Comparison of Deep Reinforcement Learning Algorithms for Trading Strategy

Xiaoyang Jiang

Department of Computer Science, University of Liverpool, Liverpool, L69 7ZX, United Kingdom
sgxjia19@liverpool.ac.uk

Abstract. A stock trading strategy refers to a structured approach used to make informed decisions in buying, selling, or holding stocks in financial markets. These strategies play a crucial role in maximizing returns while managing risk. Over the years, trading strategies have transitioned from being expert-driven, time-intensive approaches to incorporating machine learning algorithms that process vast amounts of historical data. Stock trading strategies have evolved from expert-driven approaches to incorporating machine learning algorithms and, more recently, artificial intelligence and deep learning techniques. This paper delves into the utilization of deep reinforcement learning in trade. It presents an overview of Deep Reinforcement Learning (DRL) principles and their relevance to trading, followed by an exploration of five specific machine-learning models employed in trading strategies. Each model is detailed in terms of its characteristics, principles, advantages, and limitations. Additionally, this paper discusses evaluation metrics and provides a brief insight into potential result disparities within the same stock. The discussion section analyzes the strengths and weaknesses of the presented models and highlights their potential. The conclusion summarizes the methods employed and the results observed and suggests avenues for future research and development in utilizing DRL for trading strategies.

Keywords: Reinforcement learning, Deep learning, Trading strategy.

1 Introduction

Due to its potential to improve trading tactics in dynamic and complicated market conditions, using DRL in the trading space has attracted a lot of interest. Historically, experts with a deep understanding of market dynamics often developed trading strategies. These strategies were characterized by their time-intensive nature and reliance on human decision-making. As technology advanced, machine learning algorithms emerged as tools to process vast amounts of historical market data and identify patterns that might be imperceptible to human traders. Algorithms like Moving Averages, Support Vector Machines (SVMs), and Decision Trees were used to create rule-based strategies that adapt to historical trends [1].

The integration of machine learning techniques into trading strategies marked a significant shift in the field. These techniques include linear regression, time series

B. H. Ahmad (ed.), *Proceedings of the 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*, Advances in Intelligent Systems Research 180,
https://doi.org/10.2991/978-94-6463-370-2_2

analysis, and clustering methods. SVMs have been utilized to predict price movements and identify trading signals. However, these techniques often face challenges in adapting to changing market conditions and capturing intricate patterns in highly complex environments. The limitations of traditional machine learning algorithms paved the way for the adoption of DRL in trading. DRL combines the principles of reinforcement learning with deep neural networks, enabling algorithms to learn from experience and make adaptive decisions. DRL's ability to process high-dimensional data, capture temporal dependencies, and learn complex non-linear relationships aligns well with the dynamic nature of financial markets.

Recent literature has explored various aspects of DRL applications in trading, including algorithmic trading, portfolio optimization, and risk management. Notable works include the application of Deep Q-Networks (DQN) for building autonomous trading agents that learn optimal strategies through trial and error. Proximal Policy Optimization (PPO) algorithms have been used to optimize trading policies by directly interacting with the market environment [2].

Incorporating technical indicators like Long, Short, Sign (R), and Moving Average Convergence Divergence (MACD) into DRL-based trading strategies has also gained attention. Long and Short indicators are used to capture long-term and short-term trends in market statistics. Sign (R) indicator reflects the direction of price changes, aiding in identifying potential buy or sell signals. MACD, a popular momentum indicator, has been integrated into DRL frameworks to improve decision-making based on the convergence and divergence of moving averages [3].

While DRL holds promise, researchers have also identified challenges, including the need for reliable and high-quality data, the trade-off between exploration and exploitation, the risk of overfitting, and the interpretability of DRL models. Researchers are actively working on addressing these challenges to ensure the practical applicability of DRL in trading.

Current trends in the literature point towards hybrid approaches that combine DRL techniques with traditional trading strategies to mitigate risks and enhance performance. Transfer learning, where knowledge gained from one market is transferred to another, is also gaining attention. Furthermore, the development of explainable DRL models aims to enhance 2 transparency and regulatory compliance.

