[1]Mengrui Bao

Deep Reinforcement Learning Based Optimization and Risk Control of Trading Strategies

Abstract: - Deep reinforcement learning (DRL) based optimization leverages advanced machine learning techniques to solve complex decision-making problems in various domains. Deep learning in trading involves the application of sophisticated neural network architectures to analyze financial data and make predictions in stock markets, forex, commodities, and other trading domains. Deep learning algorithms, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), traders can extract valuable insights from vast amounts of historical market data, including price movements, trading volumes, and market sentiment. These models can learn complex patterns and relationships within the data, enabling them to forecast future market trends, identify potential trading opportunities, and manage risks more effectively. Deep learning in trading has shown promise in improving decision-making processes, enhancing trading strategies, and achieving higher returns, although challenges such as data scarcity, model interpretability, and overfitting remain areas of ongoing research and development. In this paper evaluated the stock market to achieve the financial gain to achieve the significant improvement in prediction with the stock trading. The data related to the stock market are optimized with stock trend data of 7,000 stocks situated in the United States. The estimated stock trade data were computed and processed with the Multiagent Q-learning model for the data detection and classification. A deep Q-network is developed to adaptively pick distinct action subsets and train the market making agent based on the inventory states. The experimental findings demonstrate the effectiveness of our suggested methodology in feature learning and superiority in accuracy improvement when compared to deep learning-based modelling of frequency trading patterns and standard signal processing approaches for stock trend prediction.

Keywords: trading strategies, reinforcement learning, Optimization, deep Q-network, prediction

I. INTRODUCTION:

Extensive research has been conducted on methods to forecast stock prices in order to assist investors in making optimum trading decisions. Researchers in the field of economics have established several technical indicators, such as moving averages [1In contemporary times, a considerable body of research in the field of computer science has been dedicated to the development of alternative decision support systems aimed at forecasting stock prices. Numerous contemporary systems, including machine learning methodologies such artificial neural networks, have shown enhanced efficacy in comparison to systems only reliant on conventional indicators [2]. In the financial market, trading decisions may be influenced by either human or artificial intelligence. There has been a constant increase in the use of automated systems for making trading decisions in both the foreign exchange market and the stock market. The conventional methodology for training artificial intelligence in the field of financial trading involves the extraction of unprocessed data as inputs, followed by the identification of patterns via a training approach. This process ultimately generates output that guides decision-making for the specific task at hand. The approach described is often known as machine learning, and in this specific context, it may be used to understand the underlying principles that control the processes of acquiring, selling, and executing transactions. The primary emphasis of the systematic study, however, is in the domain of reinforcement learning. Re reinforcement learning methodologies include the repeated provision of novel information to a system or agent based on unprocessed input, hence facilitating their ability to optimise a pre-established reward value. Using predictions in the financial market is a growing and widespread method [3]. However, these systems are mostly reliant on supervised learning, which poses a drawback. While this kind of learning is significant, it is insufficient on its own for acquiring knowledge from encounters with long-term objectives.

When making stock price predictions, it is more advantageous to use the relative price changes from the preceding time point as goal values, as opposed to relying on absolute prices beyond a certain time frame. This strategy is often used in supervised learning techniques for the study of time series data. The use of reinforcement learning

[1] Faculty of Economics and Management, Cangzhou Normal University, Cangzhou, Hebei, 061000, China

*Corresponding author email: baomengrui666@163.com

as a potential alternative approach for forecasting stock prices involves the effective modelling and acquisition of knowledge pertaining to both delayed and immediate rewards derived from interaction processes [4].

Reinforcement learning algorithms have been shown to successfully address the difficulty of integrating stock price prediction results with dynamic trading techniques to develop an automated trading system, according to recent study. Reinforcement learning offers a method for addressing the issue of how a self-governing agent, which can see and interact with its surroundings, may acquire the ability to choose the most advantageous behaviours to accomplish its objectives [5]. Unlike supervised learning techniques such as neural networks, which depend on input and output pairings, a reinforcement learning agent learns behaviour by actively experimenting with a dynamic environment via trial-and-error interactions. In order to get the largest average rewards from the environment, the agent seeks to compute an optimum strategy [6,7].

Therefore, when faced with the challenge of building a stock trading system that operates in a rapidly changing stock market with the goal of maximising profit, it is advisable to consider using a reinforcement learning algorithm like Q-learning to train the trading system. Multiple study findings pertaining to this topic have been published in academic literature. The Q-learning method is used to make asset allocation choices in the financial market. It also includes a risk sensitivity concept in the development of the Q-function.

The contribution of this paper is:

- To enhance the stock market data .using Z score normalization pre-processing technique for Q learning predictive model.
- To evaluate the performance of the Q-learning model against traditional forecasting methods.

