Automated Stock Trading using Deep Q-learning

Lakshmisudha kondaka, Lunawat Prajakta,, Mitragotri Atharva, Patel Ijaj
Department of Information Technology
SIES Graduate School of Technology
Nerul, 400706
lakshmi.sudha@siesgst.ac.in, prajakta.lunawat17@siesgst.ac.in ,
atharva.ramesh17@siesgst.ac.in, noormohammad.ijaj17@siesgst.ac.in

Abstract

About two decades ago, stock market trading was limited only to the big industry players. The advent of technology gave birth to High-Frequency Trading (HFT) used by institutions, who used quantitative analysis for trading using their high-end computers. Over next decade, with the inception of online brokerage platforms and discount brokers, stock market trading became common among the masses. Today, many brokers provide an Application User Interface (API) for automating trades based on custom algorithms. With the exponential growth in research on application of artificial intelligence in the field of finance, HFTs have evolved in a plethora of ways - From genetic algorithms to machine learning algorithms, from neural networks to reinforcement learning. The proposed model can outperform the generic strategy by the application of reinforcement learning on past stock data and train the model to take the optimal actions at any point of time, while mitigating the risk generated by the erraticism of the stock market.

1. Introduction

Stock Trading is a financial instrument developed over years to distribute the risk of a venture and to utilize the stagnant wealth. Distribution of securities, grows company capital which in turn creates more jobs, efficient manufacturing, and cheaper goods. Trading of different securities such as bonds and stocks makes the economy more flexible while delivering benefits both for the issuer and the holder. Stock trading has gained popularity as a way of investing stagnant money for better returns, but the complicated environment of trading and the costs of expert traders and managers complicate it for the common man. Often compared with a random walk, the industry has received quite the skepticism and has often been compared with gambling. However, a handful of people have made a successful career out of stock trading and some have established it as a secondary source of income. An automated stock trading system which is able to hedge the risk factor and learn to take optimal actions at any point of time is something that would attract attention from a lot of retail traders.

There are two types of studies related to stocks; Fundamental Analysis and Technical Analysis. Fundamental Analysis deals with the fundamentals of the underlying company such as the balance sheets, pledged shares, profit and loss statement, etc. This type of analysis is more useful for long-term investment. Technical Analysis is the practice of using past stock data to predict future stock movement. Technical Analysis is widely used in short-term stock trading and core trading. The market forms several crests and troughs while moving. These levels are considered important for predicting future market movement. The crests are called resistances and the troughs are called supports. These are useful levels for buying and selling of stocks.

Artificial Intelligence is divided into three branches; Supervised Learning, Unsupervised Learning and Reinforcement Learning (RL). Unlike the other two, Reinforcement Learning is a way to train a model to take a sequence of actions or decisions in order to achieve a defined goal. In RL, we deal with the problem of training an agent in an environment to take the optimal action for a particular state in the environment by rewarding the agent based on a designed policy as shown in figure 1. This process is called the Markov Decision Process (MDP). The goal is to maximize the total reward obtained by the end of the iteration.

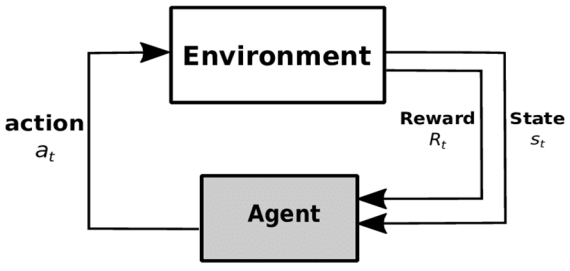

Figure 1 Markov Decision Process.

Q-learning is a form of RL where the rewards are accumulated as Q-values or action values and maintained in a matrix to plan out the optimal action and improve the decision-making process with every iteration. Q-values are defined for state-action pairs i.e. Q (S, A), where S is the current state and A is the action taken. A discount factor is used for calculating the Q-values which is used for future rewards. The Q value can be given as : $Q(S,A) = R(S,A) + \gamma * \max Q(S',A)$, where R(S,A) is the reward obtained in the current state, γ is the discount factor for future rewards, and S' is the next state.

2. Related Work

Abhishek Nan, Siddharth Perumal and Osmar R Zainane proposed a system[1] using reinforcement learning which used traditional time series stock price data and combined it with news headline sentiments, while leveraging knowledge graphs for exploiting news about implicit relationships. This model used reinforcement learning on the stock time series data along with sentiment analysis on the news.

Akhil Raj Azhikodan, Anvitha G. K. Bhat , and Mamatha V. Jadhav proposes automating swing trading [2] using deep reinforcement learning. The deep deterministic policy gradient-based neural network model trains to choose an action to sell, buy, or hold the stocks to maximize the gain in asset value.