In summary, this review introduces the transition of trading strategies from traditional approaches to machine learning-based methods, with a focus on the emerging role of DRL. By combining reinforcement learning with deep neural networks, DRL presents a potent approach for adapting to the dynamic nature of financial markets. Notable DRL techniques like DQN and PPO have demonstrated their efficacy in trading applications. Despite challenges related to data quality and interpretability, the potential of DRL in trading is evident. This paper will explore specific DRL models, their evaluation metrics, potential result variations, as well as their strengths and limitations, thereby providing comprehensive insights into their contributions to enhancing trading strategies in the following section.

2 Methodology

DRL stands at the intersection of reinforcement learning and deep neural networks, offering a powerful framework for developing intelligent agents capable of learning optimal strategies in complex and dynamic environments. In the context of trading, DRL has emerged as a promising approach to creating adaptive trading algorithms that can dynamically adjust to changing market conditions [4].

An agent interacts with its environment to learn a set of behaviors that maximize cumulative rewards over time under the reinforcement learning (RL) paradigm of machine learning. DRL extends RL by leveraging deep neural networks to approximate value functions or policy functions, enabling agents to handle high-dimensional and continuous state spaces often encountered in financial markets [5].

At its core, DRL involves several key components. 1) State: This represents the current market information observed by the agent. In trading, it includes factors like historical price data, trading volume, and technical indicators. 2) Action: An action is a decision made by the agent, such as buying or selling a certain quantity of stocks, based on the observed state. 3) Reward: The reward reflects immediate feedback to the agent following an action. It quantifies the goodness or badness of the action and guides the learning process. 4) Policy: The policy is a mapping from states to actions, defining the behavior of agents. It encapsulates the strategy the agent uses to select actions in different states. 5) Value Function: This estimates a given state's expected cumulative future rewards. It helps the agent assess the potential long-term benefits of being in a particular state. 6) Q-Function: The Q-function approximates the expected cumulative future rewards by taking a certain action in a given state. It aids in evaluating the potential outcomes of different actions [6].

DRL algorithms, such as DQN and Proximal Policy Optimization (PPO), employ deep neural networks to estimate either the value function or policy. These networks are iteratively updated using interactions with the market environment to improve accuracy. The incorporation of deep neural networks empowers DRL agents to capture intricate patterns in historical data and make informed trading decisions.

2.1 Deep Q-learning Networks (DQN)

A potent reinforcement learning technique called Deep Q-learning Networks (DQN) applies neural networks to estimate the Q function (state-action value function) [7]. Q function calculates the value of taking a certain action in a specific state, which reflects how beneficial the action is to the agent. DQN uses a neural network to learn and represent the Q values connected to various state-action combinations by parameterizing the Q function.

The fundamental objective of DQN is to minimize the mean squared error between the target Q values calculated by the neural network [8]. The target Q values are derived from the Bellman equation, which describes the optimal value of a state-activity combination based on the expected reward and the maximum Q value of the subsequent state and action:

$$M(\theta) = E[(Q_\theta(S, A) - Q'_\theta(S,A))^2] \quad (1)$$

$$Q'_\theta(S_t,A_t) = x + \gamma \operatorname{argmax}_{A'} Q_\theta(S_{t+1},A_{t+1}) \quad (2)$$

$M(\theta)$ represents the objective function that minimizes during training phase. θ represents the parameters of the neural network, and the goal of this algorithm is to adjust these parameters to reduce the value of the objective function. Minimizing this objective function is the primary goal of training DQN. x in equation (2) represents for the immediate reward obtained after taking action A_t in state S_t. γ is the discount factor, representing the importance of future rewards. γ is in the range [0, 1], controlling how much we consider future rewards. A larger γ means more emphasis on future rewards, while a smaller γ means more emphasis on immediate rewards. However, training a standard DQN might be unpredictable and unstable. To address these issues, several strategies have been developed to stabilize the training process. 1) Fixed Q-targets involve using a separate network, known as the target network, to generate the target Q values. This lessens "chasing tails," a phenomenon where the Q values change significantly during training and lowers policy variances [9]. 2) Double DQN improves the estimation of target Q values by decoupling the action selection from value evaluation. It utilizes the target network to determine the action's value in the next state and the online network to pick the action with the greatest Q value [10]. 3) Dueling DQN separates Q value into two components: the advantage of each action and state value. This separation allows value stream to receive more updates, resulting in a more accurate representation of state values [11].

