Optimizing Stock Portfolio Management Using a Dueling Deep Q-Network

Indrajith M P
School of Computer Science and Engineering
(Lovely Professional University)
Phagwara, India
indrajithmp77@gmail.com

Anand Kumar
School of Computer Science and Engineering
(Lovely Professional University)
Phagwara, India
anandkumar@gmail.com

***Abstract*— Reinforcement learning (RL), which is seen as beneficial, for refining strategies in changing market environments has garnered interest in the context of managing stock portfolios in recent times. A new approach outlined in this study involves applying a Dueling Deep Q Network (DDQN) utilizing metrics like Relative Strength Index (RSI) and Moving Average Convergence Divergence (MAC). This innovative method aims to address challenges linked to unpredictability and insecurity, within conventional financial market frameworks. The results, from the experiment show that this method works well showing a Sharpe Ratio for risk adjusted returns compared to models discussed in research papers. Using stock data for portfolio management in a trading setting has led to improved performance, by the DQN agent. This research strives to enhance the automated trading systems sector by establishing a base that considers the dynamics of markets.**

***Keywords*—Automated trading, Sharpe ratio, Moving average, RSI, MACD**

I. INTRODUCTION

Artificial intelligence has totally changed how stock trading and portfolio management work, making it possible for decisions to be automated and optimized in real-time. The traditional approach to stock trading usually relies on technical analysis and manual strategies, which can be pretty time-consuming and even a bit subjective. But over the past few years, machine learning and deep learning algorithms have started being used widely to develop models that can predict and classify movements in financial markets [3]. Thanks to advances in AI, especially in the area of Reinforcement Learning (RL), we can now create agents that are capable of making detailed financial decisions on their own [8]. RL is a natural fit for fast-changing markets because it gives agents a way to learn the best strategies through ongoing interactions with the environment around them. So, if the goal is to design a system that maximizes profit in a stock market that's constantly shifting, it makes a lot of sense to consider a reinforcement learning approach [1]. Among the different RL techniques, the Deep Q-Network (DQN) has shown to be particularly effective in settings where the action choices are more limited.

Even though Deep Q-Networks (DQNs) have shown to be useful in a variety of applications, financial markets bring their own set of challenges, like high volatility, price instability, and the need to handle delayed rewards [2]. To tackle these issues, more refined reinforcement learning architectures are needed—ones that can improve both stability and decision accuracy. This study uses a Dueling DQN, which is an extension of the standard DQN. It introduces two separate streams: one for estimating the overall value of a given state and another for gauging the benefit of each possible action [6]. In highly fluctuating environments like financial markets, this setup allows for more precise learning by separating state evaluation from action evaluation. By focusing on the "advantage" part, Dueling DQNs are able to develop more stable policies, making decision accuracy less vulnerable to sudden market swings. The system also uses key indicators, such as moving averages, the Relative Strength Index (RSI), and Moving Average Convergence Divergence (MACD), to capture market patterns and momentum. Instead of relying on absolute prices at a fixed future point, it's often better to use relative price changes from an earlier time as target values when predicting stock prices [1].

Metrics like the Sharpe ratio are used to measure risk-adjusted returns [3], an important consideration in finance, where aiming for high returns in a volatile environment isn't always ideal. This study also includes stabilization techniques, like gradient clipping, to help control the risk of large gradient values during training. The results highlight the potential of Duelling DQN for optimizing trading strategies in unpredictable markets. With these enhancements, it outperforms baseline methods in automating buy, sell, or hold decisions.

II. RELATED WORKS

Paper [1] investigates the use of deep reinforcement learning (DRL) to optimize trading strategies in financial markets by introducing a framework called MQ-Trader which combines multiple agents to improve decision-making for buy/sell signals and order executions. In order to dynamically handle actions based on stock market conditions, the study makes use of a multi-agent Q-learning model that has been adapted with a deep Q-network. Preprocessing techniques such as Z-score normalization further enhance the model.

