CSGA-GA 3033

Intro to DL Systems - Final Project:

Automated Stock Trading using Deep Reinforcement Learning

Prof.Parijat Dube

INTRODUCTION

Stock markets are up and down every day and different stocks could have different trends at the same day. So how to decide to buy, hold or sell which stocks and how to design a profitable strategy in this complex and dynamic stock market? Automated stock trading could be a solution for the following reasons:

- First, automated trading systems *minimize the interference of emotions* throughout the trading process. Since trade orders are executed automatically once the trade rules have been satisfied, traders will not be able to hesitate or question the trade. Also, automated trading can stop those people who are likely to over trade.
- Second, automated trading *allows for backtesting*. Backtesting applies trading rules to historical market data to determine if the idea works. When designing a system for automated trading, traders can make precise sets of rules and test them on historical data before risking their money in a live trading market.
- Also, since computers can *respond immediately* to the changing market conditions, automated systems are able to generate orders as soon as trade criteria are satisfied. Getting in or out of a trade a few seconds earlier can make a big difference in the trade's outcome.

In this project, we aim to create an automated stock trading and portfolio allocation strategy, which maximizes returns by using a Deep Reinforcement learning strategy that leverages historical stock-market price data. We experiment with different stock features (technical indicators) and different deep reinforcement learning algorithms to find the best configuration. Also, we compared different developed strategies and evaluated against the market indices. Our solution demonstrates the effectiveness of an ensemble strategy that our referenced paper did not show. Also, we provide experimental foundations and references for future development directions.

METHODOLOGY

Because the stock market is highly stochastic and dynamic, we employ a Markov Decision Process to model the market. We model the trading of multiple stocks as a continuous action space. Our portfolio consists of 100 stocks from NASDAQ index.

The *state space* of our stock market environment is made up of the available balance ($500k at the beginning) and features of each stock. For each stock, its price, number of shares, different technical indicator features are recorded. We consider following technical indicators:

- Moving Average Convergence Divergence (MACD): MACD is commonly used to identify moving averages.
- Relative Strength Index (RSI): RSI measures how price changes recently. It compares the price with historical strength or weakness to indicate if the stock is oversold or overbought.
- Commodity Channel Index (CCI): CCI represents how below the price is above or below its historical average price.
- Average Directional Index (ADX): ADX indicates the price trend strength by calculating the high, low and close price.
- Average True Range (ATR): ATR is a technical analysis that indicates the degree of price volatility.
- Exponential Moving Average (EMA): EMA tracks the price of an investment over time, and gives more weighting to recent price data.

Regarding the *action space*, for a single stock, the agent is able to choose an action among {-k, …, -1, 0, 1, …, k}, where k is the max number of shares it can buy, and -k is the max number of shares it can sell. k is 100 in our settings. We use an ensemble strategy of different reinforcement learning algorithms:

- Advantage Actor Critic (A2C): A2C is a typical actor-critic algorithm that optimizes the policy gradient updates. There is an advantage function which captures how better an action is compared to the others at a given state.
- Proximal Policy Optimization (PPO): PPO is an on-policy algorithm that can be used for environments with either discrete or continuous action spaces. It controls the policy gradient update and ensures that the new policy will not be different from the previous one.
- Deep Deterministic Policy Gradient (DDPG): is an algorithm which concurrently learns a Q-function and a policy. It uses off-policy data and the Bellman equation to learn the Q-function, and uses the Q-function to learn the policy.
- Twin Delayed DDPG (TD3): A common failure mode for DDPG is that the learned Q-function begins to dramatically overestimate Q-values, which then leads to the policy breaking, because it exploits the errors in the Q-function. Twin Delayed DDPG (TD3) is an algorithm that addresses this issue by introducing three critical tricks.
- Soft Actor-Critic (SAC): SAC is an algorithm that optimizes a stochastic policy in an off-policy way, forming a bridge between stochastic policy optimization and DDPG-style approaches.