By incorporating these strategies, DQN becomes more stable, converges faster, and produces better trading strategies. In this work, the advancements in DQN methodologies are leveraged to enhance the training of the trading agents and improve their performance in dynamic market environments.

2.2 Policy Gradients (PG)

Policy Gradient (PG) is an RL algorithm that optimizes the policy directly to maximize cumulative rewards. By representing the policy as a neural network parameterized by θ, denoted as $\pi_\theta(A|S)$, PG learns to generate actions that yield the highest expected rewards in a given state. The PG algorithm operates by generating trajectories from the environment and computing the cumulative rewards obtained along each trajectory. A trajectory τ is a sequence of states, actions, and rewards $\tau = [S_0, A_0, R_1, S_1, ..., S_t, A_t]$. PG intends to maximize the expected cumulative rewards τ (θ) by adjusting the neural network parameters θ using gradient ascent:

$$J(\theta) = \mathrm{E}\left[\sum_{t=0}^{T-1} R_{t+1} \mid \pi_\theta\right] \quad (3)$$

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(A_t \mid S_t) G_t \quad (4)$$

In the above equations, θ is the parameter of the neural network that represents the policy. π_θ is the policy parameterized by θ, which represents the probability distribution over actions A given a state S. G_t represents the total expected cumulative reward.

Unlike Deep Q-Networks (DQN), which learns the action-value function, PG directly learns the policy. PG may provide a probability distribution over actions when working with random policies or continuous action spaces, which is especially effective [8]. The PG training procedure makes use of Monte Carlo algorithms to sample trajectories from the surrounding space. Additionally, updates are performed only at the end of each episode. This approach can lead to slow convergence during training and may result in the algorithm becoming stuck at suboptimal local maxima. To address this limitation, various enhancements and algorithms have been offered to increase the stability and speed of convergence of PG-based techniques.

2.3 Advantage Actor-Critic (A2C)

A2C is an advancement over Policy Gradient (PG) methods, addressing the slow training convergence and suboptimal local maxima issues. It introduces real-time policy updates and combines both actor and critic networks to improve the learning process. A critic network and an actor network make up the two main parts of the A2C.

Through the output of action probabilities depending on the present state, the actor-network creates the policy. On the other side, the critic network assesses the effectiveness of the selected action within a certain condition. The interaction between these two networks forms the foundation of the A2C algorithm.

A2C seeks to maximise the objective function to renew the policy network $\pi(A|S, \theta)$:

$$J(\theta) = E\,[\log\pi(A|S,\theta)\, A_{\text{adv}}(S,A)] \tag{5}$$

The advantage function denoted as $A_{\text{adv}}(S, A)$, determines the advantage of taking action A in state S compared to the average action value. It is computed as:

$$A_{\text{adv}}(S_t,A_t) = R_t+\gamma\, V(S_{t+1} \mid w) - V(S_t \mid w) \tag{6}$$

In this formular, R_t represents for the immediate reward obtained at time step t. The advantage function quantifies the difference between the immediate reward R_t and the expected value of the next state S_{t+1} minus the value of the current state S_t.

Along with updating the policy network, A2C also employs a critical network to estimate the state value function V(s|w) with parameters w. To reduce the temporal difference (TD) error, the critic network, which contributes to the calculation of the advantage function, can be updated as:

$$J(w) = (R_t + \gamma\, V(S_{t+1} \mid w) - V(S_t \mid w))^2 \tag{7}$$

A2C offers several advantages, particularly when handling continuous action spaces. The inclusion of the advantage function helps reduce policy variance, contributing to more stable learning. Moreover, the policy is updated in real-time, leading to faster convergence [8].