Siddharth Das, the proposed system combines the technical data as well as the fundamental data of a stock to outperform the traditional approach of trading [3]. It uses the moving averages, daily volatility and volume ratios of a stock for selection, then makes use of fundamentals like the company's gross profit, liabilities to determine the entry and exit.

In 2018 Huang, Chien proposed a system [4], which uses Markov Decision Process (MDP) model along with Deep Recurrent Q Network (DRQN) algorithm for foreign exchange market. The reinforcement learning method is used in [5] to establish a foreign exchange transaction, avoiding the long-standing problem of unstable trends in deep learning predictions. The research proposes the use of Artificial Neural Network that is feed forward multi-layer perceptron with error back propagation and develops a model of configuration 5:21:21:1 with 80% training data in 130,000 cycles in [6].

Neural networks are on the threshold of becoming assimilated into the mainstream of financial decision making [7]. There is reliable evidence that technical analysis, as used by traders in the foreign exchange (FX) markets, has predictive value regarding future movements of foreign exchange prices [8]. According to the research, a generic algorithm approach for automatically generating expert advisors, computer programs that trade automatically in the financial markets. A Gen Fx was proposed to evaluate evolutionarily generated expert advisors strategies [9].

Reinforcement learning is applied on an optimal dynamic trading strategy by interaction with actual trade market acting as environment [10]. The trading agent is trained using the Q-learning algorithm of Reinforcement learning. This model outperforms the Buy and Hold and Decision-Tree based Trading Strategies.

In another paper [11] Q-learning agent trained several times with the same training data and investigate its ensemble behavior in important real-world stock markets. Experimental results in intraday trading indicate better performance than the conventional Buy-and-Hold strategy, which still behaves well in our setups. The qualitative and quantitative analyses of these results also discussed in this paper.

Reinforcement Learning Applications in Real Time Trading is proposed in [12]. They have demonstrated it is possible to apply reinforcement learning and output valid and simple profitable trading strategy in a daily setting (one trade a day), and show an example of intraday trading with reinforcement learning.

A sentiment analysis model using a recurrent convolutional neural network to predict the stock trend from the financial news is proposed in [13]. The objective of this paper is not to build a better trading bot, but to prove that reinforcement learning is capable of learning the tricks of stock trading.

A new system for short-term speculation in the foreign exchange market, based on recent reinforcement learning (RL) developments was proposed in [14]. Neural networks with three hidden layers of ReLU neurons are trained as RL agents under the Q-learning algorithm by a novel simulated market environment framework which consistently induces stable learning that generalizes to out-of-sample data.

A framework was proposed based on deep reinforcement learning, to autonomously make trading decisions and gain profits in the dynamic financial markets[15].

An innovative approach based on deep reinforcement learning (DRL) to solve the algorithmic trading problem of determining the optimal trading position at any point

in time during a trading activity in the stock market was proposed in [16].

We have developed a model that can outperform the generic strategy by the application of reinforcement learning on past stock data and train the model to take the optimal actions at any point of time, while mitigating the risk generated by the erraticism of the stock market.

3. Proposed Method

The first step involves applying some basic technical analysis to the choice of stock. First, we obtain the historical dataset using finance library of Python. This data will be in the form of candlestick data, which contains the datetime, open, high, low, close and volume for every hour. This historical data of a stock is taken and cleaned to remove all garbage data. Then we use a basic algorithm to calculate the pivot highs and lows i.e. the resistance levels and the support levels at each state. These levels are calculated by finding out the maxima and minima of every 7 candlesticks. The underlying strategy is to take an action based on the closest level at each state. This will be further explored in the next section about the model training.

3.1. Model Training

3.1.1. The problem statement

Our goal is to use reinforcement learning and train the model to find the optimal action at each state in the environment, based on the underlying strategy of resistances and supports. We shall be exploring the same using Deep Q-Learning, a branch of Q-Learning involving Deep Neural Networks.

3.1.2. Deep Q-Learning

In basic Q-Learning, a matrix (a Q-table) is created for the agent. The agent refers to this table to maximize future rewards based on the discount factor too. The drawback of this approach is that the Q-values only have relative importance. Deep Q-Learning proposes a deep neural network for calculating the Q-values and finding the optimal action.

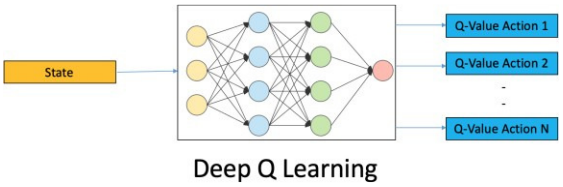

Figure 3.1 Deep Q-Learning architecture.

$$Q^{new}(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}}$$

Figure 3.2 Q-State Formula.

