Deep-Reinforcement Learning-Based Architecture for Multi-Objective Optimization of Stock Prediction

Rangappa Jyothi and Gorappa Ningappa Krishnamurthy

Abstract — **Artificial Intelligence has been established to predict the future performance of the trading in modern era including Statistics, Computer Science, and economics, especially the stack market. By analyzing the big data, making the financial decision is important for investors in the stock market. As records, several techniques like Back Propagation Neural Network (BPNN) Recurrent Neural Network (RNN) are used to predict the stock price but due to its computation complexity is major challenging of time serious financial data in the decision-making process. In this paper, we have proposed the Recurrent Q Network Learning (RQNL) to make the right decision according to movements of stock prices as reliable attention. To overcome the computational complexity shortcoming, the forgotten memory is developed in a neural network. In this way, the error arousal is pruned in a significant manner. In addition, Enhancing the prediction ability is outstanding, and reinforcement learning is demonstrated to reduce the correlations in time series data. Discussion and evaluation were based on the NSE dataset are experimented with three different learning approaches, with the proposed method for stock prediction of financial markets. The experiment result carried out that our proposed Recurrent Q Network Learning (RQNL) performs the perfect prediction compared with other existing learning.**

Keywords — **Deep Learning, LSTM, Price Prediction, Recurrent Neural Network, Reinforcement Learning, Q network.**

I. Introduction

A major goal of portfolio managers, financial organizations as well as individual traders automates the operation of trading in markets stock because of its enhanced complexity [1]. Although, the efforts of automatic trading records of digital stock markets are enhancing research, with directed as well as motivated availability i.e., volumes of trading & prices. Even though projecting forecast prices is an important challenge in finance markets, other issues are equally important since they integrate forecast prices with the actions that must be taken as a result.

For managing portfolio inputs in stock prices in the future, which concerns projections for Kelly criteria and Modern Portfolio Theory (MPT) [2]. The criterion of Kelly is based on the probability distribution for future returns, whereas MPT is primarily reliant on predicted mean and variance on the accuracy. However, even minor variations in the implementation of these techniques can have a significant impact on the portfolio's weights [3]. The fundamental issue using MPT, according to Michaud, is that it tends to have huge mistake effects in the assumptions of input. This methodology is inadequate for portfolio management in actual stock markets [4] due to challenges in projecting the mean and variation of future commodity prices. The process of management that holds the balance of stocks is relatively strong along with the relatively weak stocks selling as called as Equity Market Neutral (EMN) [5]; thus, classifying as well as quantifying stocks based on their strengths relative as well as less strong is challenging.

In computational finance, Reinforcement Learning (RL) has been employed as an alternate strategy. To construct a trading system, RL was integrated with neural networks [6] as well as genetic algorithms. RL is also used to trade American options in part in [7]. Synthetic data were employed in each of the previous experiments. RL was employed to teach agents for the market-making challenge in [8]. The first large-scale application of RL to tackle the Online controlled experiments (OEC) problem was in [9]. When a limit order book [10] approach was suggested employing a limited set of assumptions about the market structure and combining techniques from dynamic programming and RL. When compared to the buy-and-hold trading approach, the trading policies developed in [11] are superior.

Human investment actions are analogous to the RL's learning mechanism. Investors who invest the money able to gain information for future investments through trial and error (investment decision) as well as incentives from relationships between investors and or financial markets. Hence, a Portfolio Management System (PMS) is implemented successfully via networks of CNN as well as the architecture of RL to limit the mentioned overall issues & to allocate support resources i.e., decision making.

II. Related Work

In 2019, Meisheri *et al.* [12] presented a solution of data-driven to sequential class in decision-making issues involving a huge number of concurrent online decisions, with applications in operations research as well as computer systems. An issue of canonical Reinforcement Learning (RL) is modeled to make the decision, which was rectified utilizing the technique of purely data-driven. To perform the task of multi-product inventory approach in a specific instance, to ensure applicability it also modified as conventional strategy i.e., Advantage Actor-Critic (A2C). To vary data in the learned policies, the presented outcomes probe due to their

Submitted on April 28, 2022.
Published on July 31, 2022.
R. Jyothi, Assistant Professor, PES University, India.
(e-mail: jyothir@pes.edu)

G. N. Krishnamurthy, Principal BNM Institute of Technology, India.
(e-mail: bnmitprinciple@gmail.com)

robustness.

In 2020, Chu *et al*. [1] have presented More MNAS (Multi-Objective Reinforced Evolution in Mobile Neural Architecture Search) was new technology i.e., multi-objective oriented, which leverages good virtues from both RL&EA. It included the cell level that performs mutation as well as crossovers to compose numerous cells in search space to integrate the technique known as multi-objective genetic (NSGA-II) variant. In the domain of Super-Resolution (SR), the simulation was modeled to deliver rivaling as compared to the existing approach known as state-of-the-art with FLOPS in fewer.

In 2020, Bisht *et al*. [14] have discussed stock trading utilizing the system known as multi-objective-based Deep Reinforcement Learning (DRL). To maximize the profit of multi-objective systems that minimizes the risk. To design two Deep Neural Networks (DNNs), the architecture of the entire system was modeled. To achieve the investor's goal, the first is an LSTM auto encoder for robust feature extraction, and the second is deep reinforcement learning with an LSTM recurrent neural network for decision making. For the system verification, data of real historic was conducted and compared with other systems of trading.

In 2018, Xiong *et al*. [15] have presented the strategy of stock trading, the DRL’s potential was optimized to maximize the return investment. The trading stocks are chosen from a pool of 30 stocks, and their daily prices serve as a trading market environment as well as training. A strategy of adaptive trading was obtained to train the agent of DRL. In terms of the Sharpe ratio and cumulative returns, the proposed DRL approach outperforms the two baselines.