As Figure 1 shows, we first train all policies on the training window. Then, we pick the policy that has best performance on the validation data. After that, we retrain the best policy on both training and validation data before testing it. We use Python as the programming language, OpenAI gym to implement the environment and train the agent. We run the code on Google Cloud Platform with Nvidia V100 GPU. The dataset is NASDAQ 100 stocks and their prices collected from Yahoo Finance Data. The training set starts from 2000-01-01 to 2019-01-01 and the test set starts from 2019-01-01 to 2021-01-01.

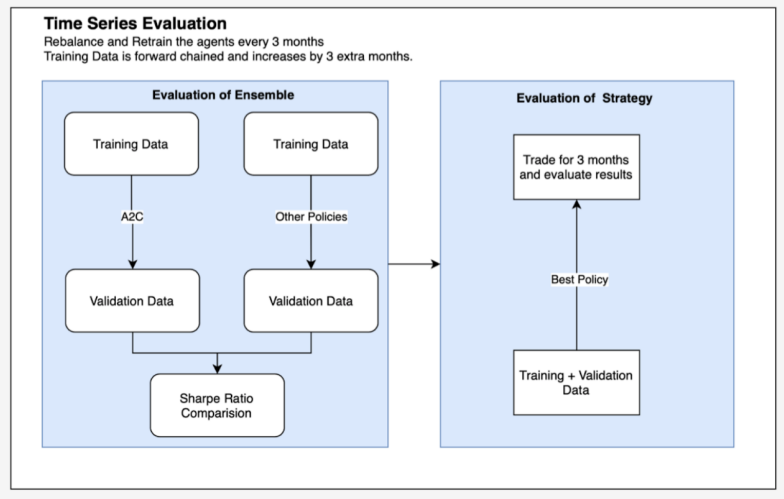

Figure 1. Workflow

EVALUATION

Our main evaluation criteria for our portfolios are Portfolio Returns and Sharpe Ratio:

- *Portfolio Returns* represent the portfolio value after the agent trades in the trading period or the test set.
    - Portfolio Returns $R_{(t)} = \frac{P_t - P_0}{P_0}$
- While an investor would want to optimize their portfolio for maximum returns, they also would want to keep the volatility of their portfolio to its minimum. *Volatility* is represented by the standard deviation of returns and more the volatility, more uncertain the investor is about their returns and the direction.
- *Sharpe Ratio* is defined as the ratio between Portfolio Returns and the Portfolio Volatility. Investors when choosing an investment strategy compares with their Sharpe Ratio to determine the best among them.
    - Sharpe Ratio = $\frac{Portfolio\ Returns}{Volatility}$

We first compared the portfolio performance of different models. As Figure 2 presents, the portfolio is traded by four different models with an initial amount of $500,000. These four models are:

- Yellow line (base model): this model includes A2C, PPO and DDPG policies. This model includes MACD, RSI, CCI, ADX and other basic technical indicators.
- Red line: this model is the same as the baseline model, except that a new technical indicator ATR is added. This configuration is to study the marginal effect of ATR indicator.

- Green line: this model is the same as the baseline model, except it does not include any technical indicators. This model is to study the effectiveness of greatly reduced state space after excluding all technical indicators.
- Blue line: This ensemble model studies the effectiveness of ensembling all five deep reinforcement learning policies: A2C, PPO, DDPG, TD3 and SAC.

We observe that the agent which uses the ensemble of all 5 policies along with baseline technical indicators perform the best.

