The Evolution of Reinforcement Learning in Quantitative Finance

NIKOLAOS PIPPAS, University of Warwick, United Kingdom

CAGATAY TURKAY, University of Warwick, United Kingdom

ELLIOT A. LUDVIG, University of Warwick, United Kingdom

Reinforcement Learning (RL) has experienced significant advancement over the past decade, prompting a growing interest in applications within finance. This survey critically evaluates 167 publications, exploring diverse RL applications and frameworks in finance. Financial markets, marked by their complexity, multi-agent nature, information asymmetry, and inherent randomness, serve as an intriguing test-bed for RL. Traditional finance offers certain solutions, and RL advances these with a more dynamic approach, incorporating machine learning methods, including transfer learning, meta-learning, and multi-agent solutions. This survey dissects key RL components through the lens of Quantitative Finance. We uncover emerging themes, propose areas for future research, and critique the strengths and weaknesses of existing methods.

1 Introduction

Over the past decade, interest and development in Artificial Intelligence (AI), particularly Reinforcement Learning (RL), have grown significantly. Progress in RL is notable in gaming, with recent algorithms achieving and surpassing human-level proficiency in Go, Chess, and StarCraft [288, 289, 316]. These advancements have spurred the exploration of RL in finance, particularly Quantitative Finance (QF). Although RL shows promise, it also faces steep challenges in the complex and dynamic QF domain. These challenges include the unpredictability of financial markets, high computational demands, and the need for robust model interpretability [89]. In addition, RL applications in finance must address critical challenges such as the transition from simulation to real-world application, sample efficiency, and the balance between online and offline RL settings. This paper reviews RL applications in QF, encompassing key aspects such as Portfolio Management (PM), option pricing, and asset allocation. However, these potential advantages come with the caveat that RL models must be carefully designed and validated to avoid overfitting and ensure robustness in real-world financial markets.

Prior to RL's introduction to specific domains in finance, traditional Machine Learning (ML) methods were employed in attempts to establish successful trading systems. Atsalakis and Valavanis [17] provide a comprehensive survey of these practices, including advanced machine learning techniques like Neural Networks (NN) and neuro-fuzzy systems for financial market forecasting. Furthermore, Ozbayoglu et al. [252] review recent developments and attempts focused on Deep Learning (DL)[1]. The general pattern in these methods consists of a two-step process:

(1) The training of an ML model such as a Support Vector Machine (SVM), NN, or Decision Tree with a specific dataset (features), followed by the generation of a forecast or signal over *n* periods ahead;

(2) The integration of this forecast or signal into a trading system to determine actual trading action or holdings (e.g., buy, sell, or hold in three discrete representations) at a single stock or portfolio level.

Despite the interest in the academic literature, this framework has several limitations [236], which can potentially be better addressed by RL:

---

[1]The term "Deep Learning" can be traced in Dechter [80]

Authors' addresses: Nikolaos Pippas, Nikolaos.Pippas@warwick.ac.uk, University of Warwick, Coventry, United Kingdom, CV4 7AL; Cagatay Turkay, Cagatay.Turkay@warwick.ac.uk, University of Warwick, Coventry, United Kingdom, CV4 7AL; Elliot A. Ludvig, Elliot.Ludvig@warwick.ac.uk, University of Warwick, Coventry, United Kingdom, CV4 7AL.

(1) **Unsuitable optimisation metrics:** In the traditional ML-based approaches, the focus on minimising forecast error is often misaligned with the practical needs of financial trading, where Risk Performance Measures (RPM) like the Sharpe ratio (SR) [281][2] are more relevant. This disconnect can lead to suboptimal outcomes. RL trading systems, by contrast, naturally optimise for selected RPMs or other desired measure. Nevertheless, a notable exception exists in Mabu et al. [222], which targets optimising forecast error rather than any RPM. Despite its potential to dynamically optimise for various performance measures, RL must contend with the high variability and noise inherent in financial data, which can lead to unpredictable results if not properly managed.

(2) **Limited computational agility:** The two-step process in supervised learning increases complexity and slows predictions. In High-Frequency Trading (HFT) settings, where market conditions change rapidly, these delays can render forecasts quickly irrelevant [117], highlighting the need for a rapid, online framework that ensures timely decision-making. RL systems, with their ability to learn and adapt in real-time, offer a more agile alternative by continuously updating strategies as new data becomes available, thus better aligning with the fast-paced demands of HFT environments.

(3) **Limited integration of the financial environment:** Traditional methods, such as the conventional portfolio optimisation [226], consists of a process that involves two distinct steps: first, calculating the expected returns and the covariance matrix of the assets[3]; second, using these inputs in mean-variance optimisation[4] [226] to manage a predefined risk budget. The separation of these steps can limit the cohesion and adaptability of the framework. In contrast, RL integrates this two-step process into one, promoting a more cohesive framework.

(4) **Limited consideration of constraints:** Traditional frameworks typically incorporate constraints like transaction costs and liquidity in a static manner, relying on predefined assumptions that may not accurately capture the dynamic nature of financial markets. In contrast, RL frameworks allow for real-time integration of these constraints. However, it's important to note that RL models often simplify transaction costs as fixed and assume certain execution, overlooking market realities like varying bid/ask spreads [47, 48, 236]. Some studies, such as [85, 102, 327], have addressed these complexities by incorporating execution and slippage costs.

(5) **Adaptability to Changing Market Conditions:** Financial markets are inherently dynamic and continuously evolving [75]. Traditional methods, such as those proposed by Markowitz [226], may struggle to capture these changes promptly, often proving too slow to adapt to shifting market conditions. In contrast, RL algorithms are capable of continuously learning and adapting in real-time, offering a significant advantage in responding to market shifts, such as regime changes or unexpected events, which traditional models may not effectively manage [238, 302].

Despite its potential, applying RL in QF faces significant challenges. QF's complexity and dynamism considerably surpass RL's conventional learning tasks. In addition, fully representing the financial environment is a near-impossible feat, necessitating substantial effort to create meaningful features. Financial data, laden with noise, exhibit non-stationarity and stochasticity [75], which can lead to unpredictable results. Researchers often counter these challenges by adopting certain simplified assumptions, such as the investor's inability to influence financial markets or the absence of risk aversion, leading to total investment [244, 245]. These assumptions imply that the investor or investment is small. A further challenge lies in maintaining the balance between exploration and exploitation: overemphasising the former could inflate transaction costs. The temporal credit assignment problem [300], which arises when the effects of an action are not immediately apparent, is another potential issue in QF [238]. Moreover, interpretability is crucial in finance as stakeholders must understand how an algorithm arrives at its decisions and validate its recommendations. While this 'black box' issue is common across many machine learning models, including deep learning, it is particularly critical in RL due to the sequential decision-making involved. The final notable challenge is the limited availability of historical financial data and the sample inefficiency of RL algorithms, which could restrict the agent's sample space for discovering the optimal policy. In the rest of the paper, we will review several proposed solutions to these challenges, such as the implementation of transfer Learning [253]. In section 8, we also present concerns about evaluation practices, model interpretability, and the potential for overengineering in the literature.

---

[2]The Sharpe ratio quantifies the return of an investment exceeding the risk-free rate per unit of risk, as measured by standard deviation.
[3]Expected returns can be, for example, the past returns of each asset over a given time window, such as one year.
[4]Mean-variance optimisation is a convex optimisation process that balances maximising expected returns and managing risk.
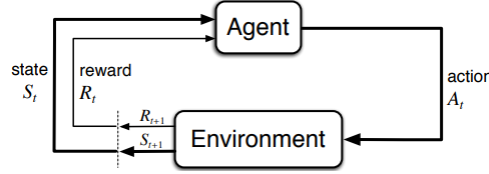
Fig. 1. The general schematic of agent-environment interaction under the RL framework [302].