Paper [2] thoroughly examines Deep Reinforcement Learning (DRL) applications in quantitative algorithmic trading and emphasizes the challenges of trading in automated, modern financial markets. The review highlights the main issues DRL faces, including the exploration/exploitation dilemma, limited observability of market conditions, and data availability and quality, all of which, if not handled appropriately, can result in major transaction costs. Pricope evaluates the shortcomings of different models in terms of reward functions and feature representation, classifying current approaches into critic-only, actor-only, and actor-critic methods. According to the results, certain DRL models have been shown to perform better than conventional trading methods in chaotic market environments, like the stock market crash in March 2020.

Paper [3] uses Deep Reinforcement Learning (DRL) techniques to suggest an ensemble strategy for stock trading. They focused on the benefits of actor-critic techniques over conventional critic-only techniques and their applicability in adjusting to the unpredictable nature of financial markets. The study describes a trading environment that imitates actual circumstances and models stock trading as a Markov Decision Process (MDP). By choosing the top-performing agent based on the Sharpe ratio, the ensemble approach improves trading performance through the use of three distinct actor-critic algorithms: Proximal Policy Optimization (PPO), Advantage Actor-Critic (A2C), and Deep Deterministic Policy Gradient (DDPG).

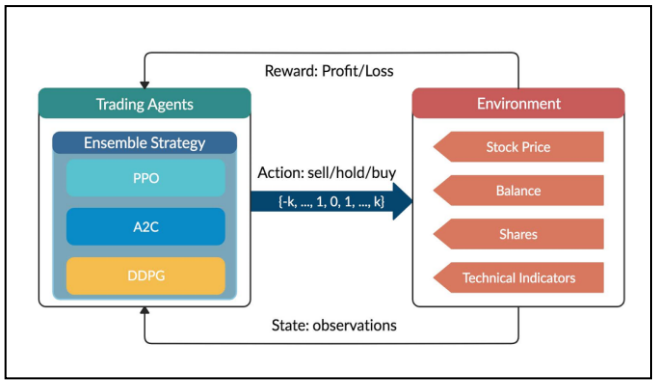

Fig 1. Ensemble Architecture for Automated Stock Trading. [3]

The application of deep reinforcement learning (DRL) in automated stock trading is examined by the authors of papers [4] and [5], with a focus on adaptive systems that are suited to individual risk profiles and market dynamics. The necessity of adaptive trading systems to maximize investor strategies is highlighted in Paper [4], which uses Deep Deterministic Policy Gradient (DDPG) for swing trading as a Markov Decision Process (MDP) and incorporates Recurrent Convolutional Neural Networks (RCNN) for sentiment analysis. In contrast, a risk-adaptive trading system is presented in paper [5] that modifies strategies according to each person's unique risk profile as determined by the Balloon Analogue Risk Task (BART).

Deep reinforcement learning has come a long way in stock trading, yet some challenges still remain, like managing transaction costs, enhancing risk-adjusted returns, and adapting to unstable markets. By focusing on these shortcomings, this study introduces a Dueling Deep Q-Network model to improve existing approaches, aiming to strengthen decision-making ability and improve portfolio performance in a constantly changing financial market.

III. Methodology

The methodology employed in this research is a Dueling Deep Q-Network (DQN) framework, which aims to tackle the problems of stock price prediction and create an optimized trading strategy. The method incorporates modern techniques from the deep learning and reinforcement learning domains in recognition of the obstacles of financial time series data. This comprises several interconnected steps, including data acquisition, feature engineering, model architecture design, training processes, evaluation metrics, and simulation of trading strategies. Every component is carefully arranged to offer a solid analysis, with the ultimate goal of producing accurate predictions and assisting in making wise trading choices.

A. Data Acquisition

The research framework relies on a systematic data collection process using the Alpha Vantage API, which gives broad access to financial market data. Over the course of a year, this systematic approach collects daily trading data for a variety of important market assets. In this way, it records various market cycles, seasonal patterns, and fluctuating market circumstances.

