Proceedings of ICFTBA 2025 Symposium: Financial Framework's Role in Economics and Management of Human-Centered Development
DOI: 10.54254/2754-1169/2025.GL25764

How Do Reinforcement Learning Algorithms Optimize Trading Strategies in Financial Markets Compared to Traditional Trading Approaches? A Literature Review

Liu Hong Yuan Tom

Faculty of Engineering, The University of Hong Kong, Hong Kong, China
liutom@connect.hku.hk

Abstract: Reinforcement learning (RL) has demonstrated significant potential in optimizing sequential decision-making within financial markets' highly dynamic and uncertain environments, offering distinct advantages over traditional trading approaches. This literature review investigates the use of RL in developing and improving trading strategies by integrating the findings of ten recent studies published between 2018 and 2025, selected for their focus on RL applications in different financial domains. These studies employ a range of RL techniques, such as Q-learning, and Proximal Policy Optimization (PPO), across a variety of financial markets, including stocks, Forex, Bitcoin, and derivatives. The review shows that RL-based strategies, which often use innovations such as multi-agent systems, ensemble learning, and sentiment analysis, demonstrate superiority such as better adaptability to non-dynamic stationary market conditions, enhanced risk-adjusted returns, and capability to learn complex relationships directly from market data, thus outperforming conventional methods and market benchmarks. Challenges hindering the practical application of reinforcement learning in trading include sample efficiency, training stability, market complexity, and the necessity for accurate market assumptions. These are areas requiring further examination and enhancement.

Keywords: Reinforcement Learning, Trading Strategies, Machine Learning, Optimization

1. Introduction

Financial markets exhibit inherent dynamism characterized by volatility and uncertainty, creating challenges for conventional trading strategies. These methods rely on static models and linear assumptions, often failing to capture evolving market patterns in ever-evolving, non-stationary market conditions [1]. Conventional approaches can struggle to quickly adapt to the market changes. Since they are defined with rules, they may become suboptimal or detrimental when used in unseen scenarios. Additionally, many traditional algorithms need manual calibration, making them less effective in real-time trading environments. In this context, a more adaptive and robust approach is needed to improve the optimization of trading decisions. As a machine learning paradigm, reinforcement learning (RL) demonstrates unique potential in complex environments. Reinforcement learning frameworks formalize trading as a sequential decision-making process

DOI: 10.54254/2754-1169/2025.GL25764

where an agent learns to find the optimal actions (e.g., buy, sell, hold) by receiving rewards or penalties through interactions with the environment [2, 3]. The purpose of this literature review is to explore the application of RL in developing and optimizing financial trading strategies. This review synthesizes findings from multiple studies (2018-2025) on RL implementations and evaluation of RL algorithms; this includes Q-learning, PPO, and ensemble methods that are applied to various markets such as Forex, and Bitcoin. Through identifying innovative approaches and challenges, this review provides a foundation for future work and application of RL in the financial field.

2. Reinforcement learning for trading

2.1. Reinforcement learning framework for trading

Reinforcement learning formalizes trading problems as a Markov Decision Process (MDP), where the RL agent interacts with the market environment at discrete time steps. At each step, the agent takes in the current market state, then takes an action (e.g., buy, sell, or hold), and receives a numerical reward/penalty reflecting action outcomes [1, 2]. The goal of the agent is to maximize the cumulative expected reward by learning the optimal policy that maps states to actions.

One of the most important aspects of this framework is the state representation. An effective state representation must capture sufficient information about the market to allow well-informed decision-making. Researchers have utilized many different techniques, such as using historical price data (open, close, high, low, and volume per security) and a spectrum of technical indicators [1, 3]. A practical example is a trading agent whose state is defined by the last 60 observations of pricing data, augmented with technical indicators including Moving Average Convergence Divergence (MACD) and the Relative Strength Index (RSI) [1]. The advantage of this over traditional methods is high responsiveness to recent price changes; however, the disadvantage is false signals may induce decision noise, potentially exacerbating over-trading, which may lead to over-trading and the accumulation of transaction costs [1, 3]. While this method is powerful, careful feature selection is needed to better understand the potential market trends [3]. Another example involves using an event-driven state representation. A trading agent could use the directional change (DC) event approach, where the state only changes when the price moves by a predefined amount [4]. Other methods, such as using LSTMs to automatically learn features from time series, provide an alternative to explicit feature engineering [5].