1.1 Reinforcement Learning and Finance

A good definition of RL is provided by Sutton in [302]: "*Reinforcement learning is learning what to do–how to map situations to actions–so as to maximise a numerical reward signal.*" This simple sentence conveys all the crucial notions of RL. To unpack this point further, the general flow of an RL system is laid out in Figure 1. The agent receives information about the state $S_t \in \mathcal{S}$ and a reward at time t and reacts upon this information with actions $A_t \in \mathcal{A}$. The resulting action feeds back to the environment, and the system generates a new state S_{t+1} and a new numerical reward $R_{t+1} \in \mathcal{R} \subset \mathbb{R}$ at time $t + 1$.[5] The goal is to learn a policy to maximise the expected total reward, which effectively creates a trajectory that can be represented as follows:

$$S_0, A_0, R_1, S_1, A_1, R_2, \ldots$$

RL is a framework for solving sequential decision-making problems. Naturally, many real-world applications fit this framework. For example, video games, robotic control, and driving can be seen as sequential decision-making problems [112, 231, 288]. The framework above also fits in finance, particularly QF. The agent could be a trader or a portfolio manager who observes the current state of financial markets (the environment) and acts upon this information to maximise a reward function (e.g., SR). Figure 2 illustrates a general RL-based framework in the context of QF, mapping key concepts such as the agent, state, action, reward, and environment. The figure shows several options for these components, depending on the specific QF application context.

Figure 3 outlines a timeline of significant RL-based contributions in QF. As is illustrated here, the first explicit applications of RL to finance can be seen as early as the mid-90s with applications of Critic-only (e.g., [244]) and Actor-only (e.g., [235]) methods. The early 2000s saw the use of Actor-Critic methods (e.g., [64]) with deep-learning methods (e.g., [167]) introduced a bit later with multi-agent solutions (e.g., [61]) gaining popularity in recent years.

1.2 Temporal Dynamics in Financial Applications

The time unit in the context of QF can vary depending on the specific application. It could be seconds, minutes, hours, days, months, or even years. The choice of time unit depends on the frequency of trading strategies employed. For example, high-frequency trading strategies might use milliseconds or microseconds, while long-term investment strategies might use days, weeks, or, in rare cases, even years. Moreover, in some QF applications, the rewards might not be immediate but delayed. For example, a reward might be given only at $t + 12$, representing a delayed return on investment, such as dividends. The RL framework must account for such delayed rewards, ensuring that the agent can still optimise its policy to maximise long-term gains despite the delay. This requires careful consideration of the timing and valuation of future rewards to ensure robust performance.

1.3 Multi-Agent systems in Finance

In RL, Multi-Agent Systems (MAS) extend the single-agent paradigm to environments where multiple agents interact, either cooperatively or competitively. The multi-agent model formalism in RL is crucial for addressing complex, dynamic and decentralised decision-making processes found in financial markets [52, 263]. Generalising from the single agent, we have:

- **Agents:** Each agent i has its own set of states S_i, actions A_i, and policies π_i. Agents can have distinct state spaces S_i depending on their roles and the information they can access. Similarly, agents can have distinct action spaces A_i, reflecting their different capabilities or roles in the environment.
- **State Space:** The joint state space S is a combination of all individual states $S = S_1 \times S_2 \times \ldots \times S_n$.
- **Action Space:** The joint action space A is the Cartesian product of all individual action spaces $A = A_1 \times A_2 \times \ldots \times A_n$.
- **Reward Function:** The reward function $R : S \times A \to \mathbb{R}^n$ provides a vector of rewards for all agents, where each component R_i corresponds to the reward received by agent i.

---

[5]$\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $\mathcal{R}$ is the reward space (a subset of $\mathbb{R}$), representing the possible states, actions, and rewards in reinforcement learning.
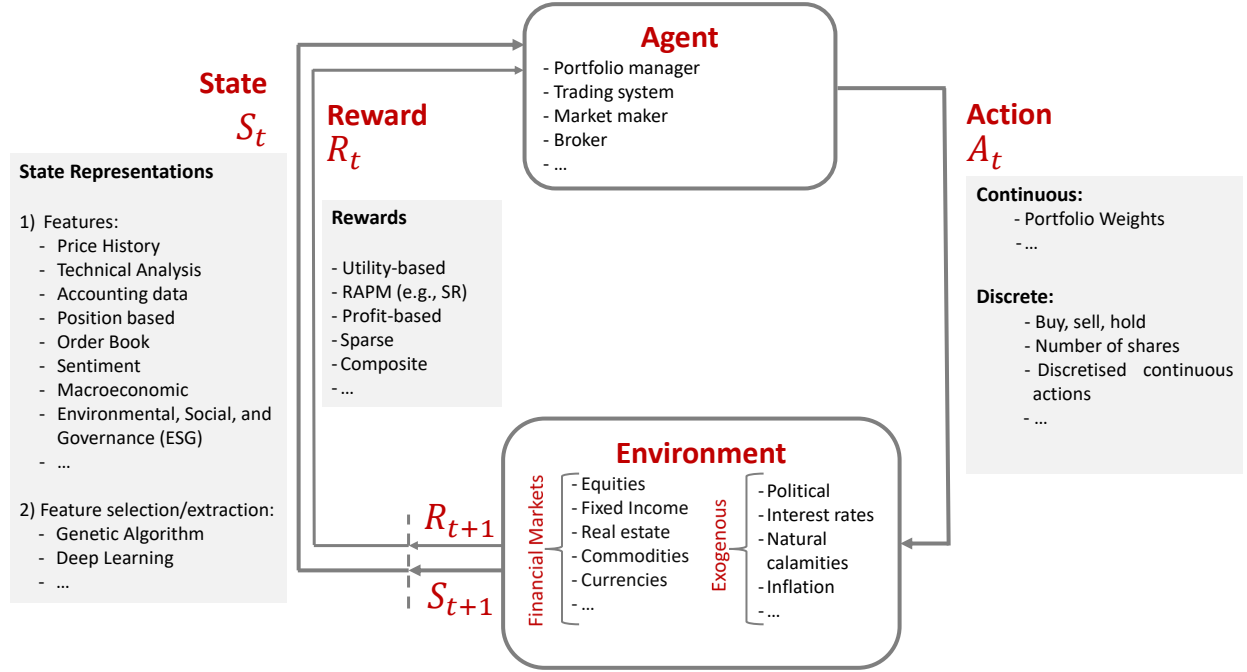
Fig. 2. The overview specifies how an agent and the environment interact in the QF domain using the classical RL framework depicted in Figure 1. With this, we map concepts, techniques and practices from QF that are identified in the survey to the components of the RL framework. (Note: Profit-based rewards can include financial gains such as dividends and pay-offs)

- **Policies:** Each agent follows a policy $\pi_i : S_i \to A_i$, mapping states to actions.
- **Objective:** The goal is to find a set of policies, $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$, where each policy π_i maximises the cumulative reward for its respective agent. Depending on the application, this may involve maximising the cumulative reward for all agents collectively or maximising individual rewards independently.

Interaction is crucial for modelling realistic financial systems in which traders might share market information or collaborate on trading strategies. In multi-agent RL (MARL), agents can interact in various ways, including:

- **Information Sharing:** Agents share observations or information to enhance decision-making, leading to more informed actions and better performance [306].
- **Shared States:** Agents access a global or subset of shared states, enabling coordinated actions [108].
- **Joint Actions:** Agents coordinate actions to achieve common goals, such as synchronising trades to influence market prices [136].
- **Cooperative and Competitive Interactions:** Agents work cooperatively for joint rewards or competitively for individual rewards, depending on the financial application [53].

In the context of finance, a multi-agent RL setup might involve various trading agents operating simultaneously, each with its distinct strategy and goal. For example, one agent might focus on long-term investments, while another specialises in high-frequency trading. These agents interact with each other and the market, influencing and responding to market dynamics. By incorporating multi-agent models, we can better simulate financial markets' complex and competitive nature. Finally, we discuss applications of MAS solutions in QF in subsection 6.6.