The subsequent report is organised in the following manner. Section II presents a compilation of relevant literature. Section III offers a thorough elucidation of the suggested methodology, including its fundamental concept, data acquisition, and feature selection. In Section IV, we introduced a basic trading model to showcase the effectiveness of the suggested algorithm in enhancing profitability. Section V provides a comprehensive summary of the whole study.

II. RELATED WORKS:

There have been a number of study findings that have been published in the relevant literature.

This study [8] extends dynamic control theory and reinforcement learning methods to optimize asset allocation, particularly focusing on portfolio rebalancing using the expected maximum drawdown (E(MDD)) as a downside risk measure. This paper presents a novel approach to reinforcement learning (RRL) by including the Calmar ratio as an objective function. The proposed technique demonstrates enhanced performance in terms of returns when compared to conventional metrics such as the Sharpe ratio and Sterling ratio. Furthermore, this study provides evidence that variable weight real-time (RRL) portfolios exhibit superior performance compared to equal weight portfolios across various transaction cost situations.

This paper [9] introduces a novel approach to automated financial trading using deep reinforcement learning. Through a rule-based policy framework, agents are trained in a virtual environment using proximal policy optimization and risk curiosity-driven learning. Experiments on real-world stocks demonstrate superior performance compared to existing methods, showcasing the system's effectiveness in optimizing trading strategies amidst market complexities.

This paper [10] introduces a deep reinforcement learning-based trading agent aimed at optimizing portfolio management by simultaneously maximizing profit and minimizing risk. Unlike conventional methods focusing solely on profit, our agent incorporates a new target policy favoring low-risk actions. Tested on cryptocurrency market data, our agent achieved a remarkable 1800% return while offering the least risky investment strategy. Moreover, it maintains robust performance even under high market volatility or short training periods.

This paper [11] introduces an ensemble stock trading strategy employing deep reinforcement learning. By combining three actor-critic algorithms, the strategy adapts to diverse market conditions. Tested on 30 Dow Jones

stocks, it outperforms individual algorithms and traditional strategies in terms of risk-adjusted return, as measured by the Sharpe ratio.

An adaptive reinforcement learning (ARL) system for fully automated trading in foreign exchange (FX) markets is presented in this research [12]. A machine learning algorithm, a risk management overlay, and a dynamic utility optimisation layer make up the system's tiered architecture. The basic method for ARL is recurrent reinforcement learning (RRL). Most notably, the dynamic optimisation layer allows users to adjust the system's risk-return balance and does away with the need for preset model tuning parameters.

A unique deep reinforcement learning model for stock trading is presented in this paper [13], with the goal of implementing dynamic trading strategies and efficiently analysing multisource data in the intricate stock market environment. Through the use of deep neural networks, the suggested model uses stock data, technical indicators, and candlestick charts to develop dynamic trading strategies. Within the reinforcement learning paradigm, the agent represents the state of the stock market and makes trading choices based on fused aspects of several data sources.

This study [14] addresses the growing interest in quantitative trading by proposing a multiagent-based deep reinforcement learning framework for multiperiod portfolio selection. Emphasizing realistic transaction costs, our approach outperforms established strategies, offering effective portfolio management methods. Specifically designed reward functions and policy network structures enhance trading decision-making, demonstrating the framework's effectiveness in risk transfer behaviors and investment performance evaluation.

A unique deep reinforcement learning (RL) architecture for dynamic portfolio optimisation is presented in this paper [15], allowing for autonomous periodic investment choices based on a global aim. Our method includes an infused prediction module, generative adversarial data augmentation module, and behaviour cloning module, in contrast to completely model-free RL agents.

The Stacked Deep Dynamic Recurrent Reinforcement Learning (SDDRRL) architecture is presented in this work [16] in order to optimise asset trading systems in real-time while addressing issues with multi-dimensional state spaces and continuous action. Based on the state of the market, SDDRRL dynamically rebalances portfolios using the Sharpe ratio as a key performance indicator. Additionally, SDDRRL optimises hyperparameters by automated Gaussian Process with Expected Improvement, guaranteeing the best possible architectural topology.

In order to overcome the difficulties posed by the complexity of the stock market, this research [17] suggests adaptive trading tactics based on deep reinforcement learning techniques. Time-series data is processed using the Gated Recurrent Unit (GRU) to extract useful financial aspects. We offer two trading methods designed specifically for adaptive trading decisions: GDPG (Gated Deterministic Policy Gradient) and GDQN (Gated Deep Q-learning). Experimental findings on trending and volatile markets from many nations demonstrate that GDPG and GDQN outperform conventional tactics and provide more consistent returns than cutting-edge techniques.