The action space design—defining all executable agent actions—constitutes a critical component. The choice of action space must be aligned with both the objective and the selected RL algorithm. Generally, there are two types of action space: discrete action space or continuous action space. Discrete spaces comprise finite unique actions (e.g., buy/sell/hold) for directional trading [4]. This is suitable for problems where the agent only decides on the direction of a trade. Continuous action spaces allow for more detailed decisions where actions are represented by real-valued numbers. This is important for sophisticated takes such as portfolio optimization where the action may represent the exact percentage or amount of capital to allocate to an asset [2]. Properly constrained action spaces directly determine learnable strategies. The reward function is also equally important as it guides the learning process of the agent and thereby must be carefully designed to match the trading objective, such as maximizing profit, minimizing risk, or Sharpe ratio.

DOI: 10.54254/2754-1169/2025.GL25764

2.2. Key reinforcement learning algorithms and adaptations

Diverse RL algorithms have been specifically adapted for financial trading. These algorithms have been carefully chosen based on the specific nature of the trading task, such as the dimensionality of the state and action spaces (discrete vs. continuous).

Q-learning is a model-free, off-policy RL algorithm that learns the optimal action-selection policy through learning a Q-function. This function estimates the expected future cumulative reward by taking an action in the state and then following the optimal policy afterward. The agent updates its Q-values using the Bellman equation in each iteration until it converges. Q-learning allows the agent to directly learn the expected rewards, making it conceptually straightforward. Additionally, as it is an off-policy algorithm, it can learn from past experiences generated by different policies, allowing improvement in sample efficiency. However, its reliance on Q-tables becomes computationally prohibitive for high-dimensional spaces which can slow down the convergence and may not generalize well to other data. Q-learning demonstrates optimal efficacy in discrete action domains. Aloud and Alkhamees employed Q-learning to find the optimal trading rule with their DC event-driven state representation. Their Q-table was updated based on the rewards from buy/sell actions, allowing the agent to learn the optimal action for DC-defined market states [4].

Policy Gradient (PG) methods directly learn and optimize the policy function which maps states to actions or a probability distribution over actions, parameterized by θ. Through the gradient ascent method on an objective function which is typically the expected cumulative reward, the algorithm adjusts the policy parameters θ. The main idea is to increase the probability of actions that lead to higher rewards and decrease the probability of actions that lead to lower rewards. Their advantage is flexibility as the algorithm optimizes the policy directly. However, its main disadvantage is high variance may happen during training, which may lead to instability and convergence to suboptimal policies. PG methods are suitable for a wide range of problems, such as requiring continuous action space or portfolio allocation. Zhang et al. uses PG for trading future contracts. They aimed to maximize cumulative rewards by directly optimizing the policy which is represented by a neural network [1].

PPO is an advanced policy gradient algorithm known for its stability and sample efficiency. It implements a clipping function that restrains the size of policy updates at each iteration which prevents destructive large changes to the policy, on top of earlier policy gradient methods. This modification ensures more stable training. Since PPO tries to keep the new policy close to the old policy but still make progress, it effectively balances the exploration-exploitation trade-off. Liu et al. utilized PPO to construct a Bitcoin trading strategy, with an LSTM network as the basis for the policy function; they concluded that their PPO-based framework can generate a Bitcoin trading strategy that is profitatble [5]. Ding et al. improved the PPO algorithm by incorporating n-step temporal difference errors to handle sparse rewards; they found out that this modification significantly improves the hedging performance and the performance of the agent [6].