1.4 Contribution and Paper Organisation

This survey assesses 167 publications to provide the first comprehensive review of the critical components of an RL agent within QF. Furthermore, our review demonstrates the diverse applicability of RL in different QF contexts. Although the recent surge in interest in RL within finance has produced several survey publications [66, 105, 139, 239], our study is distinctively focused on QF, diverging from other studies that encompass broader economic applications [66, 239] and rooted more closely within the literature

1996: Fist RL publication in finance under the Critic-only framework by **Neuneier**

1997: Moody and Wu introduced the first Actor-only based trading system (named RRL)

2001: Chan and Shelton were the first to use an Actor-Critic framework in RL under market-making problem

2002: Dempster and Romahi were the first to introduce genetic algorithms as part of the feature selection process

2003: Lee et al. along with there subsequent research made the first attempt to use a multi-agent approach to financial trading problem

2006: Nevmyvaka et al. first application of RL under the trade execution problem
2006: Dempster and Leemans publish possibly the first RL based end-to-end trading system

1994   2002   2010   2018   2024

Tan et al. use Q-learning for Business cycle stock trading:**2011**

Eilers et al. applied Q-learning to trade seasonal effects: **2014**

Jin and El-Saawy possibly the first paper to apply DQN in the domain of financial markets: **2016**
Deng et al. were the first to use DL as a feature extraction mechanism: **2016**

Jiang and Liang introduced the first portfolio optimisation solution under the RL framework:**2017**

Jeong et al. used transfer learning in the context of single stock trading:**2019**
Almahdi and Yang provide the first constraint portfolio optimisation solution based on RL: **2020**

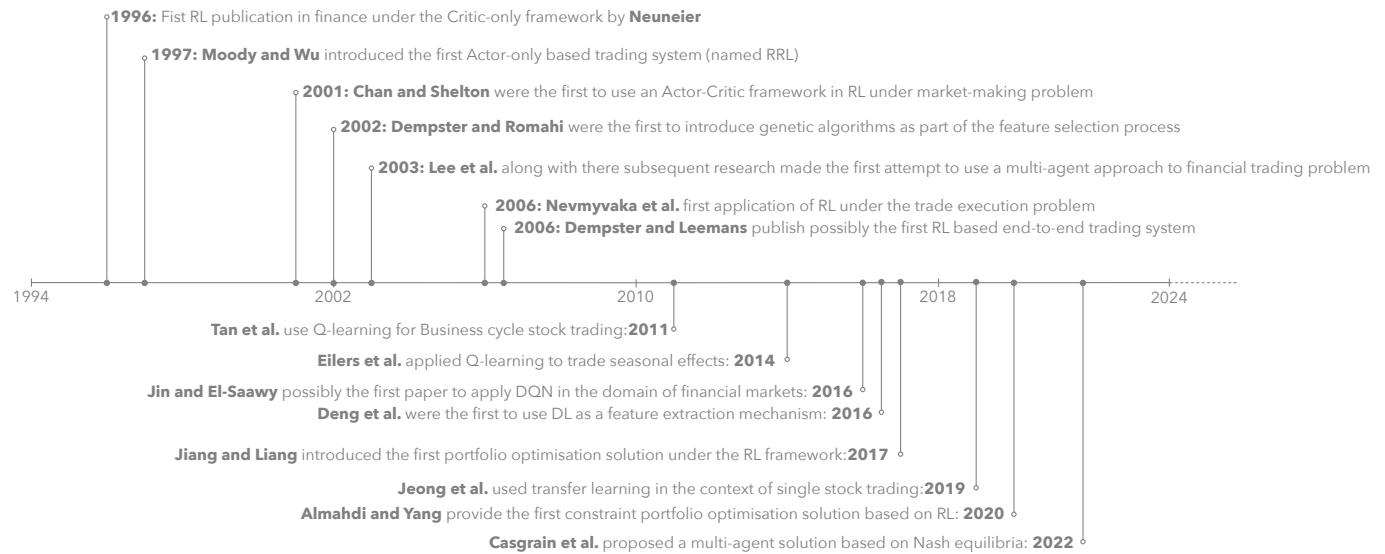Casgrain et al. proposed a multi-agent solution based on Nash equilibria: **2022**

Fig. 3.   Timeline of important publications in QF under the RL-based framework.

on finance theory and practice. Moreover, we avoid comprehensive textbook reviews of RL theory such as those presented in other surveys [139], opting for the necessary definitions and references as necessary.

The articles for this survey were sourced from Google Scholar using RL and finance keywords and employing the snowball method [336] on relevant references, spanning 1996-2022. In particular, only 27. 5% of the publications predate 2016, reflecting the recent surge in interest and contribution of RL to QF. This can be mainly attributed to the introduction of Deep Q Networks (DQN) [231, 232], and consequently, Deep Reinforcement Learning (DRL).

In other research fields, related concepts were known as trial-and-error learning, learning with a critic, optimal control, and dynamic programming. Therefore, a range of papers [121, 158, 168] link their research to concepts similar to those in RL. Additionally, while multi-armed bandits (MABs) and sequential optimisation methods are significant in online learning and decision-making, they differ from the broader RL framework covered in this survey. MAB problems are typically simpler and do not involve state transitions. Thus, this survey focuses on comprehensive RL frameworks in QF while acknowledging key MAB research, such as [18, 19, 159]. Thus, we focus solely on methods that explicitly fall into the RL research area in QF.

This survey starts with dissecting common RL methods and key RL components, e.g., the agent, environment and rewards, through the lens of Quantitative Finance. Building on an in-depth dialogue with quantitative finance literature and practice and reflecting on recent advances in machine learning, we uncover emerging themes, propose areas for future research, and critique the strengths and weaknesses of existing methods.

2   Critical Considerations for RL in QF

Before delving into the research and implementation of RL in QF, we introduce some critical dimensions and considerations of RL in QF. Figure 2, as we covered in subsection 1.1, expands the generic RL framework and illustrates several options for potential mappings between key RL components and relevant concepts, techniques and practices from QF. In addition to these, there are some critical considerations for incorporating RL in QF including the transition from simulation to real-world applications, sample efficiency, online versus offline RL settings, on-policy versus off-policy frameworks, and how RL interacts with the unique challenges of financial markets.

2.1   Transition from Simulation to Real-World Application

Deploying RL models in real financial markets presents unique challenges. Unlike controlled simulation environments, real-world markets are unpredictable and influenced by myriad factors such as information asymmetry, variable transaction costs, taxes, and noise traders, which can significantly affect model performance [236, 297]. Therefore, a robust validation and simulation process is essential to ensure that RL models accurately capture the complexities of chaotic financial markets. Incorporating domain knowledge improves model resilience and adaptability, providing a more accurate reflection of market conditions, and improving

generalisation capabilities. Furthermore, developing a comprehensive risk management framework [83, 230] is vital to mitigate regulatory and stakeholder risks. This framework should include thorough backtesting, stress testing, and continuous monitoring to ensure that RL models can adapt to market changes and maintain their performance in various scenarios.

2.2 Sample Efficiency

Improving sample efficiency is crucial in RL, especially in financial applications where data can be sparse, have a short history, or be expensive to obtain [139]. Many samples drawn from the environment are often necessary for stable convergence and low variance. Techniques such as model-based RL, experience replay, and transfer learning are valuable in this context.

Model-free methods are typically slower to learn, whereas model-based methods build a transition model from the environment's feedback, allowing the agent to use this model to understand the effects of actions on states and rewards [263]. Model-based RL can generate additional training data, enhancing learning efficiency [288]. For instance, model-based RL can simulate various market conditions to create synthetic data in a stock-trading scenario. This data helps the model learn how to handle different market situations. However, for this approach to be effective, the model must accurately represent the environment. Experience replay allows RL agents to reuse past experiences, thereby improving learning from limited data [232]. In the context of financial applications, agents can reuse previous experiences to improve learning efficiency. Transfer learning enables leveraging knowledge from related tasks to improve performance in new but similar tasks, thus addressing data scarcity [187]. For a discussion of transfer learning and model-based RL, see subsection 6.3 and Section 9. When the transition model is accurate, model-based RL can also serve as a transfer learning mechanism, further enhancing learning efficiency [263].

2.3 Online vs. Offline RL Settings

Online learning in QF refers to agents learning and adapting continuously as new data arrive. This approach benefits from the ability to dynamically update strategies in response to market changes. However, this can be highly impractical in HFT, where decisions occur in microseconds to milliseconds, because online learning requires real-time updating of algorithm parameters and significant computational resources to process new data. As pointed out by Hambly et al. [139], a potential solution in this setup might be to collect data with a pre-specified exploration scheme during trading hours and update the algorithm with newly collected data after trading closes. Transfer learning can also be leveraged in this context to bridge the gap between offline and online learning. Initially, the RL agent can be pre-trained using offline RL on historical data, learning general strategies and patterns. Once deployed, the agent can then use online learning to fine-tune these strategies in response to real-time market conditions. This hybrid approach can mitigate some of the computational burdens of online learning by reducing the frequency and extent required for real-time updates. Also, certain components of the RL framework can be constructed to support online learning, such as the introduction of Differential SR [235–238], which is an online version of SR (see subsection 5.3).