3.1.3. The Markov Decision Process

In the proposed system, the different states are the different candlesticks at a given timestamp. The state inputs are defined by calculating the distance from the closest support and resistance level, the current balance remaining and the slope of the last 5 candlesticks. The agent is allowed to take 3 actions:

1. Buy: The agent will buy x amount of shares of the stock
2. Sell: The agent will sell x amount of shares of the stock
3. Do Nothing: The agent will do nothing for that state

The reward schema is based on giving rewards for performing actions instead of just giving the actual profit or loss, since that would just lead to random actions. The agent is rewarded for buying near a level, for selling for a profit near a level, and holding if none of the other two criteria are satisfied. The agent is penalized for depleting the balance too much, or overselling, and also for staying idle so that the agent does not keep choosing Do Nothing. The terminal state is defined by the last candlestick, or when the balance is completely depleted, or when the agent tries to sell when there are no shares being held.

3.1.4. Deep Q-Network and training

A basic artificial neural network is used, with 3 dense layers and one output layer. The input is defined by the feature set (i.e. 8) and the output has 3 nodes for the 3 respective actions. The discount factor is set to 0.85 and the learning rate is 0.005. The optimizer used is Adam.

A replay memory buffer is also used for storing previous experiences of the agent as well, so that the agent does not overfit to the more recent data without having any knowledge of the past experiences. The Q-

learning updates are incremental and do not converge quickly, so multiple passes with the same data is beneficial, especially when there is low variance in immediate outcomes (reward, next state) given the same state, action pair. This memory is updated in batches, and stored as a deque, since the memory is limited.

The reason for choosing Q-learning over other frameworks is two folded: 1. Q-learning is model-free. We can learn directly from interacting with the environment without modeling the environment's distribution. 2. Q-learning is off-policy. We do not need to initiate a whole trial of iteration for each state, and only update the visited pair of state and action.

Each time an action is selected, the system provides a small probability (ϵ) for exploration, in which the best action is abandoned and other new actions are executed on a random basis. The probability of exploration increases concurrently with the value, making the agent more likely to try a new state or action. This process improves the learning effects. Epsilon greedy method is used to tradeoff between exploration and exploitation. As Epsilon (ϵ_{min} = 0.01) value decreases the exploitation of the model increases.

The problem encountered with a vanilla Deep Q-network is that since the same network is calculating the predicted value and the target value, there could be a lot of divergence between these two. To solve this issue, we use two networks (Double DQN). The main neural network is used to adjust the parameters and the target network, which has the same architecture as the function approximator but with fixed parameters, calculates the target. After a certain number of iterations, the main network parameters are copied to the target network.

This model is trained and tested on the following stocks: ASIANPAINT, BPCL and TCS. The reason for choosing these stocks is that these stocks have been through all 3 market phases in the selected training dataset; uptrend, downtrend and sideways. The agent begins with an opening balance of Rs. 100000 for each stock.

4. Results

Graphical Representation of the Episode training and portfolio appreciation for the stocks are given below. The benchmark model used for all stocks is the generic strategy of Buy and Hold.

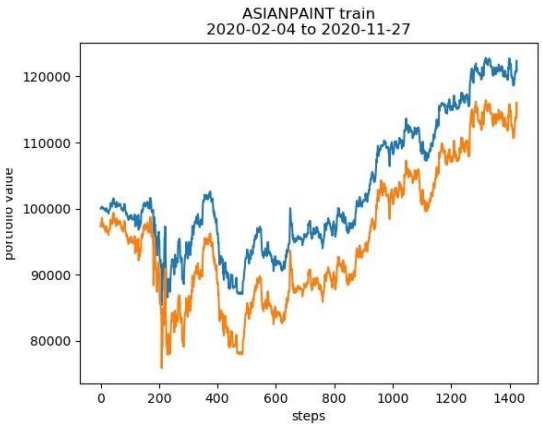

Figure 4.1 ASIANPAINTS-Train.

Model	Portfolio growth (%)	Drawdown (%)
Generic	12.69	21.42
DQN	21.61	13.7

Table 4.1 Comparison of generic model and DQN for ASIANPAINTS train dataset.