Key data points collected for each security include:

- Daily opening and closing prices
- Intraday high and low values
- Trading volume metrics
- Date-specific timestamps

These key data points provide us with a deep insight into how the market behaves and serve as the foundation for our analysis. We chose a one-year period on purpose to improve our chances of spotting trends, striking a balance between current market conditions and enough historical context. Users can enter the stock symbol they're interested in to gather relevant data for that specific stock. The data we collect is organized neatly in a time series format, with each entry showing different market variables. To keep things consistent over time, something that's really important for time series analysis, we make sure the data points are arranged in chronological order.

B. Data Preprocessing

Before starting the analysis, it's crucial to tackle the irregularities and inconsistencies that usually come with raw financial data. To maintain data consistency and quality, this research applies a multi-step preprocessing approach. This includes Missing Value Treatment, Outlier Detection and Treatment and Technical Indicator Standardization.

- Missing Value Treatment
  The study employs advanced techniques to handle missing values. For technical indicators, we use linear interpolation, while continuous price data is managed through a forward-fill method [10]. For volume data, we apply moving average imputation, and we have a special approach for dates tied to corporate actions.

- Outlier Detection and Treatment
  Strong outlier detection techniques are used in the study, such as statistical techniques that identify abnormal price movements, use rolling z-scores, and detect and analyze volume spikes. To reduce distortion while maintaining real market movements, extreme value cases are handled with care.

- Technical Indicator Standardization
  The research guarantees consistency and comparability across data by applying specific standardization techniques to technical indicators [9]. To simplify cross-security comparison, moving averages are scaled in relation to current price levels, momentum indicators are standardized, RSI values are maintained within the natural 0-100 range, and MACD components are normalized against past volatility.

C. *Feature Engineering*

In order to capture market dynamics and give the model useful inputs, feature engineering is essential. To produce an extensive feature set, the method combines current quantitative techniques with traditional technical analysis.

To capture various aspects of market behaviour, we implement key technical indicators such as

- Moving Average Convergence Divergence (MACD): The 12-day and 26-day exponential moving averages are used to calculate MACD, showing changes in momentum and trends [7].
- Relative Strength Index (RSI): The RSI, which is calculated over a 14-day period, evaluates price momentum and suggests probable overbought or oversold situations.
- Moving Averages: Trends over a range of time horizons are captured by short-term (5-day) and medium-term (20-day) moving averages, which also form the foundation for crossover signals that signal shifts in trends.
- Volatility Measures: Rolling standard deviations of returns, which indicate risk and market unpredictability, offer insights into volatility.

By integrating trends, momentum, and risk metrics, these feature engineering techniques allow the model to interpret complex market behaviours with ease [4].

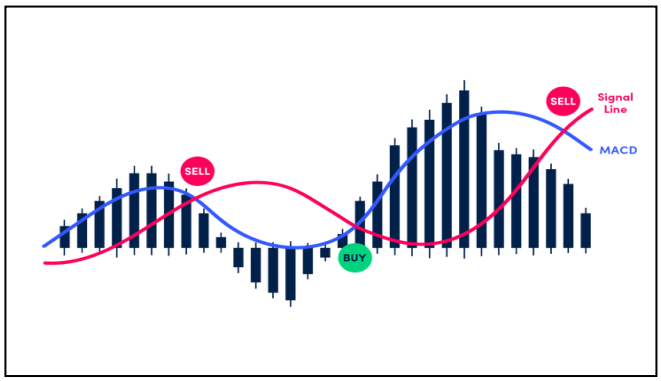

Fig 2. Visual representation of a Moving Average Convergence Divergence (MACD) indicator on a price chart. [7]

Ten carefully chosen market indicators, each chosen for its distinct contribution to market analysis, make up the final feature set. Together, these features offer a thorough understanding of market conditions by containing elements of trend, momentum, fluctuation, and sentiment. This deep learning model can capture complicated market dynamics because of to the multi-dimensional representation of the market state created by integrating these features.