In contrast, offline RL uses historical data to develop strategies without continuous interaction with the environment. This method is particularly useful for backtesting and strategy development, where real-time data streams are not necessary. Offline RL can leverage vast amounts of historical market data to train models and then be deployed in live environments after thorough validation. Furthermore, offline learning does not face the computational constraints that online learning faces, allowing for extensive preprocessing and model training using conventional computing resources. Though training time can be extensive due to large datasets, it avoids the real-time computational demands of online RL.

2.4 On-Policy vs. Off-Policy Frameworks

The distinction between on-policy and off-policy methods significantly affects their application in various financial contexts, each presenting unique challenges and opportunities.

On-policy algorithms, such as Proximal Policy Optimisation (PPO) [280], use data generated only from the current policy. This means that only actions taken by the current policy generate data at each training iteration. Consequently, the data become unusable after policy updates, making on-policy methods inefficient. However, on-policy algorithms are typically more stable and exhibit lower variance because they learn directly from the policy they are improving [263]. For example, SARSA [273], an on-policy algorithm, is used in stock trading models due to its ability to adapt to market changes efficiently, resulting in lower computational resources requirements and robust performance under varying market conditions [77, 86]. Furthermore, on-policy methods could be practical in HFT due to the vast amount of data available despite being sample inefficient. In PM, trading at lower frequencies (e.g. monthly) could be less practical due to the scarcity of available data.

In contrast, off-policy methods, such as DQN [232], do not have this limitation, allowing them to use data generated from different policies. This makes off-policy methods more sample-efficient, as they can reuse past experiences stored in memory. However, this requires significant memory storage to preserve past experiences. Additionally, off-policy methods can face challenges with stability and variance due to the complexity of updating policies from a mixture of different data sources and policies, which can amplify approximation errors and lead to instability [263, 302]. Off-policy algorithms often leverage large datasets for training and can learn from historical data, making them particularly suitable for environments where collecting real-time data is impractical. For example, the application of Q-learning [330] in financial trading demonstrates its effectiveness in using past market data to inform future trading decisions, which could lead to better adaptation to non-stationary market dynamics [77]. Furthermore, off-policy methods could be used in HFT due to their ability to process large amounts of data, although they may require longer training times, which is crucial in the HFT context. In PM, trading at lower frequencies (e.g. monthly) could benefit more from off-policy methods due to their better use of available data.

A hybrid approach can be beneficial. For example, an RL agent could be pre-trained using off-policy methods on historical data to develop a baseline strategy. This agent can then be fine-tuned using on-policy methods in a live environment for stability and adaptability. In practice, such as in equity trading or broader quantitative finance setups, an agent initially trained with DQN can switch to PPO in live trading to balance stability and responsiveness, combining the benefits of both methods [267, 355]. General care should be taken when using historical data, as it could lead to biases and unintended trading behaviours. In portfolio management, reliance on past data can result in strategies that perform poorly under new market conditions due to non-stationary dynamics [207]. Thorough testing and validation are necessary to mitigate these risks.

3 Main Reinforcement Learning Methods

Contemporary literature distinguishes four primary RL methods: Value-based (Critic-only), Policy-based (Actor-Only), Actor-Critic, and model-based. This section delves into a thorough exploration of the principal RL techniques encapsulated within these categories. We provide an overview of the general framework, the advantages and disadvantages, and the applicability of each method within the field of QF. Additionally, within these categories, we incorporate other forms of RL approaches such as multi-agent methods [157, 191].

Table 1 presents a classification of each publication reviewed in this survey according to the RL method and algorithm used. Within the Value-based approach, DQN and Q-learning [330] emerge as the most frequently used algorithms. Recurrent Reinforcement Learning (RRL) [235–238] assumes dominance in the Policy-based methodology, and PPO [280], and DDPG [208] are the most popular in the Actor-Critic approach. Overall, the Value-based method has received the most extensive research, followed by the Policy-based, Actor-Critic, and model-based methods. In the interest of simplicity, we have occasionally categorised publications under the "other" bucket when the proposed algorithm is non-standard or infrequently used. Furthermore, we have classified several publications under multiple categories due to their use of a diverse range of algorithms. Lastly, we have refrained from including publications that fall outside these four categories for the sake of brevity. For example, [342] applies Inverse RL (IRL) [247] to QF, a method that is not discussed in this survey.

3.1 Value-Based Methods

3.1.1 ***Core Framework for Value-based RL in QF.*** Value-based methods are foundational in RL. The agent's objective is to learn the value of different actions in various states to maximise cumulative rewards. Neuneier's work [244] is one of the earliest contributions in this field, which set a methodological precedent that has informed many subsequent studies. This section outlines a generic framework derived from these methodologies:

(1) **Define a Finite Set of States** S_t**:** States represent environment-derived information at each time point $t \in \{1, 2, \ldots, T\}$. This information includes financial accounting data, prices, sentiment, and technical indicators.
(2) **Define a Set of Actions** $A_t \in \{Buy, Sell, Hold\}$**:** These are the possible actions of the agent at each time $t \in \{1, 2, \ldots, T\}$.
(3) **Establish Transition Probabilities:** These probabilities, typically unmodelled, define state transitions based on actions.
(4) **Formulate a Reward Function** R_t**:** Provides numerical feedback to the agent in response to its preceding action.
(5) **Create a Policy** π**:** Maps states to actions for the agent to follow.
(6) **Construct a Value Function** V**:** Maps states to the agent's expected total discounted reward from a given state until the episode's end under policy π.

Approach	Method	RL-Algorithm	Publication	Count
Model-Free	Value-Based (Critic-Only)	DQN	[2], [55], [56], [57], [59], [79], [91], [156], [157], [163], [170], [174], [175], [183], [195], [196], [204], [218], [219], [243], [248], [255], [256], [282], [294], [305], [311], [313], [321], [327], [328], [332], [338], [347], [348], [352], [354]	37
		Q-Learning	[29], [40], [54], [63], [76], [77], [81, 82], [90], [92], [93], [104], [119], [138], [148], [155], [161], [167], [172], [191–194], [199], [201], [209], [218], [244, 245], [246], [258], [283], [290], [307], [357], [359]	36
		SARSA	[64], [67], [77], [86], [134], [177], [258], [284], [290], [305]	10
		Other	[7], [55], [88], [92], [201],[258]	6
	Policy-Based (Actor-Only)	PG	[35–38], [51], [74], [164], [207], [323], [326], [352], [353], [354]	13
		REINFORCE	[78], [87], [90], [327], [332]	5
		RRL	[5], [8, 9], [39], [43], [47, 48], [83], [84, 85], [116], [119], [125], [131], [131], [149], [200], [205], [217], [223–225], [235–238], [285], [286], [340], [349–351]	31
		Other	[102], [197], [266], [275], [324], [325], [333]	7
Model-Free	Actor-Critic	A2C	[171], [332], [343], [352]	4
		DPG	[6], [165, 166], [213], [338], [344]	6
		DDPG	[2], [6], [25], [54], [171], [180], [202], [207], [276], [329], [339], [343], [345], [346],	14
		PPO	[6], [49], [79], [91], [101], [118], [157], [171], [207], [212], [313, 314], [343], [347]	14
		TRPO	[44], [317]	2
		Other	[6], [31], [61], [64], [120], [135], [175], [200], [222], [265], [293], [347]	12
Model-Based	Model-Based	Other	[345], [135], [332]	3

Table 1. Categorisation of each publication based on RL methods and algorithms.

The agent continuously interacts with the market by processing market data, making investments, and receiving returns over many trials. Through this process, the agent aims to discover the best possible strategy, known as the optimal policy ($\pi = \pi^*$). This entire approach is an example of a Markov Decision Problem (MDP) applied to financial trading. This framework highlights the essential components required for Value-based RL in finance. By defining states, actions, transition probabilities, reward functions, policies, and value functions, we can create a system where an agent learns to make optimal trading decisions through continuous interaction with the market. This framework, despite its simplicity, serves as a building block for more advanced frameworks.