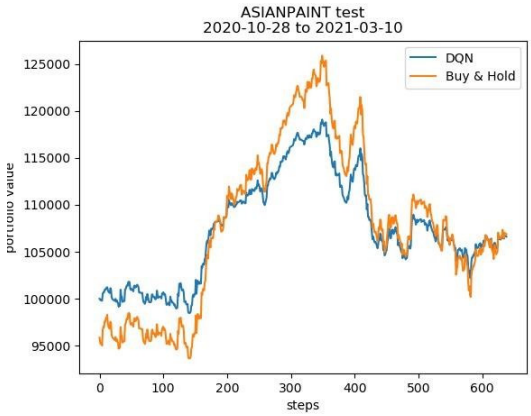

Figure 4.2 ASIANPAINTS-Test.

Model	Portfolio growth (%)	Drawdown (%)
Generic	10.73	4.11
DQN	7.2	1.12

Table 4.2 Comparison of generic model and DQN for ASIANPAINTS test dataset.

Training duration -04/02/2020 - 27/11/2020
Test duration - 28/10/2020 – 10/03/2021

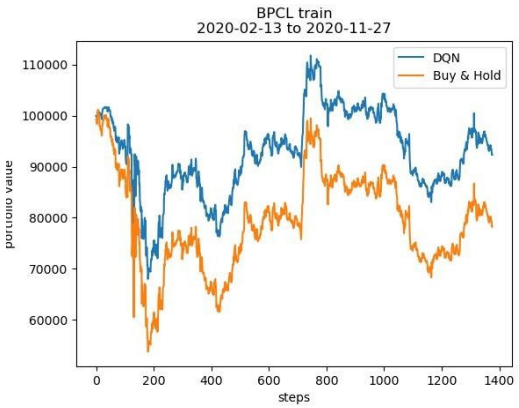

Figure 4.3 BPCL- Train.

Model	Portfolio growth (%)	Drawdown (%)
Generic	-21.87	46.42
DQN	-7.3	28.89

Table 4.3 Comparison of generic model and DQN for BPCL train dataset.

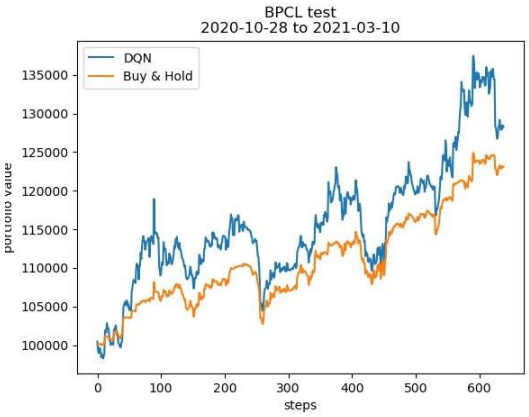

Figure 4.4 BPCL- Test.

Model	Portfolio growth (%)	Drawdown (%)
Generic	21.04	0.78
DQN	27.16	3.22

Table 4.4 Comparison of generic model and DQN for BPCL test dataset.

Training duration -13/02/2020 - 27/11/2020
Test duration - 28/10/2020 – 10/03/2021

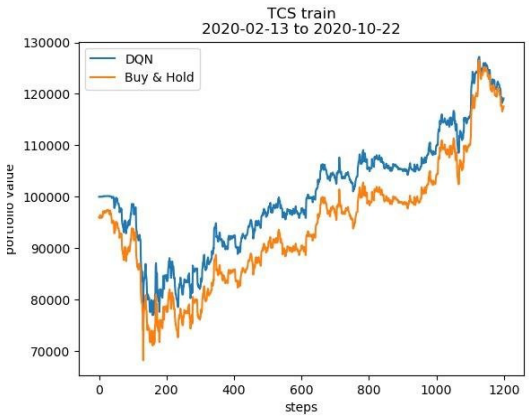

Figure 4.5 TCS-Train.

Model	Portfolio growth (%)	Drawdown (%)
Generic	18.88	31.1
DQN	20.47	21.7

Table 4.5 Comparison of generic model and DQN for TCS train dataset.

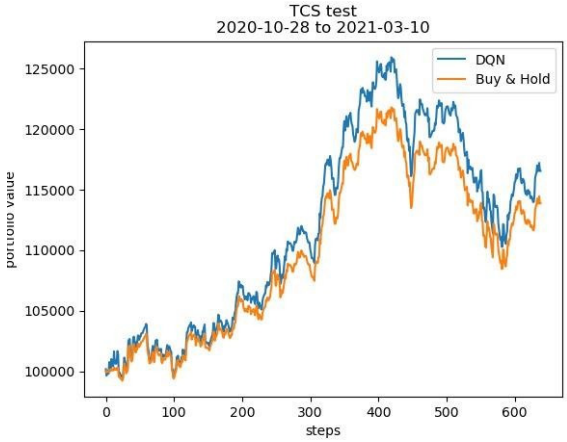

Figure 4.6 TCS-Test.