D. *Dueling Deep Q-Network Design*

The implementation of an advanced Dueling Deep Q-Network (DDQN) architecture, specifically built to handle the complexities involved in stock portfolio management, forms the foundation of this methodology. This new architecture marks a significant breakthrough in the use of reinforcement learning in the financial industry. This method stands out due to its dual-stream design, which separates action advantages from state value estimation. This separation allows for a more sophisticated learning process, allowing the model to evaluate the fundamental value of states without regard to the particular actions performed. The network can achieve improved generalization capabilities across a range of market conditions and improved learning dynamics by disconnecting these components.

The shared feature extraction layers in the architecture form the basis for the value and advantage streams that proceed. By using noise-injected linear transformations together with layer normalization strategies, these feature extraction layers promote stable training and robust learning. The value stream is in charge of calculating the total value of existing in a particular market condition and taking into account the long-term gains connected to different portfolio arrangements.

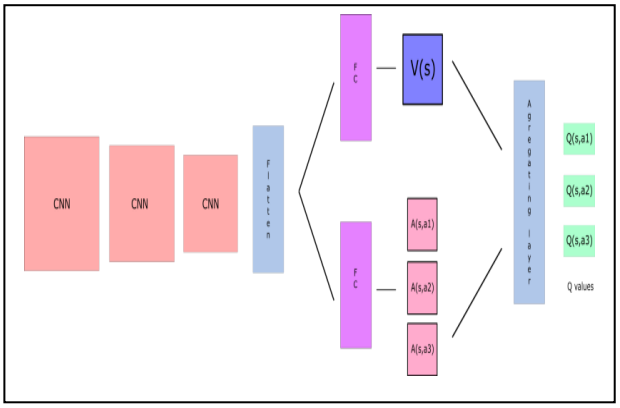

Fig 3. The Dueling Deep Q-Network (DDQN) architecture separates the estimation of the state value V(s) and action advantage A(s,a). [6]

This design brings in some unique elements to make learning process more efficient. It uses specialized linear layers that add a bit of controlled noise, making the model to explore the action space more thoroughly and avoid fixing a solution too early. Layer normalization is applied throughout which helps the network to stabilize and converge faster during training. In the fluctuating world of financial markets where volatility and uncertainty are always there, finding the right balance between stability and curiosity is integral.

The Dueling Deep Q-Network architecture provides a strong and flexible learning framework that can handle the complex problems of portfolio optimization. This methodology seeks to achieve higher accuracy in forecasting market behaviours and making well-informed investment decisions by utilizing modern deep reinforcement learning techniques.

IV. Model Training and Evaluation

The training process uses a carefully adjusted approach to model development which combines best practices from theory as well as practical factors specific to financial use. The trading strategies are more dependable and effective due to this systematic framework, which also makes sure they are prepared to handle the complex dynamics of financial markets.

A. Training Implementation

With a batch size of 32 samples, mini-batch learning is used in the training process. By selecting this choice, the model can learn from a wide range of samples while reducing the noise that comes with smaller batches, striking the ideal balance between computational efficiency and learning stability. With a learning rate of 0.0005 and a weight decay of 1e-5, we use the Adam optimizer. The model can constantly modify its parameters because of this configuration's adaptive learning capabilities, which also prevent overly complicated solutions and avoid overfitting.

The usage of gradient clipping with a threshold of 1.0 guarantees stable training. This method avoids gradient explosion, which can cause instability during training and result in poor model performance. Additionally, a Boltzmann exploration strategy with a temperature parameter of 0.5 is used to manage the exploration-exploitation trade-off. This promotes a more balanced and efficient learning process by enabling the model to experiment with new methods while taking advantage of previously learned patterns. The smooth L1 Loss function is specifically chosen for its robustness to outliers that are common in financial data. By striking a balance between L1 and L2 losses, this loss function improves training stability while maintaining sensitivity to smaller errors.

B. Performance Implementation

In order to make sure that the strategies created are not only successful but also practical for use in the real world, the stock portfolio system evaluation uses an extensive framework that evaluates both financial performance and operational efficiency.