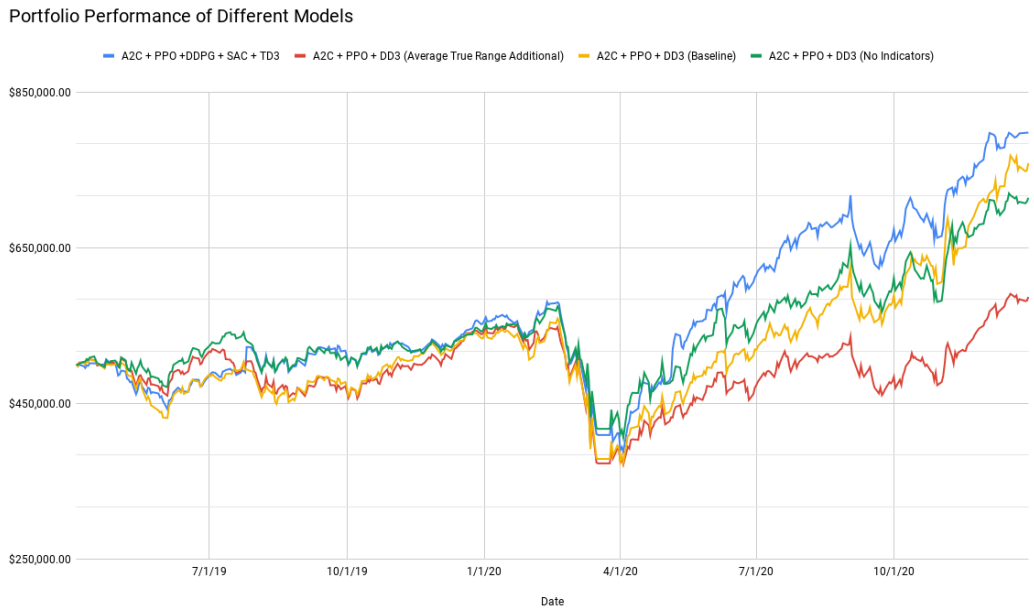

Figure 2. Performance Comparison of Different Reinforcement Learning Agents

In the following Table 2, we compare our best model with NASDAQ 100 index and S&P 500 index. When compared to S&P 500 index, our model shows significantly better performance in annual returns and cumulative returns, and slightly less risk in terms of volatility. When compared to NASDAQ 100 index, our model shows slightly poorer performance in terms of annual return and cumulative returns, but demonstrates safer portfolio level in terms of annual volatility.

	MODEL	NASDAQ 100	S&P 500
Total months	20	20	20
Start Date	2019-04-03	2019-04-03	2019-04-03
End Date	2020-12-31	2020-12-31	2020-12-31
Annual return	30.68%	35.45%	14.28%
Cumulative returns	59.55%	70.07%	26.31%
Annual volatility	26.93%	29.91%	27.77%

Sharpe ratio	1.13	1.19	0.51
Max drawdown	-33.18%	-27.23%	-33.79%

Table 1. Ensemble Reinforcement Learning Agent Backtest Performance

In Figure 3, we represent the portfolio value obtained with our best model that ensembles all five algorithms (A2C, PPO, DDPG, SAC, TD3) in comparison to the NASDAQ 100 when accounted for the portfolio volatility.

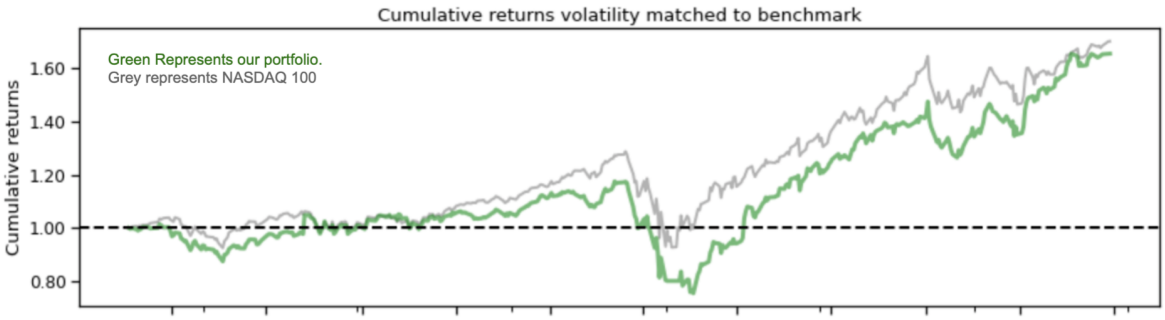

Figure 3. Cumulative Returns Volatility

In Figure 4, we present statistics of monthly returns of our best model. In Figure 5. We represent the rolling volatility and rolling sharpe ratio of our portfolio made with all five algorithms (A2C, PPO, DDPG, SAC, TD3) in comparison to the NASDAQ 100. From Figure 3, Figure 4 and Figure 5, we observe that although our model performs slightly worse than NASDAQ 100 index, our model constantly shows less risk and especially during the huge dump at the beginning of pandemic (March 2020).