In 2019, Yu *et al*. [16] have presented the unique architecture of RL, which modeled the purely relying upon an adversarial generative Data Augmentation Module (DAM), a Behavior Cloning Module (BCM), as well as Infused Prediction Module (IPM).Both on-policy and off-policy RL algorithms are supported by the model-based approach. The trading with practical constraints using historical real financial market data was simulated, and the proposed model was profitable, more robust, as well as risk-sensitive than strategies of baseline trading & agents of model-free RL from conventional techniques.

In 2021, Carta *et al*. [17] have presented the stock trader in multi-ensemble as well as multi-layer to tackle the proposed issue. The process began with hundreds of DNNs preprocessing data. Then, through successive variations, a classifier of reward-based as a meta-learner to maximize stock signals generation as well as profit. Finally, a decision trading combines several meta-learners to form more stable strategy trading, with the final decision by agents of several trading. This method performed all consideration baselines (which continue to perform in the studied period) and in the traditional Buy-and-Hold strategy, which replicates behavior in the market.

In 2020, Aboussalah *et al*. [18] have presented the architecture of SDDRRL (Stacked Deep Dynamic Recurrent Reinforcement Learning) to optimize portfolio in construction. The methodology monitors the state of the market and re-establishing the portfolio as required. Sharpe ratio, one of the most generally recognized measures of risk-adjusted rates of return, has been used as a performance metric under this broad vision. In addition, the settings of hyperparameter, the technique of ML performance depends effectively. The results of the experiments show that the proposed SDDRRL outperforms three benchmarks: the rolling horizon risk parity model, the Uniform Buy-And-Hold (UBAH) index, as well as the rolling horizon Mean-Variance Optimization (MVO) model.

In 2018, Liang *et al*. [19] have presented the Deep Deterministic Policy Gradient (DDPG), Proximal Policy Optimization (PPO), as well as Policy Gradient (PG) were the three techniques of RL in continuous state-of-art. The effectiveness of numerous conditions, i.e., objective functions, feature combinations, as well as different learning rates to provide insights for tuning parameters, selection of feature, as well as preparation of data. The extensive Chinese stock market has revealed in PG was more desirable in the financial market than PPO as well as DDPG, even though both were more advanced. The investigations showed that agent based on Policy Gradient outperforms UCRP as an outcome of new modification.

In 2018, Spooner *et al*. [20] have presented a simulation of limit in high-fidelity to book markets and also utilize it to design an agent of market-making employing temporal-difference reinforcement learning. As a value function approximator, uses a linear combination of tile coding’s and create a custom reward function that controls inventory risk. The effectiveness of our approach is demonstrated by demonstrating that our agent outperforms both simple benchmark strategies and a recently published online learning approach.

III. Proposed Methodology

The rapid investigation of deep and reinforcement learning makes the financial market perform the process of trading in automatic [21]. In which the prediction-based process plays a virtual role to makes the profits for investors through the proper decision making [22] based on economic and political conditions, Performance of companies, and much on. As a result, numerous research papers were introduced based on prediction in stock price and its movement, Stock trends, Future stock price, Time period, etc. [23]-[25]. But, complex manner of nonlinear characteristics, already taken a decision, Cost, Time constraints are challenging one as well as consider on decision making in stock market prediction. Accurate decision-making of stock market movements is achieved through analysis of the environment and gives proper feedback to stock market returns. In order to obtain accurate decision making, the complex data should be evaluated appropriately and becoming extract the relevant data. As to give a solution to all problems, we have introduced the proposed Recurrent Q Network Learning method includes preprocessing, feature extraction, and an accurate decision-making process to obtain the stock market prediction without error arousal. The proposed approach of prediction of the stock price in the financial market is given in Fig.1.

The collection of data needed to be preprocessing approach in time series data for its complex feature information. It can minimize the cost of neural network training while training the large time series. This advantage makes prone the large dimension of input data [26]. After completion of preprocessing process only, the data can be allowed to deep learn which is used to permit various layers and learning the of its characteristics in an appropriate manner, to pass the suitable activation of the network.

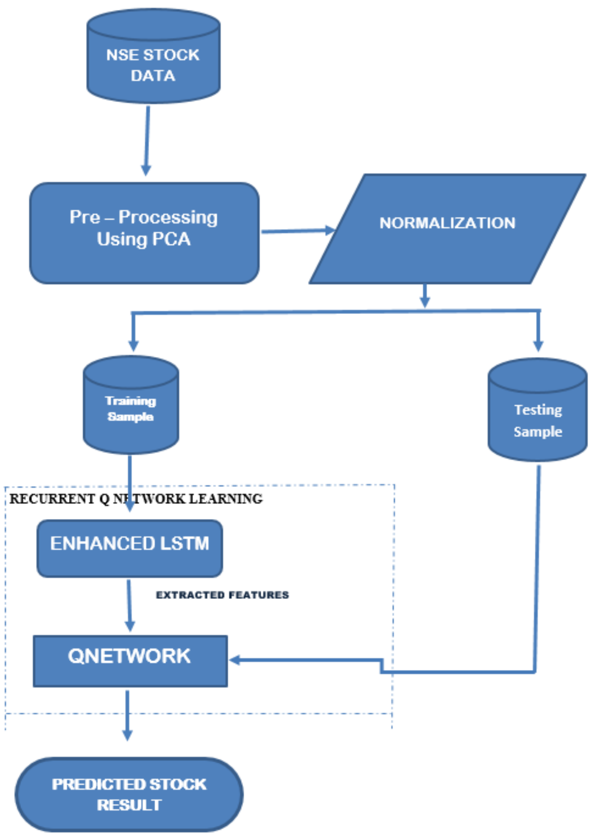


Fig. 1. Proposed Prediction Model Architecture.