Table 1. Related works

Ref No	Methods	Applications
8	Recurrent reinforcement learning (RRL)	Portfolio rebalancing, asset allocation, superior returns compared to traditional metrics
9	Deep reinforcement learning	Automated financial trading, optimizing trading strategies, real-world stocks
10	Deep reinforcement learning-based trading agent	Cryptocurrency market trading, achieving high returns with low risk, robust performance under market volatility
11	Ensemble stock trading strategy using deep reinforcement learning	Stock market trading, outperforming individual algorithms and traditional strategies
12	Adaptive reinforcement learning (ARL) framework	Automated trading in FX markets, eliminating the need for fixed model tuning parameters

13	Deep reinforcement learning model for stock trading	Stock market trading, dynamic trading strategies, utilizing various data sources
14	Multiagent-based deep reinforcement learning framework	Portfolio management, risk transfer behaviors, investment performance evaluation
15	Deep reinforcement learning (RL) architecture	Portfolio optimization, autonomous investment decisions, utilizing predictive and generative modules
16	Stacked Deep Dynamic Recurrent Reinforcement Learning (SDDRRL)	Real-time asset trading, continuous action and multi-dimensional state spaces, dynamic portfolio rebalancing
17	Adaptive trading strategies using deep reinforcement learning	Stock market trading, adaptive trading decisions, outperforming traditional strategies

III. PROPOSED METHODOLOGY:

The stock market prediction system consists of several components. Initially, it begins by gathering data from the stock market including information, on stock prices (opening, closing, highest lowest) trading volumes, market indices, economic indicators and other relevant factors. This raw data is then processed through a technique called Min Max scaling. This process standardizes attributes within a range (for example 0 to 1) to ensure consistent scales and minimize potential biases caused by varying magnitudes.

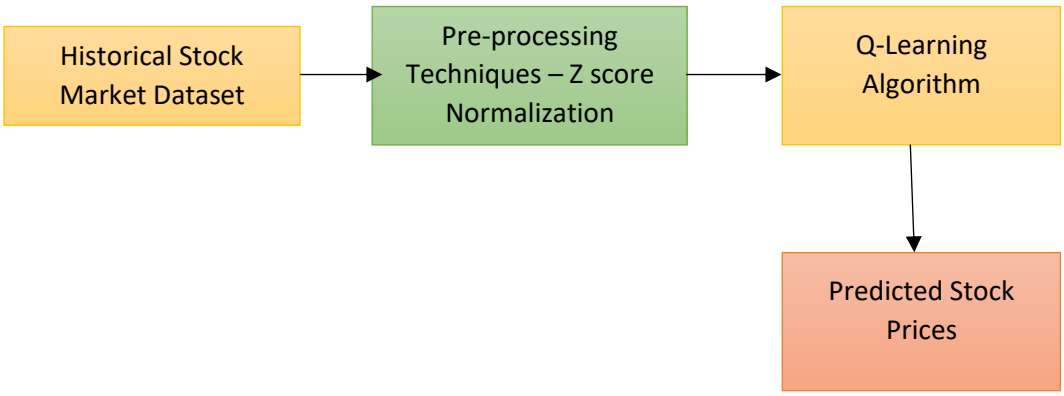

Figure. Architecture of the Proposed Model

Afterward the processed data is fed into the Q learning algorithm—a type of reinforcement learning technique used to make predictions based on stock market data. The Q learning algorithm operates within an environment where it learns the strategy for selecting actions in order to maximize rewards over time (such, as profits). It achieves this by leveraging experiences gained from the pre-processed data. Ultimately the system provides predicted stock prices generated by the Q learning algorithm as its output.

IV. PRE-PROCESSING TECHNIQUE (Z SCORE NORMALIZATION):

Stock market data research often incorporates diverse pre-processing approaches to ensure data quality, mitigate biases, and enhance the efficacy of prediction models. A statistical method called Z-score normalisation, often referred to as standardisation, is used to change numerical characteristics in a dataset such that their mean is 0 and their standard deviation is 1 [14]. It operates by rescaling the values to ensure they conform to a standard normal distribution, making the data more amenable for various analyses, particularly when dealing with algorithms that assume normally distributed features.