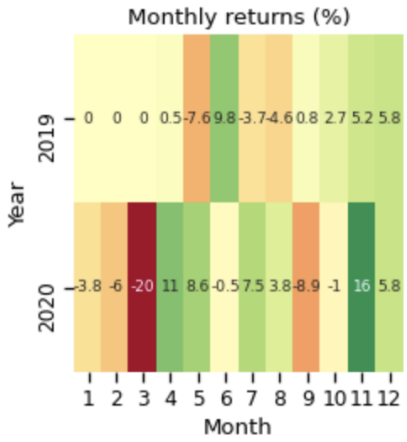

Figure 4. Monthly Returns of Our Best Model

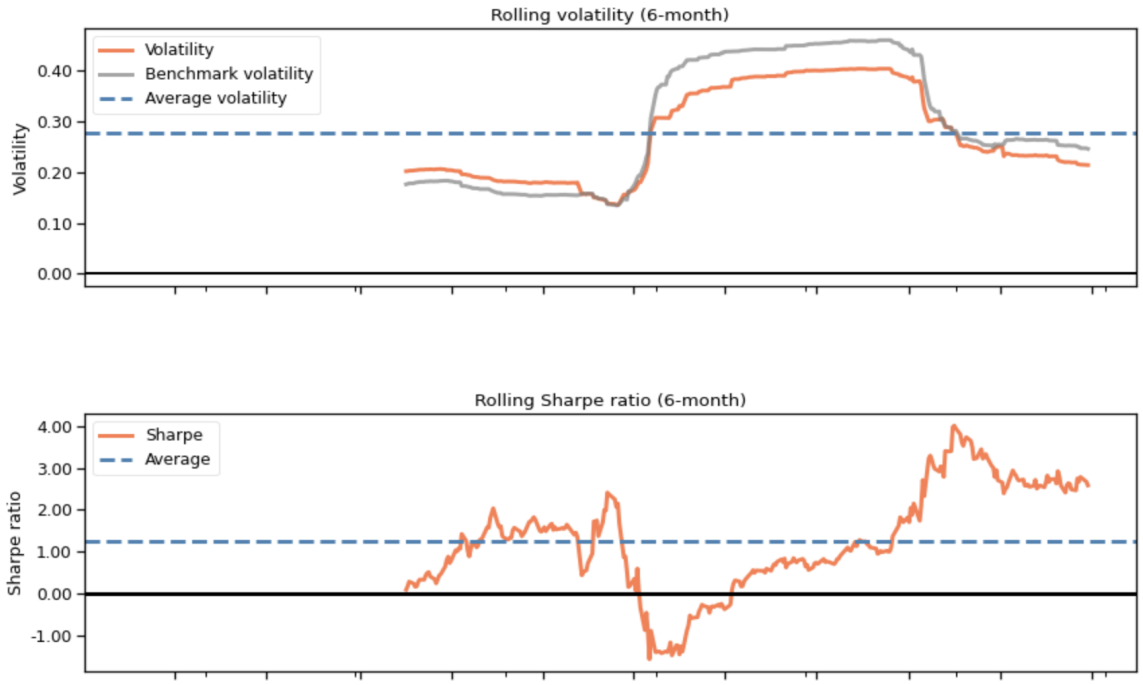

Figure 5. Rolling Sharpe Ratio

CONCLUSION

We develop an automated stock trading and portfolio allocation strategy, which maximizes returns by leveraging a Deep Reinforcement Learning method. We experiment with different stock features and different deep reinforcement learning algorithms. Our ensemble model with all five policies of A2C, PPO, DDPG, SAC and TD3 has the best performance. Our solution demonstrates the effectiveness of an ensemble strategy that our referenced paper did not show. Also, we provide experimental foundations and references for future development directions.

REFERENCE

[1] Yang, Hongyang, et al. "Deep reinforcement learning for automated stock trading: An ensemble strategy." Available at SSRN (2020).
[2] OpenAI Spinning Up (Tutorial for Deep Reinforcement Learning). https://spinningup.openai.com/en/latest/
[3] Liu, Xiao-Yang, et al. "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).
[4] Chen, Lin, and Qiang Gao. "Application of deep reinforcement learning on automated stock trading." 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2019.