A. *Preprocessing*

Thus, preprocessing approach of the PCA algorithm is applied before inserting the data in deep learning as necessary. The reason behind adding preprocessing is that PCA does not have any time step to form the input matrix before the Deep Learning process. The numerous time steps as a parameter are important in the training process of deep learning. Hence, we have to model the matrix into corresponding time steps for both the training and testing dataset.

B. *Eigen Value-based Normalization*

After the preprocessing, the data are considered as eigenvalue vector Z=$[z_1, z_2, z_3, \ldots z_n]^T$can be acquired due to its numerous data. Then, normalization is developed over eigenvalues. Thus, the eigenvectors are divided into a training sample and a testing sample. Here, Ytrain = $[y_1, y, y_3, \ldots y_s]^T$ along with s sample are chosen as training sample, other vectors are considered at the Testing (Predicted) sample as Ytest = $[y_{s+1}, y_{s+2}, y_{s+3}, \ldots y_n]^T$. The training and testing samples are reconstructed into matrix format for multiple values flow through the enhanced LSTM.

In addition, the feedback process is another important approach to enhance the performance of prediction. But remember the information in long term is a shortcoming in deep learning for aggregate performance. To avoid this problem, the special type of Recurrent Neural Network (RNN)is introduced as Long Short-Term Memory (LSTM) to able to learn information in a long time. Besides, it can be performed in variable-length learning. In this paper, we use modified LSTM which is designed to solve the feedback problem due to its complex data. The alternative model of recurrent neural networks (RNN) which is not able to handle long-term dependencies is LSTMs architectures. It has the ability to learn are basically long-term dependencies. Besides, this architecture designs to able to handle time-series learning [27], [28]. A classic LSTM construction with a specific feature concerning the simple cell unit is following as

C. *Forgetting Gate of LSTM*

The cell state is amplified using the inputs of the present time, the outcome of the sigmoid function, and the output of prior time. The prior unit state is tracked through the process of gate function. If the sigmoid function returns 0, a portion of the information must be forgotten; else, the information must be relayed throughout the United States.

D. *Input Gate of LSTM*

Old unit status is tracked through a function of the gate. The earlier forget gate layer, which is implemented by the gate layer, specifies what kind of information is ignored or added. Whether which information wants to be forgot or added is determined by the gate layer. The gate, which is made up of the secondary sigmoid + tanh function, defines which data should be given to the state. It is split into two sections here. The sigmoid layer, for example, specifies which values will be changed. The new information is uploaded into the state using the tanh layer, and this component is the same as the first layer. For instance, in the previous statement, the subject state could be replaced.

E. *Output Gate of LSTM*

The first two doors are mostly aimed at keeping track of the penetration line's status. The knowledge on the penetration line and the output of the current input information calculating module is utilized to compute through the third door. At time t, the gate control device regulates how much of the state value is shown to the outside. What information has been updated still has to be discarded and what information has to be added.

The mathematical representation of the LSTM memory cell is shown below:

$$a_t = \sigma\,(S_a y_t + U_a h_{t-1} + b_a \tag{1}$$

$$d_t = \sigma\,(S_d y_t + U_d h_{t-1} + b_d \tag{2}$$

$$\tilde{C}_t = tanh\,(S_c y_t + [S_c h_{t-1}, y_t] + b_c \tag{3}$$

$$C_t = a_t * C_{t-1} + d_t * \tilde{C}_t \tag{4}$$

$$e_t = \sigma\,(S_e y_t + U_e y_t + b_e \tag{5}$$

$$n_t = e_t * tanh\, C_t \tag{6}$$

where weight of LSTM is represented as S_a, S_d, S_c, S_e and U_a, U_d, U_c, U_e; each cell bias is denoted as b_a, b_d, b_c, b_d; cell state denotes as C_t; forget gate denotes d_t; input gate and output gate as a_t, e_t; memory cell output is indicated as h.

As a result, the input data (the collection of financial time-series of the cross-currency selected to construct a triangular arbitration) will be analyzed in depth by the LSTM network during the data processing phase, establishing in clear terms input data dynamics to preserve in memory and discard (forget gate).

To minimize the complex calculation, the enhanced version of LSTM is constructed by elimination output and input gate of the memory cell. In addition, for accurate prediction purposes, the training data is optimized with the Q network in this paper. The enhanced memory cell controls the information of the previous state is fed into the present cell state. But the adaptation of information in a hidden state is more difficult. For this reason, the attention gate is introduced by integrating the attention mechanism [28] which is the interior of the memory cell. To enhance prediction accuracy, long-term memory is applicable. The following Fig.4 and process has the information of enhanced LSTM memory cell:

$$d_t = \sigma\,(S_d y_t + U_d h_{t-1} + b_d (7)$$

$$\tilde{C}_t = tanh\,(S_c y_t + [U_c h_{t-1}, y_t] + b_c \quad (8)$$

$$a_t = \sigma\,(S_a y_t + U_a h_{t-1} + b_a \quad (9)$$

$$d_t = \sigma\,(S_d y_t + U_d h_{t-1} + b_d \quad (10)$$

$$\tilde{C}_t = tanh\,(S_c y_t + [S_c h_{t-1}, y_t] + b_c \quad (11)$$

$$Ay_{t,}h_{t-1} = q^T\, tanh(S_a y_t + U_a h_{t-1}) \quad (12)$$

$$\alpha_t = \frac{\exp(A\,(y_{t,h_{t-1}}))}{\sum_{i=1}^{n} \exp(A(y_t, h_{t-1}))} \quad (13)$$