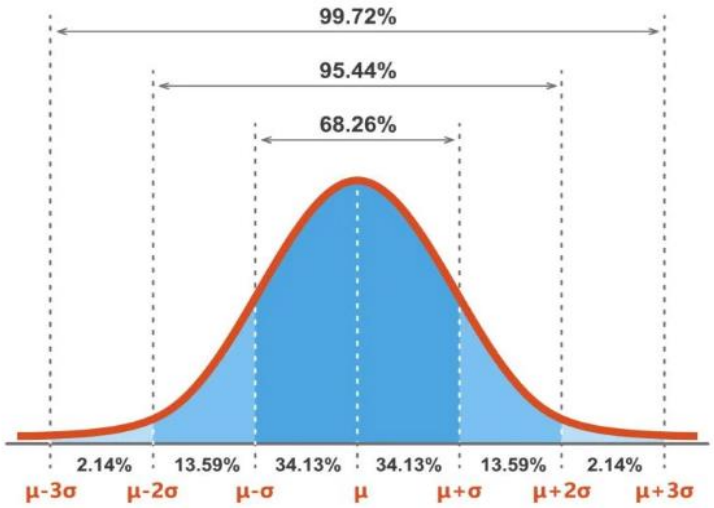

Figure.2. Z-score Normalization

The formula for computing the Z-score for a numerical attribute X is given as:

$Z=\sigma(X-\mu)$

Where:

- Z stands for the feature's standardised value (Z-score).
- X stands for the feature's initial value.
- The mean, or average, of the characteristic throughout the dataset is denoted by μ.
- σ is the feature's standard deviation, which shows how dispersed or variable it is.

Procedure:

1. **Calculate Mean and Standard Deviation:**
    - Compute the mean (μ) and standard deviation (σ) of the numerical attribute across the dataset.
2. **Standardization Process:**
    - For each value in the attribute, subtract the mean (μ) and divide by the standard deviation (σ).
    - This process centers the data around zero (μ=0) and scales it, ensuring that the resulting values have a standard deviation of 1 (σ =1).

Hence, using this pre-processing method, the input data of the stock data processed to provide better output from the proposed Q learning model

V. REINFORCEMENT LEARNING FUNDAMENTALS:

Figure 3 depicts the visualisation of reinforcement learning [15]. The reinforcement learning agent learns by interacting with the environment repeatedly, in contrast to supervised learning techniques that may analyse the whole dataset in a single run. The surroundings might have something to do with the stock market, and the agent could be thought of as a stock trader.

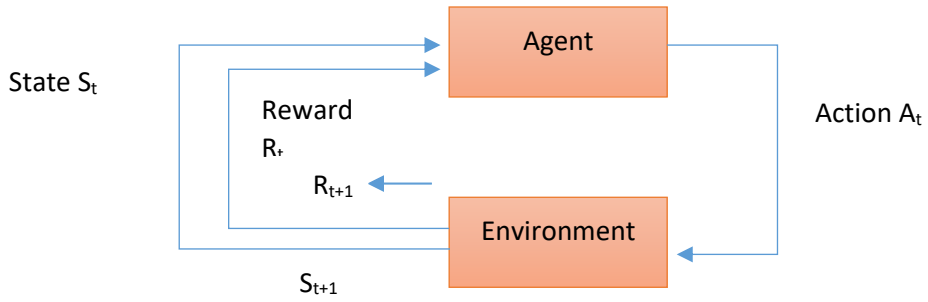

Figure 3. The communication between agent and environment in reinforcement learning

The agent acts At at time step t, and as a result, it is rewarded Rt+1 = R(St, At). After then, the surroundings change to the new state St+1 = δ(St, At).

The agent must acquire a policy π: S → A, that is, learn how to respond to the surroundings such that it may maximise the overall reward as follows:

$$V^{\pi}(S_t) = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

The coefficient γ indicates the decay factor, which is often interpreted as the interest rate in finance. It illustrates the concept that having one dollar now is more valuable than having one dollar tomorrow. Any trading strategy has to, in essence, beat the risk-free interest rate. If not, a sane investor would not think of using such tactic. Rather, they would choose to invest their funds in risk-free options like T-bonds or opening a savings account. She will be less risky and more profitable if she goes with the second option. On the other hand, we may put γ near to 1 in high-frequency trading and short time frames. The notation $\pi*$ indicates the ideal policy.

1.1 Proposed Framework:

To mimic the actions of a human investor and enhance the efficiency of solving the learning issue, MQ-Trader establishes four agents. The stock trading issue may be separated into two main components: timing and pricing. The purpose of the timing problem is to discover the optimal time for trading, while the pricing problem aims to find the optimum price for trading. This automatically results in the introduction of the following two categories of agents: 1) The signalling agent and 2) The ordering agent. Furthermore, the rationale for separating the buy signal agent from the sell signal agent stems from the recognition that an investor employs distinct criteria for decision-making when purchasing or selling a stock. When purchasing a stock, investors often take into account the potential for both increases and decreases in the stock price. Conversely, when selling a stock, the investor takes into account not only the direction of the stock price fluctuations but also the resulting gain or loss from the stock. Consequently, the division is essential to enable the agents to own distinct state representations. Specifically, the purchase signal agent utilises the price history information to predict future trends by analysing long-term price changes. On the other hand, the sell signal agent takes into account the present profit/loss together with the price history. Ultimately, the agents responsible for executing buy and sell orders produce orders to purchase and sell a stock at a predetermined price. These are referred to as bid and offer. The goal of these order agents is to determine the optimal price for trading within a 24-hour period in order to maximise profitability. Figure 1 illustrates the comprehensive learning process outlined in Q learning. The objective is to optimise investment returns by taking into account both the overall global stock price trend and the intraday price fluctuations. Within this concept, individual agents possess distinct objectives as they engage with one another to exchange experiences throughout the process of learning. To be more precise, when a stock item is randomly chosen, a learning episode begins by randomly picking a certain day, represented as δ, from the historical data of the stock item. As seen in Figure 1, the purchase signal agent initially forecasts prices by looking at the given stock's historical price changes. The potential of a price rise in the near future is therefore taken into consideration when deciding whether to go forward with the stock purchase. If you decide not to buy the stock, the current episode ends, and a new one begins on a randomly chosen day in the future. In Figure 1, the stages designated as 1a and 2a are applicable to this specific situation.