A2C training can be carried out synchronously or asynchronously, as seen in the Asynchronous Advantage Actor-Critic (A3C) algorithm. This work adopts a synchronous approach, where multiple agents operate in parallel in distinct environments. This

parallel execution facilitates faster training and enhances the exploration of diverse market scenarios.

2.4 Proximal Policy Optimization (PPO)

PPO, a sophisticated actor-critic deep reinforcement learning algorithm which has gained prominence for addressing challenges associated with training RL agents in complex environments. PPO operates on the foundation of the actor-critic architecture, aiming to concurrently train the actor and the critic models.

The actor in PPO learns the policy, determining the agent's response in specified states. The critic evaluates the effectiveness of a selected action in a particular state. PPO improves upon traditional policy gradient methods by introducing a clipped surrogate objective that limits the policy update to a certain threshold. This limitation prevents drastic policy changes that can lead to instability during training.

PPO's core contribution is its proximal optimization approach, which maintains policy updates within a "trusted region" around the current policy. This region restricts policy updates to prevent the agent from diverging too far from the existing policy, addressing the instability issues associated with RL training.

PPO has been applied successfully to complex environments, such as trading, as evidenced by its effective use in ensemble strategies [12]. It provides a balanced compromise between policy stability and exploration, making it a robust choice for training trading agents.

2.5 Generalized Distributional Policy Gradient (GDPG)

Generalized Distributional Policy Gradient (GDPG) is another notable algorithm that extends the actor-critic framework to achieve enhanced performance in trading scenarios. GDPG aims to strike a balance between risk-adjusted returns and stability, an essential characteristic for successful trading agents.

The foundation of GDPG lies in the combination of the Q-Network from the Generalized Distributional Q-Learning (GDQN) system and a policy network. The advantages of both the policy gradient approach and Q-learning are combined in this special integration. The primary novelty of GDPG is its capacity to beat benchmarks like the Turtle trading technique by generating more steady risk-adjusted returns.

In comparison to Turtle trading strategy, which is a renowned trend-following approach, GDPG demonstrates its superiority by achieving enhanced performance while maintaining stability. The strategy's ability to combine the Q-Network with a policy network highlights the potential synergy between value-based and policy-based approaches [13].

GDPG serves as an important reference for researchers and traders seeking to explore advanced trading strategies that leverage both value estimation and policy optimization. Its performance demonstrates the efficacy of combining these approaches to navigate the complexities of financial markets.

2.6 Assessment Criteria

The evaluation of the various deep reinforcement learning models introduced in the preceding sections requires a comprehensive set of metrics to gauge their performance. These metrics encompass both the models' ability to generate returns and their capacity to manage risk. Key evaluation metrics include:

Cumulative Return: Cumulative return measures the total profit or loss generated by the trading strategy over a specific period. It provides a quantitative assessment of the model's overall profit-generating ability [14].

Sharpe Ratio (SR): The Sharpe ratio quantifies the risk-adjusted return of a trading strategy. It assesses returns generated per unit of risk undertaken, enabling a comparison of risk and return among different models [15].

Annualized Return (R%): The annualized return indicates the annual rate of return achieved by the trading strategy. This metric facilitates a standardized comparison of returns across different time frames [16].

3 Results

The presented performance results in Fig.1 and Table 1 provide valuable insights into the effectiveness of various deep reinforcement learning models in the context of trading across different asset classes. These results demonstrate notable disparities in their performance, which can be attributed to several key factors.

Firstly, the Generalized Distributional Policy Gradient (GDPG) system consistently outperforms other models, including the Turtle trading technique, which is a well-established benchmark in the field of trading performance comparison. The cumulative trade returns and risk-adjusted returns (R%) for GDPG are notably higher for most of the evaluated U.S. stocks over the three-year period from 2016 to 2019. This superiority in performance may be attributed to GDPG's ability to capture market dynamics effectively while maintaining a level of stability. Its quantitative comparison with other models, as provided in Table 1, clearly showcases its dominance.

Moreover, Deep Q-Network (DQN) also warrants attention in the context of algorithmic trading. DQN exhibits commendable performance, with competitive Sharpe Ratios (SR) and positive returns (R%) for several U.S. stocks. While not consistently outperforming GDPG, DQN presents an alternative approach that leverages Q-learning techniques to make trading decisions. This underscores the versatility of reinforcement learning models, allowing traders to choose between DQN and GDPG based on their specific objectives and risk tolerance.