Actor-critic methods combine the advantages of both policy-based (Actor) and value-based (Critic) approaches. The “Actor” part is responsible for learning and deciding which action to take, and the “Critic” part is responsible for evaluating the action taken by the Actor by learning a value function that estimates how well the state-action is. Then, the Actor’s policy parameters are updated using the Critic’s feedback which will guide the algorithm to making better decisions. The main advantage is having a better balance between bias and variance compared to pure PG algorithms or value-based methods, which leads to more stable training. However, it can be more complex than other simpler PG methods or Q-learning. This type of algorithm is best for both discrete or

DOI: 10.54254/2754-1169/2025.GL25764

continuous action spaces in various markets. Several variations of AC methods, such as A2C, are widely used.

Advantage Actor-Critic (A2C) uses an advantage function in the critic to evaluate how much better or worse an action is compared to the average action from state s. This reduces the policy variance in policy gradient updates. Asynchronous Advantage Actor-Critic (A3C) introduced asynchronicity on top of A2C which allows the training of multiple agents in parallel in the same environment independently and updates a global set of parameters. Sarani and Rashidi-Khazaee pioneered a multi-agent A3C framework for Forex trading and Li et al. and Ye et al. include A2C as part of their ensemble model [7-9].

Deep Deterministic Policy Gradient (DDPG) is an actor-critic, model-free algorithm designed for continuous action spaces. Instead of learning a probability distribution, it learns a deterministic policy (the Actor) that maps states to specific continuous actions. The Critic learns an action-value function similar to Q-learning. DDPG extends DQN to continuous action spaces; hence it is suitable for tasks such as precise trade sizing or portfolio allocation. However, it is sensitive to hyperparameters and convergence issues, especially with complex Q-function. Liu et al. used DDPG to find the optimal stock trading strategies. Their work shows DDPG's framework and the use of target networks for stable training [2].

Soft Actor-Critic (SAC) is an off-policy actor-critic deep RL algorithm with the integration of entropy. Besides maximizing the expected cumulative reward, maximizing the entropy of the policy is one of the goals of SAC. This allows more exploration by ensuring more exploration of different actions while still satisfying the task. This helps avoid suboptimal local optima and improves robustness. Li et al. used SAC as one of the three algorithms in their ensemble strategy [7]. The introduction of SAC in their ensemble strategy aimed to utilize its strong exploration capabilities and stable learning.

Multi-Agent Deep Deterministic Policy Gradient (MADDPG) extends DDPG to multi-agent environments. Each agent has its actor and critic. Each agent's critic is trained with the observations and actions of other agents, which allows learning a centralized Q-function. However, during execution, each agent's actor only uses the observation by the corresponding agent to select actions. This method addresses the non-stationarity problem in multi-agent training where the environment changes as the agents learn, thus being able to learn complex behaviors. However, since it requires access to global stat-action information, it may not always be feasible, and scalability can be an issue with an increasing number of agents. This algorithm suits tasks designed for scenes with multiple interacting agents. Qiu et al. proposed DA-MADDPG, which is a modification of MADDPG by abstracting the actions and observations of other agents through publicly available DA (Double-Side Auction) market data for each agent's critic [10]. These algorithms are frequently adapted to the financial context not just their core mechanics but also the state representations, action spaces, and reward functions. The choice of the algorithm is caused by the requirements for the trading problem, such as whether the actions are discrete or continuous, the need for exploration, and the market environment complexity.

3. Reinforcement learning algorithms vs. traditional methods

Liu et al. demonstrated that their DDPG-based stock trading algorithm achieved higher returns than the Dow Jones Industrial Average (DJIA) and a traditional min-variance portfolio allocation strategy [2]. For instance, in backtesting with an initial portfolio value of $10,000 , their DDPG strategy scored an annualized return of and a Sharpe ratio of 1.79 , outperforming the min-variance strategy which has an annualized return and Sharpe ratio, and DJIA which has an annualized return and

DOI: 10.54254/2754-1169/2025.GL25764

Sharpe ratio. Aloud and Alkhamees demonstrated that their DCRL (Directional Change Reinforcement Learning) trading strategy achieved superior trading returns and enhanced Sharpe ratios across various stock markets when compared to Zero-Intelligence (ZI) agents and traditional DC methods, particularly with the incorporation of Q-learning [4].