Alternatively, if the choice is made to acquire the stock on day δ, the buy order agent is notified after verifying the viability of the purchase by referencing the maximum permissible BP. After determining a suitable buying price (BP), the buy order agent proceeds to execute the actual purchase on the next day (δ + 1) if it is deemed practicable. Occasionally, the BP may be set at an excessively low level, leading to a failed transaction. During this occurrence, the purchasing agent tries several business partners until a successful transaction is achieved. The circular shape seen in Figure 1 symbolises the iterative nature of stages 3 and 4.

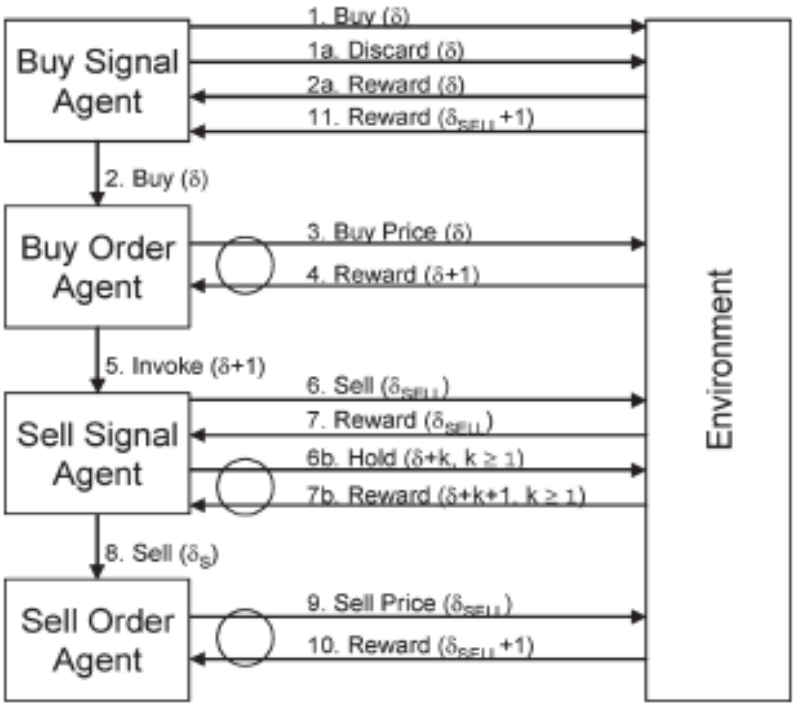

Figure. 4 multi agent Q learning Framework

Depending on how close the purchasing agent's price (BP) is to the optimum value—that is, the lowest BP necessary for a successful purchase on day δ + 1—the environment rewards the buying agent. After buying the stock on day δ + 1, the sell signal agent looks at the stock's recent price history and calculates the current profit or loss. On each day after that, starting from δ + 1, the agent decides whether to hold onto the stock or make a sell based on this evaluation. As stated in step 6b of Figure 1, the environment provides the sell signal agent with an updated state and a reward when it decides to keep the stock on a day δ + k, where k is higher than or equal to 1. This allows the agent to repeat the same process the next day. On day δSELL + 1, which is the day the sell signal agent decides to sell the stock, the sell order agent will get the same rewards as the buy order agent if the stock is sold at the SP (selling price) that they decided. The environment notifies the buy signal agent of the profit rate attained as a reward, and that's how each episode ends.

5.2 Multiagent Q – Learning algorithm:

Multi-Agent Q-learning, involves multiple agents interacting in an environment, learning and making decisions cooperatively or competitively. Applying multi-agent Q-learning for stock price prediction involves multiple learning agents aiming to optimize their trading strategies and collectively predict stock prices.

1. Multi-Agent Environment Setup:

Multiple Agents:

- N agents in the environment, each indexed by i=1,2,...,N.
- Each agent represents an independent learner making decisions based on observed states and interactions.