3.1.2   *General Observations, Comments and Definitions for Value-based Methods.* Value-based methods are a well-established branch of research in RL applied to trading systems, despite their noted shortcomings. A significant challenge is the discrete nature of the action space, complicating practical trading applications 5.2.

The agent creates a value function to estimate the results of actions such as buying, holding, or selling, helping to choose the best action. This approach often uses model-free RL algorithms like Q-learning [330] and SARSA [273] to optimise the expected total reward. Q-learning, an off-policy RL algorithm, is proven to converge to the optimal solution in the tabular setting under specific conditions [331]. In Q-learning, the Q-values, $Q(S_t, A_t)$, represent the expected rewards for taking an action A_t in state S_t. These Q-values are updated according to the rule:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]. \tag{1}$$

This update rule adjusts the current Q-value by incorporating the observed reward R_{t+1} and the maximum estimated future reward, weighted by the learning rate $\alpha \in (0, 1]$. Over time, this process helps refine the Q-values, guiding the agent toward the optimal action values under specific conditions. Specifically, Q-learning is guaranteed to converge to the optimal solution with probability 1, provided that all state-action pairs are visited infinitely many times and the learning rate satisfies certain decay conditions. These convergence conditions are crucial for ensuring that the algorithm performs well in practice.

Q-learning and SARSA are traditionally used in tabular settings, which are suitable for small, discrete state spaces. However, these algorithms can also be adapted to more complex state spaces through function approximation techniques. While tabular methods are inherently limited in scalability, FA allows these algorithms to generalize from similar states, making them applicable

in real-world scenarios with large or continuous state spaces [302]. Despite this adaptability, using function approximation introduces practical challenges, such as the risk of overfitting and increased computational demands.

The field experienced a transformative breakthrough with the introduction of DQNs by Mnih et al. [231, 232], leveraging neural networks as function approximators to achieve human-level performance in Atari video games. This amalgamation of RL with deep learning is known as Deep Reinforcement Learning (DRL), and in Q-learning's specific context, it is known as DQL or DQN. The authors used experience replay [211] and target networks for a stable DRL agent. Subsequent proposed improvements have included a Prioritised Replay Buffer [277], Double Q-learning (DDQN) [143], Multi-step learning [301], Noisy Networks [110], Dueling DQN [319], and Distributional RL [32].

In conclusion, the contemporary surge of RL in QF can be traced back to the advent of DQN. Despite the constraints of the Value-based framework, it remains a vibrant research area with numerous significant publications within this survey's scope. The subsequent section explores Policy-based methods that address some of these limitations by directly optimising the policy.

3.2 Policy-Based Method

3.2.1 *Core Framework for Policy-Based RL in QF*. Policy-based methods in RL focus on directly optimising the policy that dictates the agent's actions. Unlike Value-based methods, which estimate the value of actions to derive the best policy, Policy-based methods directly search for the optimal policy that maximises the cumulative reward. This approach can be particularly advantageous in environments with continuous action spaces and complex dynamics.

Among these methods, the RRL framework stands out due to its extensive use in the reviewed literature and its clear relevance to finance [235–238]. The RRL framework is particularly suited for financial applications because it can capture the temporal dependencies and sequential nature of trading decisions. Therefore, this subsection will focus on the RRL framework. Following Moody and Wu [235], an agent's actions are represented by:

$$A_t = A\left(\theta_t : A_{t-1}, I_t\right) \in \{-1, 0, 1\} \text{ with } I_t = \{z_t, z_{t-1}, z_{t-2}, \ldots, y_t, y_{t-1}, y_{t-2}, \ldots\}. \tag{2}$$

Here, θ_t refers to the learned parameters of the agent, A_t corresponds to a trading position that can assume one of three states - sell, neutral, or buy, while A_{t-1} represents the preceding action at time $t-1$, I_t, which denotes the information set at time t, is used as a state representation, composed of lagged asset prices z_t and other external variables y_t. A_t can be defined [238] as:

$$A_t = sign\left(uA_{t-1} + v_0 r_t + v_1 r_{t-1} + \ldots + v_m r_{t-m} + \omega\right), \tag{3}$$

where, r_t denotes the price return, and $\theta_t = \theta = \{u, v_i, w\}$ with $i \in \{0, 1, \ldots, m\}$. The RRL framework further optimises the trading system by maximising a performance function U_t. This function can represent different goals, such as maximising profit, improving a utility function of wealth, or enhancing performance ratios like the Sharpe Ratio (SR). Essentially, it helps the trader aim for better financial performance based on their specific goals. One of those choices is the additive profits utility reward function [238]:

$$U_t(\theta) = P_T = \sum_{t=1}^{T} R_t = \mu \sum_{t=1}^{T} \left\{ r_t^f + A_{t-1}\left(r_t - r_t^f\right) - \delta_t \left|A_t - A_{t-1}\right| \right\}, \tag{4}$$

where, P_T denotes the cumulative profit at the end of trading period T, R_t the profit or loss at time t, and T the total time steps. $\mu > 0$ represents a fixed position size when buying/selling a stock, r_t and r_t^f are the absolute prices changes for the period from $t-1$ to t for the risky asset (stock) and the risk-free asset (e.g., T-Bills), respectively, and δ_t denotes transaction costs from buying/selling the risky asset.

An important aspect of the RRL framework is the online optimisation approach using stochastic gradient ascent to adjust the parameters. This process involves calculating how changes in the agent's strategy (represented by the parameters θ) affect its performance (measured by the utility function U_t). Essentially, it looks at how both the current and previous actions influence the agent's success, allowing for continuous improvement. Imagine a trader adjusting their strategy based on both their recent trades and past experiences to maximise profits. The gradient calculation is described by the following equation:

$$\frac{dU_t(\theta)}{d\theta} = \frac{dU_t(\theta)}{dR_t} \left\{ \frac{dR_t}{dA_t} \frac{dA_t}{d\theta} + \frac{dR_t}{dA_{t-1}} \frac{dA_{t-1}}{d\theta} \right\}. \tag{5}$$

This equation shows that we need to compute the gradient of the utility function with respect to the actions and then update the parameters based on this gradient. The update is done using the learning rate ρ, as in $\Delta\theta_t = \rho \frac{dU_t(\theta)}{d\theta_t}$. In simpler terms, we

adjust our model parameters by considering how both the current and previous actions influence the reward, and we use this information to make our model more accurate over time.

Numerous versions of this RRL framework have been proposed. For instance, the study by Gorse et al. [131] presents a learning rule based on associative reward-penalty [28], with the output from equation (3) defined as a probability. Gold [125], inspired by [235], introduced an additional hidden layer in the network, capturing more complex patterns compared to the single-layer network, thus serving as a precursor to Deep Recurrent Reinforcement Learning (DRRL) [85].

3.2.2   ***General Observations, Comments and Definitions for Policy-Based Methods****.* Policy-based methods represent the second most explored area of RL in the reviewed literature. These methods provide a direct mapping from states to actions, eliminating the need to compute the expected outcome of different actions as in the Value-based approach, resulting in faster learning processes.

A significant advantage of Policy-based methods is the continuous action space for the agent. Consider a portfolio of stocks: with the Value-based approach, portfolio weights can only take discrete values like buy, sell, or hold. In contrast, the Policy-based approach allows portfolio weights to assume any value in $[0, 1]$ in the long-only case.

Policy-based methods exhibit robust performance even when dealing with noisy datasets, common in stock-related data [90, 238]. Furthermore, Policy-based methods typically converge more swiftly than Value-based methods [133], although the high variance of the gradient leads to a slower learning rate [178]. These methods inherently perform exploration [263], as the stochastic policy yields a distribution over the action space. However, Policy-based methods require a differentiable reward function. Policy-based approaches in finance were pioneered by Moody and Wu [235], with many variants since then proposed. RRL typically uses a recurrent NN structure, creating dependencies over previous steps and facilitating multi-period optimisation.