$$\tilde{C}_t = \tanh(\alpha_t) * (S_{\tilde{c}}\, y_t + U_{\tilde{c}} h_{t-1} + b_{\tilde{c}} \quad (14)$$

$$C_t = d_t * C_{t-1} + (1 - d_t) * \tilde{C}_t/2 + (1 - d_t) * \tilde{C}_t/2 \quad (15)$$

$$C_t = h_t \quad (16)$$

where q denotes a vector of attention gate; S and U indicate weight matrices; t is the critical vector. the update gate and attention gate are used to promote the memory and take the perfect role to perform the accurate prediction through different functions. By updating in state of the cell at the current time, an update gate is performed. the attention gate has the input as the output of the update gate. Besides, the attention gate plays the role for concentrate valuable information by eliminating the irrelevance information in a precise manner. As a result, the network estimation is boosted and achieved target prediction accuracy in the proposed approach.

The estimation model can examine the accuracy of the prediction results. The normalized data are fed into the input layer. After that 4 FC layers are used as a hidden layer for future learning. Superimposed multiple memory units are used to develop the prediction model. In addition, each neuron transfers the information among them to mine the cover connection among the layers and also recognize the self-connection over on spatial-temporal correlation. In addition, the Q network is merged with enhanced LSTM in training processes. It is called a deep Q network because reinforcement learning is associated with the recurrent neural network as LSTM. In this way, 2 Fully Connected layers are constructed in which the agent participates in the way of a series of recognize, performance, and target are considered. The purpose of the agent is to choose action in such a way that the total future target is maximized. To estimate the optimal action-value function, we employ an enhanced LSTM.

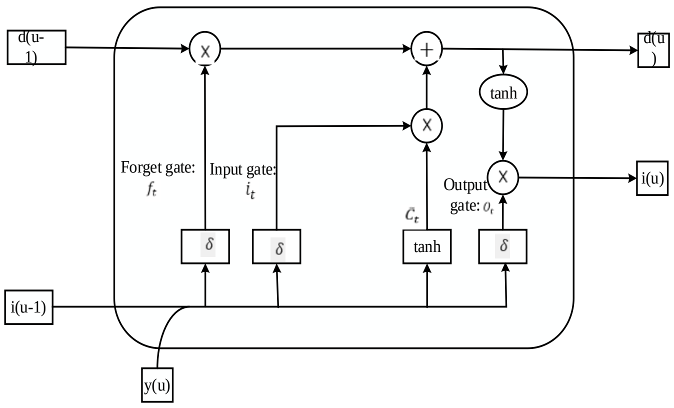


Fig. 2. Enhanced LSTM Model.

Reinforcement learning [29] is also called volatile or even diverges due to the combining of a neural network to give the Q function or Action value function. The presence of correlations in the time series, the fact that slight changes to Q can considerably modify the policy and hence alter data transfer, and the correlations amid the action-values (Q) and the target values all contribute to the instability. We offer a Q-learning variat ion that employs two essential concepts to solve these instabilities. First, we applied experience replay, a biologically based method that randomizes data, reducing correlations and smoothing out variations in data transfer. In addition, we have given an iterative approach to shifting the Q (action-values) towards just periodically updated final values, decreasing correlations with the target objectives. In addition, the Q network acts as an agent to perform the target action. So that, it is observed the situation and take action. Because of that we utilize enhanced LSTM to parameterize an approximation value function Q (x, y, θ_i) where are the weights parameters of the Q-network at iteration i. x denotes observing the training features and analyzing the correlation variation stock data. As a result, the prediction output is predicted from the y value. The Proposed architecture is presented in Fig.3.

We utilize enhanced LSTM to parameterize an approximation value function Q(x,y, θ_i) where are the weights parameters of the Q-network at iteration i. To accomplish experience replay we record the experiences of agents at each time-step t on the data set. We use Q-learning updates on samples of observation taken randomly from a pool of selected samples during the learning process. At iteration i the Q-learning update employ the following MSE-based loss function:

$$L(\theta_i) = \frac{1}{n}\sum_{i=1}^{n} Q\,(y_t^i - \hat{y}_t^i)^2 \quad (17)$$

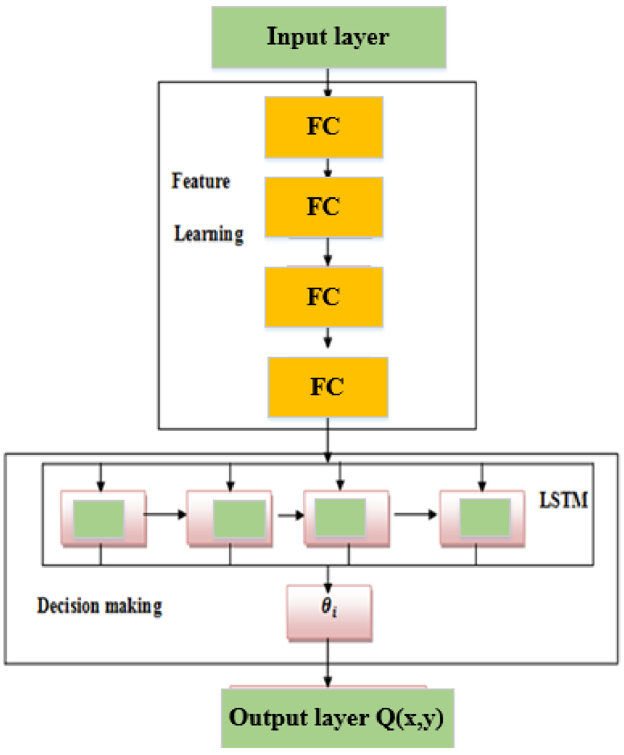


Fig. 3. Architecture of Recurrent Q Network Learning.

where the length of the training sample is denoted as n; a true value at t time y_t^i; predicted value at t time $\hat{y}\iota_t^i$; the following stages can be used to construct the flow chart for the proposed method:

Step 1: Get the National Stock Exchange Dataset.