On the other hand, the observed performance of the Advantage Actor-Critic (A2C) algorithm is noteworthy. A2C demonstrates strengths in capturing larger market moves without frequently changing positions. This attribute can be particularly advantageous in volatile markets, where sudden price movements can lead to substantial gains. Furthermore, A2C exhibits resilience in navigating markets characterized by mean-reverting behavior, indicating its adaptability to different market conditions.

It is essential to highlight that the observed differences in performance underscore the significance of selecting an appropriate algorithm tailored to the specific

characteristics of the trading environment and asset class. The choice between GDPG, DQN, and A2C, for instance, depends on the trader's objectives and risk tolerance. GDPG excels in providing stable and risk-adjusted returns, making it suitable for risk-averse investors. DQN and A2C offer alternative strategies, with DQN leaning towards Q-learning and A2C combining actor-critic architecture, each catering to distinct trading preferences.

Lastly, the differences in performance can be further analyzed by considering the underlying mechanisms of each model. GDPG's emphasis on risk-adjusted stability may be attributed to its distributional policy gradient approach, which allows it to make informed decisions based on a probabilistic understanding of the market. DQN's performance stems from its ability to estimate Q-values and make decisions based on long-term rewards, providing traders with an alternative method to approach trading decisions. A2C's strength in capturing large market moves could be linked to its actor-critic architecture, which combines both policy and value estimation, enabling it to exploit opportunities more effectively.

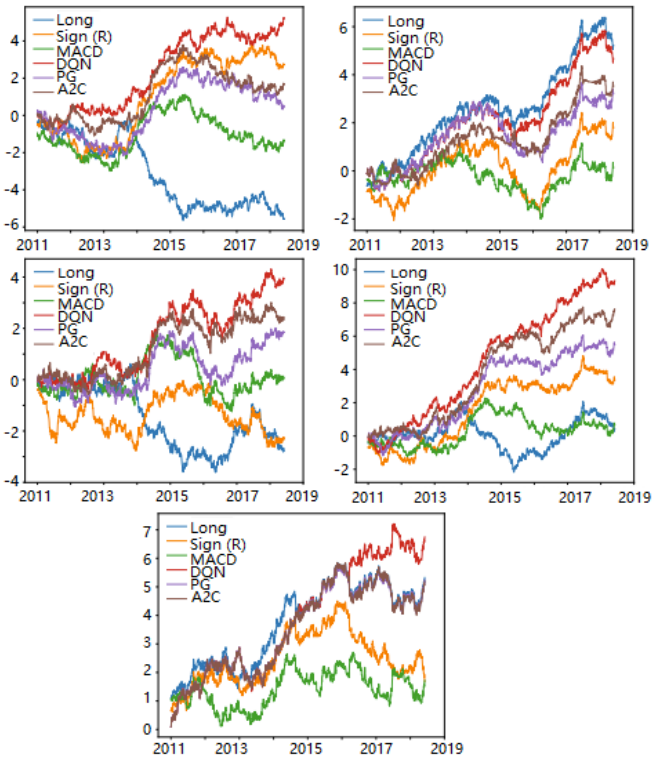

Fig. 1. Returns for commodity, equity index (first row); FX and portfolio of using all contracts (second row); fixed income (third row) [8].

Table 1. Comparisons between GDQN, GDPG and Turtle strategy in some U.S. stocks over 3 years (2016-2019) [12].

	GDQN		GDPG		Turtle	
Symbol	SR	R(%)	SR	R(%)	SR	R(%)
AAPL	1.02	77.7	1.30	82.0	1.49	69.5
GE	-0.13	-10.8	-0.22	-6.39	-0.64	-17.0
AXP	0.39	20.0	0.51	24.3	0.67	25.6
CSCO	0.31	20.6	0.57	13.6	0.12	-1.41
IBM	0.07	4.63	0.05	2.55	-0.29	-11.7

4 Conclusion

In conclusion, this paper has delved into the realm of trading strategies and explored their evolution from traditional expert-driven approaches to the incorporation of cutting-edge machine-learning techniques, culminating in the application of DRL. The integration of DRL has showcased its potential to create adaptive trading algorithms capable of navigating the dynamic and complex landscape of financial markets.