3.3   Actor-Critic Method

3.3.1   ***Core Framework for Actor-Critic RL in QF****.* The Actor-Critic method in RL is a hybrid approach that combines the strengths of Policy-based and Value-based methods to create a robust learning framework. It consists of two main components: the Actor and Critic modules. The Actor module takes the state S_t as input and determines the action A_t at time t. The Critic module subsequently receives the state S_t and the action A_t determined by the Actor module, evaluates the state-action pair, and computes the reward, adhering to the general framework outlined in subsection 3.1. Generic Actor-Critic frameworks are discussed in [263, 302]. The core components of an Actor-Critic framework can be described as follows:

- **Actor:** The Actor is responsible for selecting actions based on the current state. It learns a policy, which is a mapping from states to actions. This policy can be:
  - **Deterministic:** The Actor always chooses the same action for a given state.
  - **Stochastic:** The Actor chooses actions according to a probability distribution.
- **Critic:** The Critic evaluates the action taken by the Actor by computing a value function. This value function estimates the expected cumulative reward (discounted over time) of being in a given state and taking a particular action.
- **Advantage Function:** The advantage function helps to determine how much better or worse a particular action is compared to the average action taken from that state. It is defined as:

$$A(S_t, A_t) = Q(S_t, A_t) - V(S_t), \tag{6}$$

  where Q is the action-value function and V is the state-value function.
- **Gradient Ascent:** Both the Actor and the Critic are trained using gradient ascent. The Actor updates its policy parameters to maximise the expected cumulative reward, while the Critic updates its parameters to provide more accurate evaluations of the actions.
- **TD Error:** Temporal Difference (TD) error is used to update both the Actor and the Critic. It is the difference between the expected reward and the actual reward received, given by:

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t). \tag{7}$$

The integration of deterministic and stochastic policies within the Actor-Critic framework has shown potential for enhancing decision processes across various complex environments. The first application of the Actor-Critic framework in our survey is found in [64], where the Actor-Critic method is employed to solve the market-making problem (more details, in subsection 7.5).

3.3.2 ***General Observations, Comments and Definitions for Actor-Critic Method***. Among the areas covered in this survey, the Actor-Critic method is the least represented (see Table 1). Yet, it remains among the most compelling of the four primary approaches, as it combines the advantages of both Policy-based and Value-based RL methods. As indicated in subsection 3.2, a notable challenge with Policy-based methods is their *high variance*, which may result in slower convergence or the propensity to become stuck in local optima. The Critic component in Actor-Critic methods mitigates this by providing a value function that stabilises policy-gradient updates, reducing variance. Moreover, Value-based RL methods are prone to *high bias* due to approximation errors in the value function [263]. The Actor component in Actor-Critic methods helps to directly optimise the policy, reducing bias by ensuring continuous adjustment based on accurate evaluations of actions. Several Actor-Critic algorithms featured in this literature are specifically engineered to surmount these obstacles. The algorithmic frameworks predominantly observed within the literature encompassed by this survey include the Deterministic Policy Gradient (DPG) [287], Deep Deterministic Policy Gradient (DDPG) [208], Asynchronous Advantage Actor-Critic (A3C) and the non-parallel version (A2C) [233], Trust Region Policy Optimisation (TRPO) [279], Proximal Policy Optimisation (PPO) [280], Soft Actor-Critic (SAC) [137].

Though comparatively under-researched in QF, the Actor-Critic category shows significant promise in contemporary applications. Various methods within this category, as documented in works by Mnih et al. [208, 233] and Lillicrap et al., [208], have demonstrated state-of-the-art performance, marking this as a potential area for future QF breakthroughs.

3.4 Model-Based RL

3.4.1 ***Core Framework for Model-Based RL in QF***. In model-free RL methods, the agent updates a policy directly from the environment's feedback on its actions. On the contrary, in model-based methods involve constructing a model of the environment, which is then used to simulate and plan actions. A foundational framework can be described as follows:

(1) **Define a Finite Set of States** S_t**:** States represent environment-derived information at each time point t. This information includes financial accounting data, prices, sentiment, and technical indicators.
(2) **Define a Set of Actions** A_t**:** These are the possible actions of the agent at each time t, such as Buy, Sell, and Hold.
(3) **Learn Transition Probabilities:** Construct a transition model that predicts the next state S_{t+1} and reward R_{t+1} given the current state S_t and action A_t. This model can be a neural network or any other function approximator.
(4) **Formulate a Reward Function** R_t**:** Provides numerical feedback to the agent in response to its preceding action, incorporating factors such as profit, risk, and transaction costs.
(5) **Planning and Policy Optimisation:** The learnt model can be used to simulate future states and rewards, allowing the agent to plan and optimise actions. Techniques such as Monte Carlo Tree Search (MCTS) or Dynamic Programming can be used for this purpose [263, 302].

This framework highlights the core components required for model-based RL in finance, facilitating the creation of a system where an agent learns to make optimal trading decisions through simulation and planning. Model-based RL is the least represented method in the current literature [135, 332, 345].

3.4.2 ***General Observations, Comments and Definitions for Model-Based RL***. Model-based methods in RL offer several advantages and present unique challenges, particularly when applied to financial trading systems [6]:

(1) **Learning Speed:** Model-based RL methods typically learn faster than model-free approaches by using the learnt model for planning and action optimisation, crucial for timely financial market decisions.
(2) **Computational Complexity:** Simulating and planning with complex financial models is computationally intensive, requiring efficient algorithms and high-performance computing, especially for HFT applications.
(3) **Risk Management:** Robust risk management is vital. Model-based RL can simulate extreme market scenarios to assess potential risks, helping to develop strategies that maximise returns and manage risks effectively.

4 Environment

4.1 Introduction

In the RL framework, the environment characterises the current state of the system. The agent, the learner, and decision maker interact with this environment, selecting actions based on state information. This setup requires that the agent and the environment

---

[6]see also Section 2.

are mutually exclusive [301], providing distinct boundaries for rewards, actions, and states[7]. In financial contexts, the agent, an asset owner, uses the state of financial markets (the environment) and external factors such as stock indices, interest rates, commodity prices, and macroeconomic, political, and natural risks to inform actions (see, Figure 2).

Assuming that the agent can access all relevant information, the RL problem can be addressed under the MDP framework [28]. However, this assumption implies that the environment is fully observable and that future states depend only on the current state and action, a property known as the Markov property. In financial markets, this assumption may be overly simplistic. Empirical evidence suggests that financial markets exhibit longer memory, influenced by factors such as investor behaviour, economic cycles, and external events. These factors introduce dependencies that extend beyond immediate state transitions [75, 215].

Given the complexity and partial observability of financial markets, a Partially Observable Markov Decision Process (POMDP) framework is more appropriate [16]. In Partially Observable (PO) environments, the transition probabilities between states in financial markets are typically not explicitly modelled. Instead, historical data and statistical methods, such as LSTM or RNN [146], are used to approximate these transitions. This approach acknowledges the inherent randomness and partial observability of financial markets, making exact environment representation virtually impossible.

Financial markets epitomise a PO environment, lacking deterministic representation; a single day's pricing cannot encapsulate the state of the environment, unlike full observability in Atari video games [231, 232]. The use of LSTM and RNN models is particularly relevant in this context, as they are capable of capturing long-term dependencies and patterns in sequential data. These models help address the limitations of the Markov assumption by incorporating memory effects and providing a more robust representation of the financial market dynamics.

In early adoptions of RL in QF, the environment was assumed to be fully observable [244], mainly because the inherent randomness of financial markets makes an exact environment representation virtually impossible. Furthermore, adding more features does not necessarily improve performance [227], necessitating strategic feature selection to manage randomness, avoid the curse of dimensionality [33], and address interpretability issues. Finally, we note that empirical studies have shown that models that incorporate memory effects, such as LSTM and RNN, significantly outperform those based on the Markov assumption, particularly in predicting long-term trends and capturing market anomalies [106].

4.2 Features

Modelling financial markets with their inherent randomness is a considerable challenge. Consequently, input data, feature selection, and extraction become critical to the success of an RL system. The literature presents a diverse range of data sources, features, and feature selection mechanisms, reflecting the current state of research and highlighting the domain-specific relevance of different features. Table 2 presents features and data used in the top fifteen most cited papers in our sample (as of October 2022), which are also ordered chronologically.