Step 2: The dimensional is reduced using the PCA algorithm.

Step3: Normalization is performed for input values, then get training sample Ytrain and predicted sample as testing sample Ytest.

Step 4: Constructed multivariable matrix.

Step 5: The training samples are fed into the enhanced LSTM to train the model.

Step 6: Features as stock price are extracted to make future prediction.

Step 7: Accurate decision making, the neutral fitted Q iteration is achieved using Q network.

Step 8: Then, action values are adjusted to reach the target values from the training dataset.

Step 9: The estimation results are output and analyzed.

IV. Result and Discussion

This part explains the reinforcement deep learning in NN classification for predicting the stock market using the NSE dataset. The experiment was implemented using Python Software. The dataset consists of the minute-wise stock price for 1721 NIFTY 50 companies for the period of July 2018 to June 2021. Also, the error of the model can be proved using experimental data and it was compared with existing as well as proposed classification techniques.

Table I explains various classifications for predicting the error analysis for stock prediction in reinforcement learning. Here, we have used NN in that LSTM is proposed to get the minimum error values. By getting the least error values, various iterations were happened using training and testing data. In LSTM, the number of hidden layer neurons is 6. Finally, compared to all other techniques, LSTM gives the minimum accuracy of 90.21%.

TABLE I: Various Classifications for Predicting the Error Analysis

Classification techniques	Iterations	Training	Testing	No. of. hidden layers	Error in %
RNN	100	0.32567	0.1564	6	0.1732
CNN	150	0.32432	0.2732	6	0.14563
The proposed method (LSTM)	200	0.33234	0.2478	6	0.1034

A. Comparison Analysis

Fig. 4 explains MSE for epochs at 500.The compared classification techniques are RNN, CNN, and LSTM. At epochs 500, the iterations are repeated until we get the least error values. The X-axis represents epochs, and the y-axis represents MSE for corresponding classification techniques. This is the chart for the tick dataset. This gives the best error values. Fig. 4 (b) for 15-min dataset and gives the maximum MSE values.

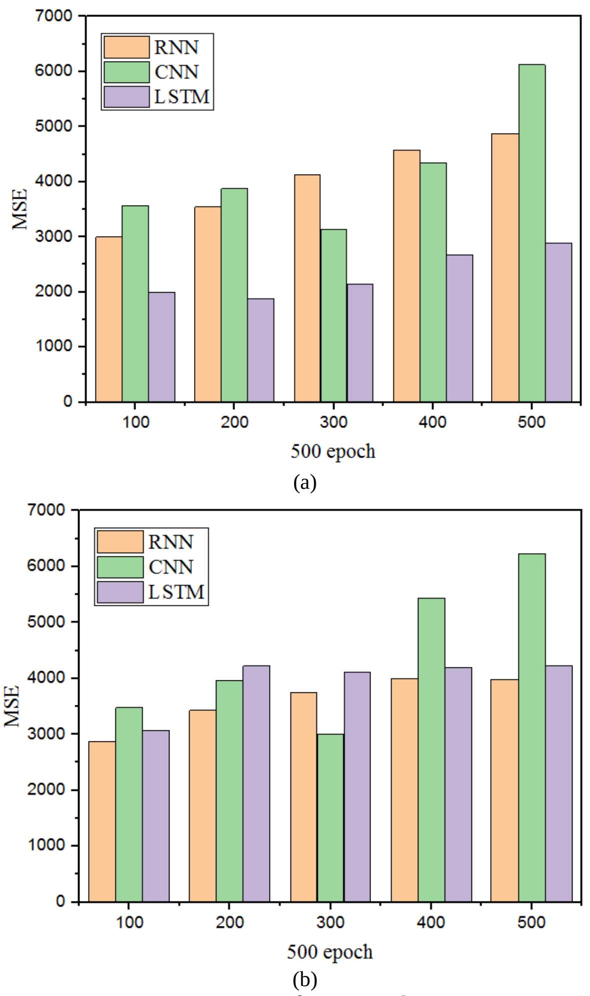
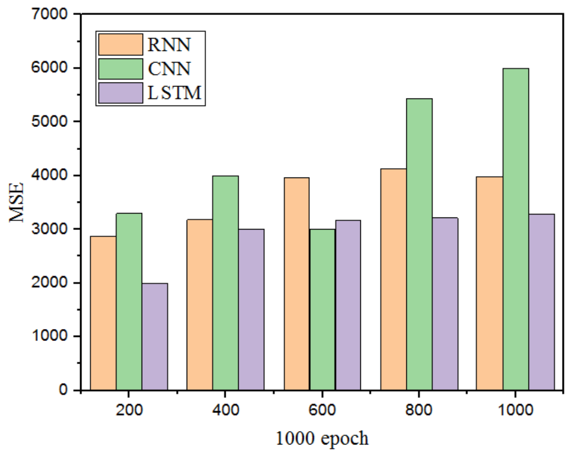
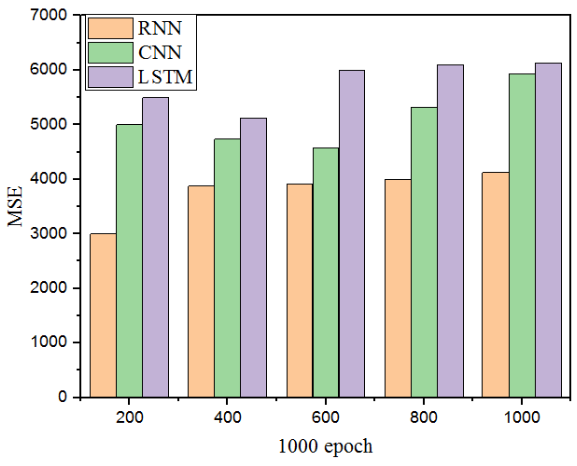


Fig. 4. MSE for 500 epochs.