Model	Portfolio growth (%)	Drawdown (%)
Generic	13.9	2.07
DQN	16.09	0.93

Table 4.6 Comparison of generic model and DQN for TCS test dataset.

Training duration – 13/02/2020 - 22/10/2020
Test duration - 28/10/2020 – 10/03/2021

5. Conclusion and future dimensions

The model successfully outperform the generic retail trading model in most cases. The drawdowns were substantially low with the appreciation also being better in most cases albeit a few. It should be noted that the training duration included the steep market fall of March 2020. Hence the double DQN model is successfully able to learn to take optimal actions and hedge the drawdowns by a substantial amount by utilizing the balance amount properly. Future improvements on the model could include implementing a discount and decay factor for the rewards to include the profits/losses as well i.e. a trade-off between rewards for performing the actions as well as the profits or losses obtained. The use of a Recurrent Neural Network could also be explored, since it is widely used for time-series forecasting.

6. References

[1] Abhishek Nan ; Siddharth Perumal ; Osmar R Zainane "Sentiment and Knowledge Based AlgorithmicTrading with Deep Reinforcement Learning" IJSTE - International Journal of Science Technology & Engineering Volume 3 Issue 10 April 2017.

[2] Akhil Raj Azhikodan ; Anvitha G. K. Bhat ; Mamatha V. Jadhav "Stock Trading Bot Using Deep Reinforcement Learning" 7th International Conference on Communication, Computing and Virtualization 2016.

[3] Siddharth Das "An Automated Stock Trading Heuristic, Dependent on Volatility, for Common People" 2018 IEEE International Conference on Electro/Information.

[4] Huang, Chien. (2018). Financial Trading as a Game: A Deep Reinforcement Learning Approach.

[5] Wang, Chun-Chieh & Tsai, Yun-Cheng. (2019). Deep Reinforcement Learning for Foreign Exchange Trading.

[6] Wanjawa, B.W., & Muchemi, L. (2015). ANN Model to Predict Stock Prices at Stock Exchange Markets. *ArXiv, abs/1502.06434*.

[7] Trippi, Robert & Desieno, Duane. (1992). Trading Equity Index Futures With a Neural Network. Journal of Portfolio Management - J PORTFOLIO MANAGE. 19. 27-33. 10.3905/jpm.1992.409432.

[8] Dempster, Michael & Payne, Tom & Romahi, Yazann & Thompson, G.W.P.. (2001). Computational learning techniques for intraday FX trading using popular technical indicators. Neural Networks, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 12, NO. 4, JULY 2001.

[9] Ibrahim, Alaa Eldin. (2014). Evolutionary Approach to Forex Expert Advisor Generation. Intelligent Information Management. 06. 129-141. 10.4236/iim.2014.63014.

[10] Chakole, Jagdish & Kolhe, Mugdha & Mahapurush, Grishma & Yadav, Anushka & Kurhekar, Manish. (2020). A Q-learning Agent for Automated Trading in Equity Stock Markets. Expert Systems with Applications. 163. 113761. 10.1016/j.eswa.2020.113761.

[11] Carta, Salvatore & Ferreira, Anselmo & Podda, Alessandro & Reforgiato Recupero, Diego & Sanna, Antonio. (2021). Multi-DQN: an Ensemble of Deep Q-Learning Agents for Stock Market Forecasting. Expert Systems with Applications. 164. 10.1016/j.eswa.2020.113820.

[12] Liu, Y. (2019). "Reinforcement Learning Applications in Real Time Trading," Joseph Wharton Scholars. Available at https://repository.upenn.edu/joseph_wharton_scholars/65

[13] Azhikodan, Akhil & Bhat, Anvitha & Jadhav, Mamatha. (2019). Stock Trading Bot Using Deep Reinforcement Learning. 10.1007/978-981-10-8201-6_5.

[14] Carapuço, João & Neves, Rui & Horta, Nuno. (2018). Reinforcement learning applied to Forex trading. Applied Soft Computing. 73. 10.1016/j.asoc.2018.09.017.

[15] Li, Yang & Zheng, Wanshan & Zheng, Zibin. (2019). Deep Robust Reinforcement Learning for Practical Algorithmic Trading. IEEE Access. 7. 1-1. 10.1109/ACCESS.2019.2932789.

[16] Théate, Thibaut & Ernst, Damien. (2021). An Application of Deep Reinforcement Learning to Algorithmic Trading. Expert Systems with Applications. 114632. 10.1016/j.eswa. 2021.114632.