Publication	Data Freq.	Period	PH	TI	CP	ME	PV	CH	OB	Oth.	FM
Neuneier [244]	Daily	1986-1996	x		x	x	x				
Moody and Wu [235]	Monthly/30-min.	Several	x		x						
Moody et al. [236]	Monthly/30-min.	Several	x		x						
Moody and Safell [238]	Monthly/30-min.	Several	x		x						
Dempster et al. [81]	1-min.	1994-2002		x	x						x
Nevmyvaka et al. [246]	Milliseconds	1.5 years of LOB	x		x				x	x	
Dempster and Leemans [83]	1-min.	2000-2002	x	x	x						
Lee et al. [194]	Daily	1999-2005	x	x			x			x	
Deng et al. [85]	Tick-level	Several	x							x	x
Jiang and Liang [165]	30-min.	2014-2017	x	x	x						x
Jiang and Liang [166]	30-min.	2014-2017	x	x	x						x
Almahdi and Yang [8]	Weekly	2011-2015	x		x						
Liang et al. [207]	Daily	2015-2017	x							x	x
Xiong et al. [339]	Daily	2009-2018	x		x			x			x
Jeong and Kim [163]	Daily	Several	x								x

Table 2. Mapping of Features for Most Cited Publications in the Survey. PH - Price History, TI - Technical Indicators, CP - Current Position, ME - Macroeconomic, PV - Profit/Value, CH - Cash, OB - Order Book, Oth. - Others, FM - Feature Mechanism.

---

[7]In financial markets, this separation becomes somewhat blurred as an agent's actions—buying and selling assets—can influence the environment. However, most of the current literature assumes that the agent has minimal market impact, a realistic assumption for small investors.

State representation commonly incorporates discrete state, technical analysis, pricing data, macroeconomic indicators, sentiment data, current position, and Limit Order Book (LOB) data. Few publications experiment with or compare different state configurations. For example, Nevmyvaka et al. [246] blended private and market variables, while Briola et al. [49] tested three different state scenarios for the RL agent, each with more features.

4.2.1 **Price History.** Price history and Open-High-Low-Close (OHLC) or OHLCV when incorporating volume [156, 311] values are prevalent features in the literature for assets such as stocks or bonds, along with their derivatives, like 20-day lagged returns [68] or co-integration [94] based features [197]. However, reliance on price-based features should not hinder the consideration of other features. For example, Sherstov and Stone [284] found that using a single price feature underperformed compared to benchmark strategies, hinting at the need for a richer state representation to capture the complexities of the financial market. Moreover, Benhamou et al. [35] noted the high correlation between OHLC features, which could introduce input noise. Thus, feature selections on raw features might be a more effective strategy in certain contexts, as discussed in subsection 4.3.

Volatility, a critical feature derived from historical prices, is surprisingly overlooked in the literature, despite its crucial role in PM [226] and in identifying shifts in financial markets (regime changes) [188]. Its absence in RL-based PM applications [213, 344, 345] is noteworthy given its importance [35]. Volatility has only recently been used in PM applications [35–38]. In other RL-based finance applications, different volatility measures have been successfully used as a tool to discover regime changes [223–225]. In effect, the authors extended the original work of Moody and Wu [235] by adding a regime-switching extension to the RRL framework. In Bekiros [31], a comparison is made between changes in 20-day volatility and the previous day, while Tan et al. [307] used the standard deviation of a stock price and its correlation with the Dow Jones index to identify stock-specific cycles. Zhang and Maringer [351] included Garch [46] volatility in state representation, marking its first use in an Policy-based approach. Finally, certain authors incorporate the full covariance of closing prices in state modelling [171].

4.2.2 **Technical Analysis.** Technical analysis consists of indicators or rules designed to predict the direction of the price using past prices and volume variables. However, the effectiveness of these indicators is often challenged by the Efficient Market Hypothesis (EMH), which posits the unpredictability of stock market prices [97]. As such, research on these indicators has produced mixed results, and some studies have methodological issues in their testing procedures [254].

Despite such criticism, numerous academics have questioned the EMH, arguing that future prices can be predicted based on past prices, thus challenging the random walk hypothesis [214]. Consequently, several researchers have used technical indicators to represent the agent's environment within this domain, including [81–83, 255, 327, 338, 352]. However, there are also publications that question the efficacy of technical analysis in the realm of RL in QF [83].

Key technical indicators often used include the Moving Average (MA), Exponential Moving Average (EMA), Moving Average Convergence/Divergence (MACD), Japanese candlestick, and the Relative Strength Index (RSI) [241]. Moreover, some researchers have created features based on the original definitions of technical indicators, demonstrating the adaptability of these tools [116].

4.2.3 **Fundamental Data and Factor Investing.** Accounting data from financial statements underpin traditional factor investing, with numerous strategies developed over half a century based on such data aimed at explaining expected asset returns [72]. As identified by Harvey and Liu [142], more than 400 factors have been proposed, covering a wide spectrum from macroeconomic to statistical commonalities. With advances in computational power, the discovery of these factors and their risk premiums has been streamlined. Various research studies, for example, [98], have revealed factors that explain equity returns and contribute to long-term active outperformance. Prominent factors include *Price momentum*[8] [162], *Value*[9] [98], *Size*[10] [24], *Quality*[11] [132], *Low beta*[12] [113]. These factors can be employed as features to describe the RL agent's environment.

These features are not often used in the RL-based financial literature, likely due to the infrequent updates of accounting data, which occur quarterly. This temporal mismatch, as in the case of the price-to-earnings ratio, can limit the effectiveness of these factors as daily signals. However, some researchers have begun to integrate accounting data or investment factors derived from RL. Zhang and Maringer [349, 351] pioneered the use of accounting ratios in an evolutionary feature selection scheme, while Wang et al. [323] included ratios like price to earnings in state representations. More recently, Coqueret and André [78] demonstrated the effective incorporation of various investment factors, including those rooted in accounting data, into RL-based trading systems.

---

[8]Trading strategies where one buys winner stocks that performed well and sells those that did poorly in the past to generate profits over a 3-12 months holding horizon.
[9]Buying stocks with relatively low prices based on fundamental ratios and following the opposite when the high share prices are detected.
[10]Small companies perform better than big companies based on some measure, such as market capitalisation.
[11]Buying good quality companies and selling poor quality companies based on information reported in the financial statements.
[12]Buying stocks with low risk and selling stocks with high risk.

4.2.4 **Order Book.** Among the literature surveyed, we find a substantial focus on the areas of trade execution[13] and market making in financial markets. These specific applications tend to employ a unique set of features to model the environment, distinguishing them from other domains. While we explore these applications more comprehensively in subsections 7.4 and 7.5, it is beneficial to highlight some of the most commonly observed features in the associated literature. For example, Chan and Shelton [64] used order imbalance-based features[14], depth of the market[15], and time to fill a limit order, among others. Other authors used the Bid-Ask spread[16], and remaining inventory, among others, as part of their environment [248]. Numerous other features are also deployed in this context, but a detailed discussion is omitted here for brevity.

4.2.5 **Sentiment Data.** The impact of sentiment data on stock prices is well documented in the financial literature [11, 309]. Its application in RL strategies has also been demonstrated [104, 157, 172, 180, 243, 276, 342, 344]. Feuerriegel and Prendinger [104] pioneered the use of sentiment data, employing DGAP for an RL-based trading strategy. Kaur [172] followed, enriching the environment of the Q-learning agent with sentiment data. In particular, diverse sources such as Reuters News Corpus, Twitter, Google News, and Thomson Reuters News analytics have been used to create sentiment signals for RL strategies, affirming that diverse information mitigates uncertainty in the environment [180, 243, 276, 342, 344]. However, the use of sentiment/news data presents challenges. It primarily covers prominent companies, limiting its utility to lesser-known stocks. Despite sentiment's potential value, its integration into state representation remains scarce in published work.