Fig. 5 explains MSE for epochs at 1000.The X-axis represents epochs, and the y-axis represents MSE for corresponding classification techniques. The compared classification techniques are RNN, CNN, and LSTM. At epochs 1000, the iterations are repeated until we get the least error values. This is the chart for the tick dataset. This gives the best error values. Fig. 5(b) for 15-min dataset and gives the maximum MSE values.



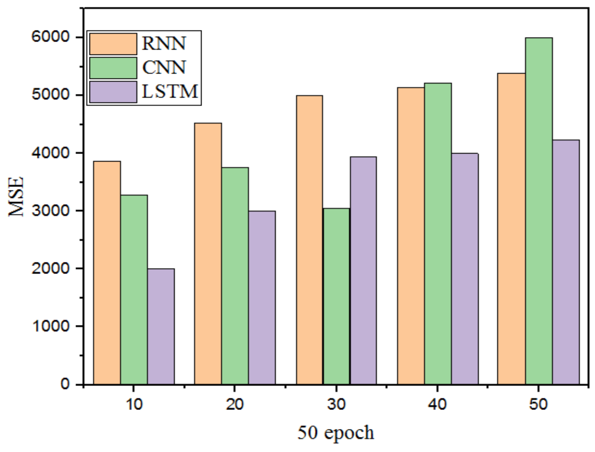
(a)



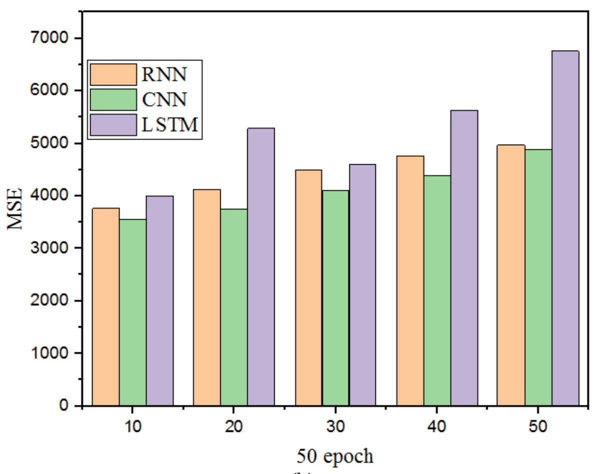
(b)

Fig. 5. MSE for 1000 epochs.

Fig. 6 explains MSE for epochs at 1000.The X-axis represents epochs, and the y-axis represents MSE for corresponding classification techniques. This is the chart for the tick dataset. This gives the best error values. Fig. 6 (b) for 15-min dataset and gives the maximum MSE values. The compared classification techniques are RNN, CNN, and LSTM. At epochs 50, the iterations are repeated until we get the least error values.



(a)



(b)

Fig. 6. MSE for 50 epochs.

Fig. 7 demonstrates the MSE analysis with NN and compared with LSTM. The X-axis represents epochs at 500 and the y-axis represents MSE. Here, the training, testing, and validation result have been taken for this analysis and compared with LSTM. We got the least error value by using LSTM.

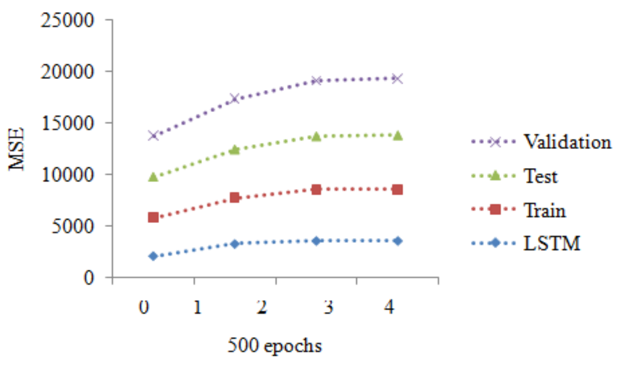


Fig. 7. MSE Analyses for 500 epochs.

Fig. 8 demonstrates the MSE analysis with NN and compared with LSTM. Here, the training, testing, and validation result have been taken for this analysis and compared with LSTM. The X-axis represents epochs at 1000 and the y-axis represents MSE. Our experimental data are trained, tested, and validated using classification techniques. Compared with other tests, we got the least error value by using LSTM.

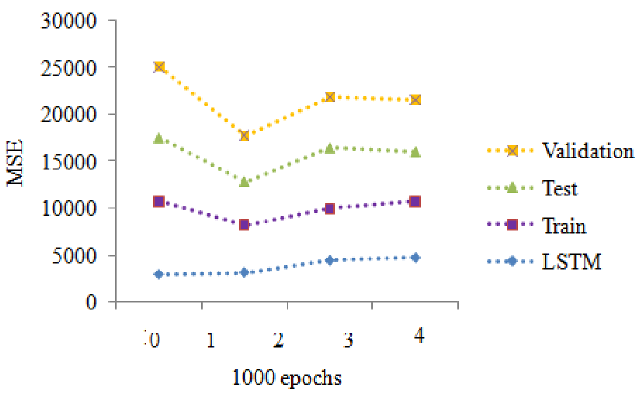


Fig. 8. MSE Analyses for 1000 epochs.