Another advantage of RL algorithms is their adaptability. Unlike static rule-based strategies requiring scenario-specific optimization for specific scenarios, RL agents learn by directly interacting with the market condition [1, 4]. The agent's policies are not fixed; they are constantly updated based on the rewards or penalties they received for their actions in a market state. With optimization algorithms, this process allows the agent to dynamically tweak their policies. This allows the agent to explore other new actions and converge to the optimal policy. Since the RL agent can adapt to different market environments without explicit reprogramming or modifying the algorithm, endowing RL agents with non-stationarity robustness enables them to maintain the best performance as the market changes. A common theme in the reviewed literature is that RL-based trading strategies outperform traditional methods and market benchmarks. This is typically measured by metrics such as high cumulative returns, improved risk-adjusted returns, and greater adaptability to the ever-changing market conditions.

4. Addressing reinforcement learning challenges and innovation in trading

Despite their promises, applying the RL algorithm to financial markets can be challenging due to market complexity, non-stationarity, noises, and the high dimensionality of data. However, ongoing research is addressing these issues and developing innovative ideas.

Market complexity and non-stationary are the primary obstacles. Innovative solutions such as multi-agent systems and ensemble strategies have been developed. Multi-agent systems decompose decision-making by utilizing several specialized agents to focus on their corresponding aspect of the market, thereby adapting to the collective behavior of other participants [9, 10]. Their implementation involves specific coordination mechanisms or information-sharing protocols to deal with non-stationarity induced by concurrently learning agents. Results show that these systems are more robust than single-agent methods [9, 10]. Ensemble methods amalgamate the predictions or policies of various trained reinforcement learning agents and consolidate their outputs or select the most effective output according to established criteria. The variation is diminished by utilising the capabilities of various algorithms, hence enhancing generalisation across market regimes [7, 8]. Qiu et al. utilized a multi-agent RL (MARL) approach (DA-MADDPG) for peer-to-peer energy trading where the agents learn to work together in a dynamic auction market [10]. Their experiment shows that agents using this MARL method received more economic benefits compared to traditional approaches and that DA-MADDPG outperformed other MARL algorithms and conventional strategies such as Zero Intelligence by coordinating agents' trading activities and scoring a better balance between local demand and generation [10].

Effective feature extraction and state representation are also very important. Li et al. employed Principal Component Analysis (PCA) to lower the dimensions of stock feature vectors [7]. Ding et al. used a spatiotemporal attention-based Transformer for probabilistic financial time series forecasting as an input to their hedging agent, capturing the complex nonlinear asset relationships [6].

Sample efficiency and training stability are factors that cannot be ignored.

• Asynchronous Learning: The A3C algorithm used by Sarani and Rashidi-Khazaee improves sample efficiency and contributes to training stability through asynchronous parallel learning [9].

DOI: 10.54254/2754-1169/2025.GL25764

• Experience Replay and Target Networks: Algorithms like DDPG, used by Liu et al. and Ye et al., used experience replay and target networks to remove the correlation between samples and provide a stable target, thus improving training stability [2, 8].

• Careful Reward Engineering: Issues like sparse rewards can impact stability. Ding et al. addressed the issue of sparse rewards in derivative hedging by using Behavior Cloning with Recurrent PPO (BC-RPPO) and λ -returns to transmit reward signals more effectively [6].

• Centralized Critic in MARL: Qiu et al. proposed an extension of MADDPG that learns a centralized critic for each agent to stabilize multi-agent learning [10].