Interaction:

- Agents interact with the environment, which simulates historical stock market data.
- At time *t*, Agent *i* observes state $S_t^i$and takes action A_t^i, receiving reward $R_t^i$from the environment.

2. States, Actions, and Rewards:

States:

- State S_t^i derived from historical stock market data, comprising technical indicators, market sentiment, etc.
- S_t^i represents the observed features at time *t* for Agent *i*.

Actions:

- Action A_t^i available to each agent, such as buying, selling, or holding stocks based on observed states and individual strategies.

Rewards:

- Individual reward systems for each agent, R_t^i, reinforcing profitable actions and penalizing losses.
- Encourages agents to improve strategies based on their own experiences.

3. Multi-Agent Q-Learning Framework:

Q-Value Update:

- Each agent maintains a Q-table, updating Q-values based on experiences in the environment.
- Q-value update for an agent *i* at time *t* after taking action A_t^i in state S_t^i:

$$Q_t^i(S_t^i, A_t^i) = (1-\alpha).Q_{t-1}^i(S_t^i, A_t^i) + \alpha.(R_t^i + \gamma.max_{a'}.Q_{t-1}^i(S_{t+1.a'}^i)$$

- $Q_t^i(S_t^i, A_t^i)$: Updated Q-value for state-action pair of Agent *i* at time *t*.
- α: Learning rate determining the impact of new information.
- γ: Discount factor balancing immediate and future rewards.

Cooperation or Competition:

- Agents can learn cooperatively (sharing strategies) or competitively (striving to outperform others).

Q-Table Structure:

- Each agent's Q-table stores Q-values for state-action pairs, guiding their decision-making.
- $Q_t^i(S, A)$ represents the Q-value for state *S* and action *A* of Agent *i* at time *t*.

5.3 Workflow for Multi-Agent Q-learning in Stock Price Prediction:

Initialization:

1. **Initialize Agents:**
    - Create *N* independent agents, indexed as *i*=1,2,...,*N*.
    - Initialize each agent's Q-table Q_i with random or zero values.
    - Set hyperparameters: learning rate (α), discount factor (γ), exploration rate (ϵ), and other relevant parameters.

Interaction Loop (Per Time Step *t*):

2. **Observe States:**
    - Agents observe the current state S_{ti} from the stock market environment based on historical data.
3. **Choose Actions:**
    - For each agent *i*:
        - Based on its observed state S_{ti}, select an action A_{ti}using an exploration-exploitation strategy (e.g., epsilon-greedy).
4. **Execute Actions:**
    - Agents execute their chosen actions A_{ti} in the environment.
    - Receive individual rewards R_{ti}based on their actions' outcomes.
5. **Update Q-Values:**
    - For each agent *i*:
        - Update Q-values using the Q-learning update equation:

$$Q_t^i(S_t^i, A_t^i) = (1-\alpha).Q_{t-1}^i(S_t^i, A_t^i) + \alpha.(R_t^i + \gamma.max_{a'}.Q_{t-1}^i(S_{t+1.a'}^i)$$

            - Update the Q-value for the observed state-action pair based on received rewards and future estimations.
            - α: Learning rate controlling the impact of new information.
            - γ: Discount factor balancing immediate and future rewards.
6. **Cooperative/Competitive Learning:**
    - Agents update their strategies and learning based on their individual rewards and the Q-value updates.
    - They learn cooperatively (sharing strategies) or competitively (aiming to outperform others).
7. **Iteration:**
    - Repeat steps 2-8 for multiple time steps or episodes, allowing agents to refine their strategies and Q-values.
8. **Termination:**
    - Terminate the learning process after a predefined number of iterations or convergence criteria are met

VI. 6. EXPERIMENTS

6.1 Datasets

We use the daily stock price data of over 7,000 stocks situated in the United States, gathered up to November 10, 2017. We consistently use the time frame from January 1, 2017, to November 10, 2017, for testing purposes for each stock. The training set, on the other hand, consists of data from January 1, 2015, to December 31, 2016. Therefore, there are a total of 504 samples available for training and 218 samples available for testing. The sample size is very less in comparison to a well-recognised supervised learning issue like ImageNet [20], which

consists of one million annotated pictures. However, as shown in Section 5.2, we are still able to produce profitable methods. Figure 4 shows the stock price of Google.

Figure 5 The stock price of Google from 01-Jan-2015 to 10-November-2017

- We assessed the back test results using six performance measuring metrics:

• Cumulative return (CR): the entire amount that an RL agent makes over the course of the trading session, less transaction expenses.

- Maximum drawdown (MDD): the greatest percentage of loss experienced during a trading session from peak to trough.