Fig. 9 explains the chart analysis for MSE with experimental training data. At epochs 50, MSE was calculated for every test result. At 0, the training data gives an error at 12000, at 1, training data gives an error at 121000, and similarly, for others, we got various error values. For our proposed LSTM, we got the least error values compared with other results.

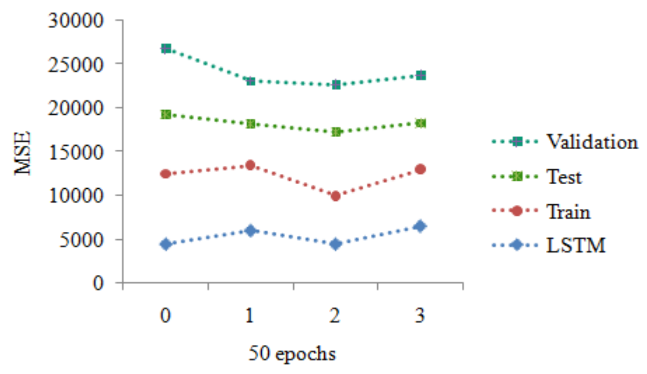


Fig. 9. MSE Analyses for 50 epochs.

B. Performance Analysis

Fig. 10 explains the chart analysis for various organizations for various years. In 2018, the performance of WIPRO gives maximum results. In 2019, Infosys gives maximum performance at 97%. The CIPLA gives 82% in 2020 and TCS gives 75% and WIPRO gives 98% in year of 2021. Like that, in other years we got maximum performance while taking the WIPRO. It gives the better performance compared to other companies such as CIPLA, INFOSYS and TCS.

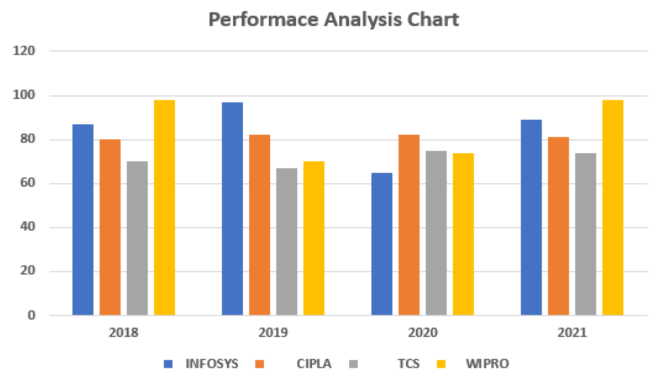


Fig.10. Performance Analysis Chart.

Every year, the percentage of profitability for item buy, retain, and sell is computed. Every time, the traders monitor the price of the product, and if the price drops quickly, they sell the product at the proper time.

Fig. 11 explains performance analysis of stock market traders in the year from July 2018 to June 2021. At each and every time of Accord synergy key statistics, the traders check the cost of the product, when the prices fall immediately; they sell the product at right time based on performance.

In 2018, the sell is 87%, buy is 77% and hold is 67% for WIPRO. In Infosys has the sell, buy, and hold percentages are 97, 77 and 87 at 2019 year. Cipla achieves sell 92%, Buy88% and hold 90% in 2020 year. Similarly, the high-performance company of TCS in 2021 has 98% to sell, 94% to buy and 96% of hold. In 2018, the sell is 87%, buy is 77% and hold is 67%. Similarly, we got different results from other years.

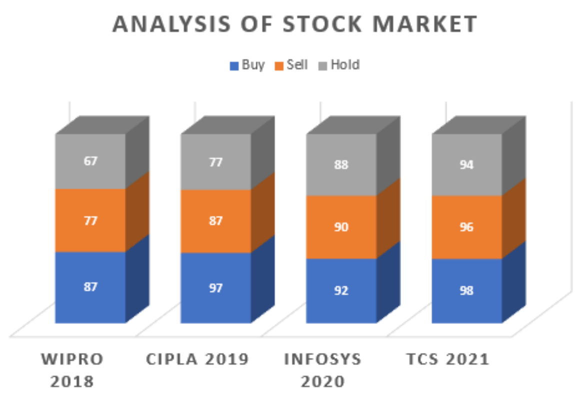


Fig. 11. Performance Analysis of Stock Market.

V. Conclusion

To enhance the accurate prediction of stock price, we have introduced Recurrent Q Network Learning in this paper. It consists of an effective feedback mechanism and a decision-making process. Thus, the calculation complexity is reduced and also features are extracted inappropriate manner. In addition, promoting the decision making is achieved through the Q network which utilizes the feedback action to construct the further action. Experiment and evaluation are carried out using existing methods along with our proposed techniques through Mean Square Error. As a result, the efficiency of stock prediction is high using Recurrent Q Network Learning through minimizing Mean Square Error compared to existing methods. Our proposed method will be analyzed by using tick by tick data for Indian Stock Markets or complicated datasets. To predict the stock market, any other hybrid model incorporating new methods such as reinforcement learning, and optimization algorithm will be introduced for future enhancement.