The overview of DRL's key components and principles has underscored its significance in adapting to high-dimensional and continuous state spaces, making it particularly well-suited for the intricate patterns and non-linear relationships characteristic of financial markets. The discussion of DRL methodologies such as DQN, PG, A2C, PPO, and GDPG has provided insight into their mechanisms, strengths, and potential applications in trading strategies.

The comparative analysis of these DRL models across various asset classes has demonstrated the superiority of GDPG in achieving more stable risk-adjusted returns, surpassing the performance of the renowned Turtle trading strategy. The study also highlighted the nuanced differences in performance between models when applied to a single stock, showcasing the importance of aligning model selection with specific market dynamics. However, it's important to acknowledge the challenges associated with DRL, including the need for reliable data, risk management, and interpretability of models. These challenges serve as avenues for future research and development, shaping the direction for refining DRL-based trading strategies.

This paper's contributions lie in providing a comprehensive understanding of the application of DRL in trading, presenting a clear picture of the strengths and limitations of various DRL models, and demonstrating their performance across different scenarios. The insights gained from this exploration not only shed light on the potential of DRL but also underscore the importance of tailoring strategies to the unique characteristics of the trading environment. As researchers continue to address challenges and refine these models, the future of DRL in reshaping trading strategies appears promising, holding the potential to usher in a new era of intelligent and adaptive approaches to navigating financial markets.

References

1. Pai, P.-F., and Lin, C.-S. A hybrid arima and support vector machines model in stock price forecasting. Omega 33(6), 497–505 (2005).
2. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017).
3. Chong, T. T. L., & Ng, W. K. Technical analysis and the London stock exchange: testing the MACD and RSI rules using the FT30. Applied Economics Letters, 15(14), 1111-1114 (2008).
4. Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, 34(6), 26-38 (2017).
5. Morales, E. F., & Zaragoza, J. H. An introduction to reinforcement learning. In Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions. 63-80 (2012).
6. Langlois, M., & Sloan, R. H. Reinforcement learning via approximation of the Q-function. Journal of Experimental & Theoretical Artificial Intelligence, 22(3), 219-235 (2010).
7. Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., et al. Deep q-learning from demonstrations. In Proceedings of the AAAI conference on artificial intelligence 32(1), 3223-3230 (2018).
8. Zhang, Z., Zohren, S., and Stephen, R. Deep reinforcement learning for trading. The Journal of Financial Data Science 1-16 (2020).
9. Innanen, K. A. Inversion of the seismic AVF/AVA signatures of highly attenuative targets. Geophysics, 76(1), 1-14 (2011).
10. Van Hasselt, H., Guez, A., & Silver, D. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI conference on artificial intelligence 30(1), 2094-2110 (2016).
11. Sewak, M., & Sewak, M. Deep Q Network (DQN), Double DQN, and Dueling DQN: A Step Towards General Artificial Intelligence. Deep Reinforcement Learning: Frontiers of Artificial Intelligence, 95-108 (2019).
12. Yang, H., Liu, X. Y., Zhong, S., & Walid, A. Deep reinforcement learning for automated stock trading: An ensemble strategy. In Proceedings of the first ACM international conference on AI in finance. 1-8 (2020).
13. Wu, X., Chen, H., Wang, J., Troiano, L., Loia, V., & Fujita, H. Adaptive stock trading strategies with deep reinforcement learning methods. Information Sciences, 538, 142-158 (2020).
14. Ariel, R. A. A monthly effect in stock returns. Journal of financial economics, 18(1), 161-174 (1987).
15. Bailey, D. H., & Lopez de Prado, M. The Sharpe ratio efficient frontier. Journal of Risk, 15(2), 13 (2012).
16. Fama, E. F. Stock returns, expected returns, and real activity. The journal of finance, 45(4), 1089-1108 (1990).