- Sharp ratio (SR): When trading techniques are faced with a unit of risk, the Sharp ratio is used to calculate the rate of return that they can accomplish. It is the most widely used mainstream standardised statistic for assessing the success of portfolio strategies.

• Calmar ratio (CMR): The MDD idea is utilised to calculate the risk using the Calmar ratio.

• Alpha: The extra return that the model achieves in comparison to the benchmark within the trading range is measured by the alpha value. In relation to the benchmark, the higher the alpha value, the more potential for further gains. The method used to calculate the alpha value is shown in:

$$\text{Alpha}=Rp-[Rf+\beta p(Rm-Rf)]$$

where Rp is the yield of the model, βp is the model's beta value, and c is the benchmark strategy's yield.

• Beta: Indicator used to evaluate the model's systemic risk in comparison to the benchmark is the beta value. When the beta value is higher than one, the model's volatility exceeds that of the benchmark; when the beta value is lower than one, the model's volatility falls short of the benchmark; and when the beta value is exactly one, the model's volatility is identical to the benchmark's. Below is an illustration of the computing procedure.

$$\text{Beta}=\text{Cov}(Rp,Rm)/\sigma 2m$$

where CovCov is the covariance, and $\sigma 2m$is the variance of the benchmark strategy.

SR stands for risk-adjusted return, which may take performance, comprehensive income, and strategy risk into account. CR and alpha values are used to assess the profitability of the RL agent independently; MDD and beta values are used to gauge the capacity of risk management.

Table.2 Simulation parameters

Parameter	**Value**
Trading window size (s)	20
Risk-free rate of return (rf)	
A2C learning rate	0.0007
A2C n_steps	5

Table. 3 Performance analysis of existing method and proposed method

Method	**CR (%)**	**MDD (%)**	**SR**	**CMR**	**Alpha**	**Beta**
Proposed Model	88.5	-21.7	1.37	1.71	0.28	0.69

SVM	67.7	-33.9	0.99	0.87	0.16	1.01
LSTM-AE	62.3	-36.3	0.91	0.75	0.14	1.06
PCA&DWT	36.3	-33.1	0.71	0.05	0.05	0.94
RNN	49.1	-29.2	0.89	0.76	0.10	0.90