Reference

[1] Ricky C., Davis M., Kumiega A., and Van Vliet B. Ethics for automated financial markets. *Handbook on ethics in finance*, 2020:1–18.
[2] Wu, Mu-En, Jia-Hao Syu, Jerry Chun-Wei Lin, and Jan-Ming Ho. Portfolio management system in equity market neutral using reinforcement learning. *Applied Intelligence*, 2021;51(11): 8119–8131.
[3] Du, Guansan, Zixian Liu, and Haifeng Lu. Application of innovative risk early warning mode under big data technology in Internet credit financial risk assessment. *Journal of Computational and Applied Mathematics*, 2021;386:113260.
[4] Ahmed, Abdullahi D., and Rui Huo. Volatility transmissions across international oil market, commodity futures and stock markets: Empirical evidence from China. *Energy Economics*, 2021;93: 104741.
[5] Syu, Jia-Hao, Mu-En Wu, and Jan-Ming Ho. Portfolio management system with reinforcement learning. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 4146–4151. IEEE, 2020.
[6] Li, Yuming, Pin Ni, and Victor Chang. Application of deep reinforcement learning in stock trading strategies and stock forecasting. *Computing*, 2020;102(6):1305–1322.
[7] Ahmed, Abdulrahman A., Ayman Ghoneim, and Mohamed Saleh. Optimizing stock market execution costs using reinforcement learning. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1083–1090. IEEE, 2020.
[8] Ganesh, Sumitra, Nelson Vadori, Mengda Xu, Hua Zheng, Prashant Reddy, and Manuela Veloso. Reinforcement learning for market making in a multi-agent dealer market. *arXiv preprint arXiv:1911.05892*, 2019.
[9] Gupta, Somit, Lucy Ulanova, Sumit Bhardwaj, Pavel Dmitriev, Paul Raff, and Aleksander Fabijan. The anatomy of a large-scale experimentation platform. In*2018 IEEE International Conference on Software Architecture (ICSA)*, pp. 1–109. IEEE, 2018.

[10] Alzubi, Omar A., Jafar A. Alzubi, Mohammed Alweshah, Issa Qiqieh, Sara Al-Shami, and Manikandan Ramachandran. An optimal pruning algorithm of classifier ensembles: dynamic programming approach. *Neural Computing and Applications*, 2020;32(20): 16091–16107.
[11] Bakhach, Amer M., Edward PK Tsang, and V. L. Raju Chinthalapati. TSFDC: A trading strategy based on forecasting directional change. *Intelligent Systems in Accounting, Finance and Management*, 2018;25(3):105–123.
[12] Meisheri, Hardik, Vinita Baniwal, Nazneen N. Sultana, Balaraman Ravindran, and Harshad Khadilkar. Reinforcement learning for multi-objective optimization of online decisions in high-dimensional systems. *arXiv preprint arXiv:1910.00211* (2019).
[13] Chu, Xiangxiang, Bo Zhang, and Ruijun Xu. Multi-objective reinforced evolution in mobile neural architecture search. In *European Conference on Computer Vision*, pp. 99–113. Springer, Cham, 2020.
[14] Bisht, Kiran, and Arun Kumar. Deep Reinforcement Learning based Multi-Objective Systems for Financial Trading. In *2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, pp. 1–6. IEEE, 2020.
[15] Xiong, Zhuoran, Xiao-Yang Liu, Shan Zhong, Hongyang Yang, and Anwar Walid. Practical deep reinforcement learning approach for stock trading. *arXiv preprint arXiv:1811.07522*, 2018.
[16] Yu, Pengqian, Joon Sern Lee, Ilya Kulyatin, Zekun Shi, and Sakyasingha Dasgupta. Model-based deep reinforcement learning for dynamic portfolio optimization. *arXiv preprint arXiv:1901.08740*, 2019.
[17] Carta, Salvatore, Andrea Corriga, Anselmo Ferreira, Alessandro Sebastian Podda, and Diego Reforgiato Recupero. A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. *Applied Intelligence*, 2021;51(2):889-905.
[18] Aboussalah, Amine Mohamed, and Chi-Guhn Lee. Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization. *Expert Systems with Applications*, 2020;140: 112891.
[19] Liang, Zhipeng, Hao Chen, Junhao Zhu, Kangkang Jiang, and Yanran Li. Adversarial deep reinforcement learning in portfolio management. *arXiv preprint arXiv:1808.09940* (2018).
[20] Spooner T., Fearnley J. Savani R. and Koukorinis A. Market making via reinforcement learning. *arXiv preprint arXiv:1804.04216*, 2018
[21] Mosavi A.*et al*. Comprehensive review of deep reinforcement learning methods and applications in economics. *Mathematics*, 2020;8.10: 1640.
[22] Rouf N.*et al*. Stock Market Prediction Using Machine Learning Techniques: A Decade Survey on Methodologies, Recent Developments, and Future Directions. *Electronics*, 2021;10.21: 2717.
[23] Wang L., Hajric V. The Cost of Bad Market Timing Decisions in 2020 was Annahilation Bloomberg, 2020.
[24] Wang B, Huang H, Wang X. A novel text mining approach to financial time series forecasting.*Neurocomputing*, 2012;83(6):136–145.
[25] Guo Z, Wang H, Liu Q, Yang J. A Feature Fusion Based Forecasting Model for Financial Time Series Plos One, 2014;9(6): 172-200.
[26] Halko N, Martinsson PG, Tropp JA. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev*. 2001;53(2):217–88.
[27] Ghosh A., Bose S., Maji G., Debnath N., Sen S. Stock price prediction using lstm on indian share market. *Int. Conf. Comput. Appl. Ind. Eng*. 2019;63:101–110.
[28] Azzouni A., Pujolle G. A long short-term memory recurrent neural network framework for network traffic matrix prediction. 2017. *arXiv preprint. arXiv*:17050 5690.
[29] Enlu L, Chen Q, and Qi X. Deep reinforcement learning for imbalanced classification. *Applied Intelligence*, 2020;50.8:2488–2502.