Further innovations include the integration of RL and known financial concepts. Ding et al. integrate RL with probabilistic forecasting and risk-neutral pricing concepts for derivatives hedging [6]. The challenge of unrealistic assumptions (e.g., no transaction cost, perfect liquidity) is often acknowledged, with some studies such as Zhang et al. explicitly mentioning that transaction costs are incorporated into their reward functions to train more realistically [1].

5. Conclusion

The findings demonstrate that RL algorithms offer significant advantages over conventional trading approaches. RL agents demonstrate superior adaptability in non-stationary markets, and algorithms such as Q-learning, PPO, DDPG, and A2C enable them to learn complex policies that often yield higher returns. Many innovations such as multi-agents, dynamic ensemble strategies incorporating sentiment analysis, and the integration of RL with financial models such as probabilistic forecasting are pushing the boundaries of automated trading. Despite these successes, the research also highlights several challenges and limitations. The complexity of financial markets means that RL models can be difficult to train, which may demand high-dimensional data and intensive computation. Additionally, ensuring training stability and achieving robust generalization across different markets remain the main obstacles. The design of suitable state representations, action spaces, and more importantly, reward functions that reflect the training objectives is critical. Future research should focus on enhancing the robustness and reliability of RL trading agents. This includes developing more sample-efficient algorithms, improving transfer learning capabilities across different market conditions, and creating more sophisticated frameworks within RL policies. Hybrid applications such as combining RL with other AI techniques (like NLP for sentiment analysis) or traditional financial modeling, may yield further improvements. Addressing issues of interpretability and making sure that RL strategies align with the requirements will also be important for wider adoption.

References

[1] Zhang, Z., Zohren, S., & Roberts, S. (2019). Deep reinforcement learning for trading. arXiv preprint arXiv: 1911.10107.

[2] Liu, X. Y., Xiong, Z., Zhong, S., Yang, H., & Walid, A. (2018). Practical deep reinforcement learning approach for stock trading. arXiv preprint arXiv: 1811.07522.

[3] Yasin, A. S., & Gill, P. S. (2024). Reinforcement Learning Framework for Quantitative Trading. arXiv preprint arXiv: 2411.07585.

[4] M. E. Aloud and N. Alkhamees, 2021. “Intelligent Algorithmic Trading Strategy Using Reinforcement Learning and Directional Change, ” IEEE Access, vol. 9, pp. 114659–114671. doi: 10.1109/ACCESS.2021.3105259.

[5] F. Liu, Y. Li, B. Li, J. Li, and H. Xie, 2021. “Bitcoin transaction strategy construction based on deep reinforcement learning, ” Appl Soft Comput, vol. 113, Dec, doi: 10.1016/j.asoc.2021.107952.

[6] Ding, Y., Yuan, G., Zuo, D., & Gao, T. (2025). Hedging with Sparse Reward Reinforcement Learning. arXiv preprint arXiv: 2503.04218.

DOI: 10.54254/2754-1169/2025.GL25764

[7] F. Li, Z. Wang, and P. Zhou, 2022. “Ensemble Investment Strategies Based on Reinforcement Learning, ” Sci Program, vol. 2022, doi: 10.1155/2022/7648810.
[8] Ye, A., Xu, J., Veedgav, V., Wang, Y., Yu, Y., Yan, D., ... & Xu, S. (2024). Learning the Market: Sentiment-Based Ensemble Trading Agents. arXiv preprint arXiv: 2402.01441.
[9] Sarani, D., & Rashidi-Khazaee, P. (2024). A Deep Reinforcement Learning Approach for Trading Optimization in the Forex Market with Multi-Agent Asynchronous Distribution. arXiv preprint arXiv: 2405.19982.
[10] Qiu, D., Wang, J., Wang, J., & Strbac, G. (2021, August). Multi-Agent Reinforcement Learning for Automated Peer-to-Peer Energy Trading in Double-Side Auction Market. In IJCAI (pp. 2913-2920).