts, the advantage and the probability distribution(π)
Using gradient descent, we are able to maximize the advantage. A3C consists of multiple training agents while A2C only utilizes one, but still, A2C is able to achieve relatively accurate results while being more efficient than A3C.

$$\nabla_\theta J(\theta) \sim \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t))$$
$$= \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) A(s_t, a_t)$$

Fig. 3. Structure of market simulation with environments and agents

IV. Evaluation and Current Result

Currently, we have set up an environment based on the DJIA, which consists of 30 popular stocks and their prices from 2009-04-01 to 2022-11-01 (previously discussed in the dataset section). We have also successfully reproduced the results provided by the tutorials on the trading systems. By using the training dataset for building the environment and validation, we successfully trained the A2C model with 40000 total timesteps (a measurement of training steps) of stock prices. Since the A2C model is a relatively naive model compared to A3C, TRPO, and also because of the complexity of the real-world stock markets, the policy loss of the A2C model during the training process has been bumped dramatically and seems not to converge (bounce from the range of -50 to 500). When applying this model to the testing (trading) stage, we are able to receive a total cumulative return of 0.472371 percent. Compared to the baseline (DJIA) cumulative return of 0.388402 percent, it is a relatively good result. (See Fig. 4)

In addition, to further explore the performance of models under different circumstances, we adjust some parameters in the environment settings for both the training and testing processes of the model. One significant result we have achieved so far is the reward scaling term.

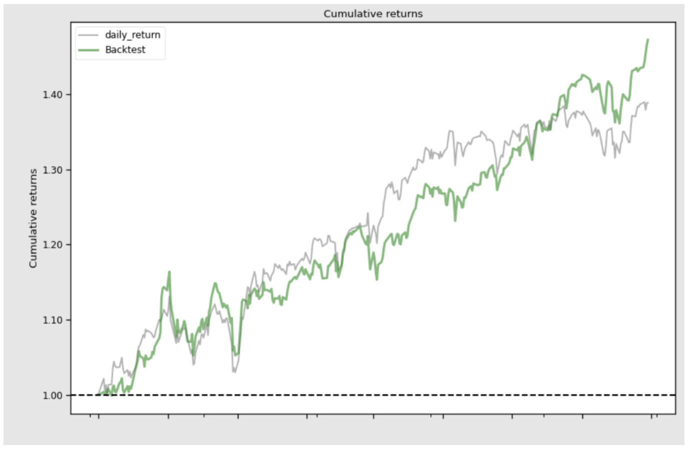

Fig. 4. A2C model results comparing to DJIA, green symbolize A2C

By changing the reward scaling from 1e-4 to 1e-3, we get completely different results. We can check Fig.5 that the cumulative returns of the model are much lower in this setting and are much lower than the baseline returns. It shows how the same models would generate such different results with minor changes of environments.

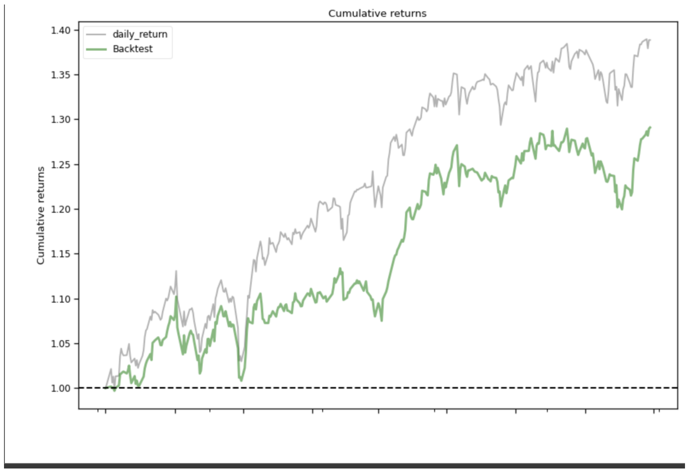

Fig. 5. A2C model results comparing to DJIA, green symbolize A2C

V. Future plan for improvement

After we have successfully implemented the trading system and reproduced our results, we are able to further analyze whether extreme environments such as high volatility would further decrease the behavior of current models. To accomplish this, we first have to implement more complex models such as A3C and DDPG. With these more robust models, we can compare their performance under different environments and conclude whether they are still able to hold. While there are high possibilities for the models to fail if we perturb the environment enough, we still hope that the models would remain robust with limited perturbation. Further, analyze has to show the correctness of this hypothesis.

Based on the conclusions of the previous experiments on different models, we hope to explore further whether and how we should modify the original models to fit extreme environments and achieve better performance. If that process fails again, we also want to explore the possibility of building another model which could still achieve robust results in extreme environments. This also fulfills our final target to find whether we can find the model to beat the baseline in extreme environments.

References

[1] Liu, X. Y., Rui, J., Gao, J., Yang, L., Yang, H., Wang, Z., ... Guo, J. (2021). FinRL-Meta: A Universe of Near-Real Market Environments for Data-Driven Deep Reinforcement Learning in Quantitative Finance. arXiv preprint arXiv:2112.06753.

[2] Zhang, Z., Zohren, S., Roberts, S. (2020). Deep reinforcement learning for trading. The Journal of Financial Data Science, 2(2), 25-40.

[3] Marco Raberto, Silvano Cincotti, Sergio M Focardi, and Michele Marchesi. Agent-based simulation of a financial market. Physica A: Statistical Mechanics and its Applications, 299(1-2):319–327, 2001.

[4] Liu, Xiao-Yang and Li, Zechu and Wang, Zhaoran and Zheng, Jiahao. ElegantRL: Massively Parallel Framework for Cloud-native Deep Reinforcement Learning. Github, 2021.

[5] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

[6] Mnih, V., Kavukcuoglu, K., Silver, D. et al. Human-level control through deep reinforcement learning. Nature 518, 529–533 (2015). https://doi.org/10.1038/nature14236

[7] Yahoo. https://finance.yahoo.com/

[8] Kabbani, T., Duman, E. (2022). Deep reinforcement learning approach for trading automation in the stock market.

[9] W. F. Sharpe, "The sharpe ratio," The Journal of Portfolio Management, vol. 21, no. 1, pp. 49–58, 1994.

[10] Understanding actor critic methods and a2c — by Chris Yoon — towards ... (no date). Available at: https://towardsdatascience.com/understanding-actor-critic-methods-931b97b6df3f (Accessed: November 18, 2022).

[11] Appel, G., Dobson, E. (2007). Understanding MACD (Vol. 34). Traders Press.