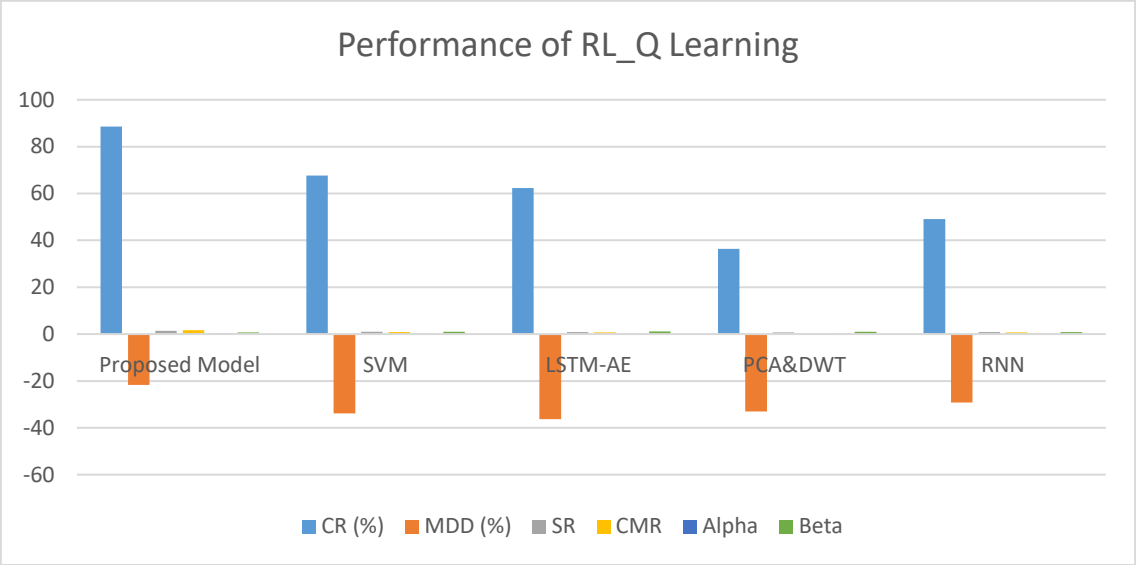

Figure 6 Performance analysis of existing method and proposed method

The Fig. 6 presents the performance metrics of the RL-Q learning method compared to some benchmarks or existing methods. The RL-Q learning method exhibits promising results across various indicators. It achieved a notable cumulative return (CR) of 88.5%, indicating substantial profitability over the trading period. Despite fluctuations, the method experienced a relatively moderate maximum drawdown (MDD) of -21.7%, suggesting resilience during market downturns. Moreover, the method demonstrates a strong risk-adjusted return, as evidenced by its Sharpe Ratio (SR) of 1.37, surpassing many existing strategies. Additionally, the cumulative market return (CMR) of 1.71 signifies its outperformance relative to the market benchmark. Furthermore, the RL-Q learning method exhibits positive alpha, measuring its excess return compared to the market index. With an alpha of 0.28, the method indicates consistent outperformance against the benchmark. Moreover, its beta coefficient of 0.69 suggests lower volatility compared to the overall market, implying potentially more stable returns. Overall, the RL-Q learning method emerges as a robust approach, offering significant returns, prudent risk management, and favorable risk-adjusted performance. These results underscore its potential as a viable strategy for investment and trading applications.

a. CONCLUSION:

This paper presents a reinforcement learning framework, especially using Q-learning algorithms, for the purpose of predicting stock market trends. We used cooperative multi-agent systems inside the Q-learning framework to improve trading performance and make precise predictions of stock prices. By using meticulous preparation approaches such as Z-score normalisation, our objective was to enhance the input data, making it more suitable for the Q-learning model. We explored the core principles of reinforcement learning, specifically focusing on the complexities of multi-agent Q-learning in the context of predicting stock prices. This suggested model was shown effective via real-world trials utilising authentic stock market data, clearly demonstrating its better performance in comparison to standard approaches such as Support Vector Machine (SVM). However, despite the positive results of our research, the use of reinforcement learning in actual trading situations faces difficulties with the accuracy of data, the intricacy of models, and the capacity to adjust in real-time. This study adds to the current investigation of reinforcement learning methods in financial markets, offering a viable approach to enhance trading tactics and enhance stock price forecasts.

REFERENCES:

[1] Gandhmal, D. P., & Kumar, K. (2019). Systematic analysis and review of stock market prediction techniques. *Computer Science Review*, *34*, 100190.

[2] Selvamuthu, D., Kumar, V., & Mishra, A. (2019). Indian stock market prediction using artificial neural networks on tick data. *Financial Innovation*, *5*(1), 1-12.

[3] Meng, T. L., & Khushi, M. (2019). Reinforcement learning in financial markets. *Data*, *4*(3), 110.

[4] Kumbure, M. M., Lohrmann, C., Luukka, P., & Porras, J. (2022). Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications*, *197*, 116659.

[5] Busoniu, L., Babuska, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *38*(2), 156-172.

[6] Carta, S., Ferreira, A., Podda, A. S., Recupero, D. R., & Sanna, A. (2021). Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting. *Expert systems with applications*, *164*, 113820.

[7] Carta, S., Ferreira, A., Podda, A. S., Recupero, D. R., & Sanna, A. (2021). Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting. *Expert systems with applications*, *164*, 113820.

[8] Almahdi, S., & Yang, S. Y. (2017). An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems with Applications*, *87*, 267-279.

[9] Hirchoua, B., Ouhbi, B., & Frikh, B. (2021). Deep reinforcement learning based trading agents: Risk curiosity driven learning for financial rules-based policy. *Expert Systems with Applications*, *170*, 114553.

[10] Shin, W., Bu, S. J., & Cho, S. B. (2019). Automatic financial trading agent for low-risk portfolio management using deep reinforcement learning. *arXiv preprint arXiv:1909.03278*.

[11] Yang, H., Liu, X. Y., Zhong, S., & Walid, A. (2020, October). Deep reinforcement learning for automated stock trading: An ensemble strategy. In *Proceedings of the first ACM international conference on AI in finance* (pp. 1-8).

[12] Dempster, M. A., & Leemans, V. (2006). An automated FX trading system using adaptive reinforcement learning. *Expert systems with applications*, *30*(3), 543-552.

[13] Li, Y., Liu, P., & Wang, Z. (2022). Stock trading strategies based on deep reinforcement learning. *Scientific Programming*, *2022*.

[14] Lin, Y. C., Chen, C. T., Sang, C. Y., & Huang, S. H. (2022). Multiagent-based deep reinforcement learning for risk-shifting portfolio management. *Applied Soft Computing*, *123*, 108894.

[15] Yu, P., Lee, J. S., Kulyatin, I., Shi, Z., & Dasgupta, S. (2019). Model-based deep reinforcement learning for dynamic portfolio optimization. *arXiv preprint arXiv:1901.08740*.

[16] Aboussalah, A. M., & Lee, C. G. (2020). Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization. *Expert Systems with Applications*, *140*, 112891.

[17] Wu, X., Chen, H., Wang, J., Troiano, L., Loia, V., & Fujita, H. (2020). Adaptive stock trading strategies with deep reinforcement learning methods. *Information Sciences*, *538*, 142-158.