A complete portfolio simulation framework is used in the study, and it begins with a ₹5,000 initial portfolio value. We can more accurately simulate trading conditions because of this realistic setup. The costs of carrying out trades in actual markets are reflected in the simulation's integration of market frictions, such as a transaction cost of 0.1% per trade. The model's output informs daily portfolio rebalancing decisions, and position sizing is tailored to efficiently manage risk exposure. By taking into account how transaction costs and market conditions affect portfolio performance, this method strengthens the robustness of our model.

A more thorough understanding of the model's effectiveness is provided by the evaluation network's thorough approach to performance assessment. We can compare strategies with varying risk profiles by using the Sharpe Ratio, which is an integral metric for assessing risk-adjusted returns. Finding possible weaknesses in the strategy is made easier with the help of maximum reduction analysis, which provides insights into downside risk management.

In summary, This method easily combines advanced machine-learning techniques with traditional financial analysis, representing an extensive approach to automated portfolio management. The study puts a solid foundation for creating and putting into practice, a systematic trading strategies that can successfully adapt to the dynamic nature of financial markets by concentrating on reliable training.

V. Results and Discussion

Thorough testing was done on a number of significant stocks from various industries, including Reliance, Apple and Infosys in order to assess the effectiveness of our Dueling Deep Q-Network (DDQN) for portfolio management. A ₹5,000 initial portfolio value was used for each test, and the model was trained to optimize returns while controlling risk in these markets that were volatile. The outcomes show how well the model performs in different market conditions and demonstrate how flexible it is in reaching the best possible investment choices. Looking ahead, this model could be improved by integrating NLP for news sentiment analysis, enabling it to offer real-time inter day trading suggestions.

TABLE I. Performance Metrics of Dueling Deep Q-Network PPO and DDQN model

Company Name	FINAL PORFOLIO VALUE		
	DDQN	***PPO***	***DDPG***
AMAZON	₹9733.23	₹5384.34	₹9987.43
APPLE	₹9803.56	₹4677.12	₹5004.56
INFOSYS	₹9908.20	₹5157.40	₹5,000.00

VI. References

[1] Bao, M. (2024b). Deep Reinforcement Learning Based Optimization and Risk Control of Trading Strategies. Deleted Journal, 20(5s), 241–252. https://doi.org/10.52783/jes.1943.

[2] Pricope, T. (2021, May 31). *Deep Reinforcement Learning in Quantitative Algorithmic Trading: A Review*. arXiv.org. https://arxiv.org/abs/2106.00123.

[3] Yang, H., Liu, X., Zhong, S., & Walid, A. (2020). Deep Reinforcement Learning for Automated Stock Trading: An Ensemble Strategy. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3690996.

[4] Azhikodan, A. R., Bhat, A. G. K., & Jadhav, M. V. (2018). Stock Trading Bot Using Deep Reinforcement Learning. In Lecture notes in networks and systems (pp. 41–49). https://doi.org/10.1007/978-981-10-8201-6_5.

[5] Quek, C., & Cao, Q. (2022). Risk-Based Adaptive Stock Trading System Using Reinforcement Learning. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.423736.

[6] Jain, A. (n.d.). Deep Q Networks to perform reinforcement learning in continuous state enviorments | Blog by Aditya Jain. Aditya Jain | Portfolio. https://adityajain.me/blogs/deep-q-learning.html.

[7] Moving Average Convergence Divergence (MACD) | Learn to Trade |OANDA. (n.d.). OANDA.https://www.oanda.com/us-en/learn/indicators-oscillators/dete rmining-entry-and-exit-points-with-macd/.

[8] *Deep Reinforcement Learning: A brief survey*. (2017, November 1). IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/document/8103164

[9] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

[10] The MIT Press, Massachusetts Institute of Technology. (2024, June 18). *Book Details - MIT Press*. MIT Press. https://mitpress.mit.edu/9780262035613/deep-learning/.