4.2.6 **Macroeconomic.** Macroeconomic data plays a pivotal role as it encapsulates the overarching economic landscape in which companies operate. This encompasses widely recognised indicators such as interest rates, inflation, and GDP. In particular, cyclical companies are intimately tied to these macroeconomic variables due to their strong correlation with the state of the economy. Despite the importance of macroeconomic data, its incorporation into the state representation is found in a limited number of studies in the existing literature. Neuneier's work [244] is a likely pioneer in this regard, incorporating interest rates into the analysis [105]. Subsequently, Benhamou's series of studies [35–37] further integrated the slope of the US treasury yield curve, serving as a precursor risk indicator within contextual information.

4.2.7 **Others.** Earlier subsections discussed numerous environmental features implemented or potentially applicable within finance-related RL scenarios. The selection and use of these features often depends on the specific application, resulting in bespoke adaptations. This section consolidates such unique variations.

In trade execution applications, state representation often incorporates aspects such as the elapsed time, the remaining shares to order [246], and the current time [248]. Deng et al. [84] employ over 80 raw features, including non-standard ones like the volume-weighted average price (VWAP), historical bond equity correlations and credit spreads of global corporate bonds [35–38]. Other studies integrate current and cash positions into state representation [40, 165–167, 172, 284]. The financial literature and the interest of practitioners have spotlighted several underexplored datasets in RL in finance, including the following.

(1) Environmental, Social, and Governance (ESG) factors have emerged as significant themes in the investment world [126, 249] recently. The integration of ESG-based features into the RL environment may pave the way for novel research trajectories.

(2) Supply chain data, a valuable dataset in finance literature [73, 228], offers insights into the interconnected relationships among firms, including suppliers, customers, competitors, and joint ventures. These relationships, varying in transparency, result in unstructured, incomplete, and highly complex supply chain data. Integrating supply chain data into the RL environment could help capture the intricate market dynamics of individual stocks in relation to others.

4.3 Feature Selection and Feature Extraction

4.3.1 **Introduction.** As outlined in subsection 4.2, the complexity of financial markets requires many features to define an agent's environment. However, this proliferation of features inflates the dimensionality of the environment and requires intense computational resources. During the early years of Value-based applications when computation was more of a limiting factor, authors extensively discretised the state to mitigate this issue, for instance, Dempster and Romahi's work [82] where they binarised eight technical indicators, forming a bit string [153]. A significant change in the feature selection and representation framework is observed after 2016, mainly due to advances in DRL [231, 232] and the adoption of NN architectures, such as Convolutional Neural

---

[13]The agent (for example, a broker) receives and completes an order either as a buy or sell with a given quantity.
[14]*Order imbalance* is when an excess of buy or sell orders for an asset causes difficulties in matching the orders of buyers and sellers.
[15]The amount of price change until the order is filled.
[16]The amount by which the Ask price exceeds an asset's Bid price.

Networks (CNN) [182, 189], propelled by improved computational capabilities. This subsection discusses the evolving mechanisms for feature selection and extraction in financial RL applications.

4.3.2 **Genetic Algorithm as a Tool for Feature Selection.** The extensive use of Genetic Algorithms (GA) in RL applications traces back to Dempster and Romahi [82], who proposed a hybrid evolutionary RL system with GA used for optimal feature selection. Bates et al. [29] extended this framework to include order flow and order book features in addition to technical indicators. Hryshko and Downs [155] mirrored this approach but suggested the Stirling ratio as the fitness function, reducing computational demands by removing the role of the RL algorithm in determining the count of state representation features.

Chen et al. [67] proposed a Genetic Network Programming method based on SARSA to select relevant technical indicators from a candlestick chart [17], a strategy also adopted by Mabu et al. [222] within an Actor-Critic framework. Gu et al. [134] further developed Chen et al.'s work by incorporating plural subroutines. Zhang and Maringer [349] introduced GA as a pre-screening tool, feeding selected features into an RRL trading system. The model encompassed eight Boolean indicator variables, sourced primarily from technical analysis. Selection was based on an average fitness function scheme, providing an improvement over Maringer and Ramtohul [224] by reducing overfitting. Zhang and Maringer [351] later refined their approach employing a roulette wheel selection approach during the training phase. Zhu et al. [359] applied GA to choose features for a Support Vector Regression model that predicts stock trading bounds based on Q-learning.

4.3.3 **Deep Learning Based Features.** Earlier, we mentioned that the introduction of DL and DRL had significantly impacted the RL literature. From 2016 onwards, many publications in this survey have used deep network architectures. Therefore, in this section, we will highlight some key advancements instead of listing all the developments in the literature. A critical paper on current RL-based trading systems is by Deng et al. [85]. The authors presented the first comprehensive framework, namely DRRL, that uses DL as a feature extraction tool and applies the RRL framework. Moreover, the authors propose "task-aware backpropagation thought time" (task-aware BPTT) to solve the gradient vanishing problem [34].

Recurrent Neural Network and Long Short-Term Memory: The RRL and its derivative DRRL are based on Recurrent Neural Networks (RNN), a vital tool in POMDPs due to their ability to record and use past states to optimise future actions [85, 123, 272]. However, vanishing gradients in deep structures is an issue [34]. Hence, Lu [217] adopted Long Short-Term Memory (LSTM) networks [152] with dropout incorporated [128] for feature learning and capturing the underlying financial market conditions.

The use of LSTM in decision-making is advantageous due to its ability to remember features over longer steps, an essential quality for financial market trading. In addition, its memory cell retains trading actions, thereby capturing the impact of transaction costs more effectively. Bisht and Kumar [43] developed two LSTM-based systems, one functioning as an autoencoder for feature extraction and the other using the decoded features within an LSTM module for decision-making. Lin and Beling contrasted two network structures, one combining a fully connected network (FCN) and LSTM and the other based purely on FCN [212]. They concluded that the former generally outperformed the benchmarks. However, regular LSTM's assumption that information is uniformly spaced can be problematic in varying data environments. To address this, Sawhney et al. proposed a time-aware LSTM [30, 276]. Other notable applications of LSTM can be found, for example, in [68, 164, 183, 265, 313].

Gate Recurrent Units (GRU) [69] are used to distill informative characteristics from raw financial data [213, 338], as studies such as Chung et al. [71] have shown their superiority over LSTM in handling time series data.

The Advent of Convolutional Neural Networks: In the current RL financial literature, CNNs are commonly employed for feature extraction. Jiang et al. [165, 166] were among the first to apply CNN and other network architectures, specifically in the context of PM. They introduced an Ensemble of Identical Independent Evaluators, where the price movement of each asset is extracted independently. However, as the network depth increases, this CNN structure could face challenges such as vanishing gradient and gradient explosion. Additionally, this approach may fail to capture any non-linear relationships among assets. Consequently, Liang et al. [207] adopted a similar framework but replaced CNN with a Deep Residual Network [145].

Shi et al. [285] highlight a gap in RL-based PM literature, which primarily relies on price-based features, neglecting temporal information across different scales. To address this, they proposed the Ensemble of Identical Independent Inception using the Inception Network for simultaneous multiscale application [304]. Their innovative architecture integrates three parallel components: short-term and medium-term price movement features and maximum price features. This pioneering work demonstrates the value of parallel, multi-component feature extraction in network architectures.

---

[17]A candlestick chart visually represents an asset's price movements over a specific period, showing open, close, high, and low prices.

Soleymani and Paquet [293] apply an autoencoder for feature extraction and dimensionality reduction. A Graph Convolutional Network [147] is then used to capture the interactions among financial assets. Aboussalah et al. [6] implemented dual CNNs in an Actor-Critic agent, with one for time series and the other for cross-sectional features, the first attempt to combine these features. We note that this is the first study to combine time series and cross-sectional features. However, the results may exhibit survivorship bias due to stock selection. Some other publications using CNNs include [35–38, 305].

Introducing Attention Mechanisms: Several network architectures have been used as feature mechanisms across a range of applications. However, effectively estimating the temporal relationships between assets remains a challenge. To address this, the integration of an attention layer [303, 315] in the network architecture has been proposed to capture temporal and time series data. Consequently, variations in attention have garnered considerable interest within QF in the RL framework.

Lei et al. [197] employed a GRU to model long-term relationships, complemented by a temporal attention mechanism [257] to discern stock interdependencies. Weng et al. [333] proposed the use of separable convolution [70] and three-dimensional gated attention networks for feature extraction. Zhang et al. [353] took a unique approach to model asset interdependencies, introducing a two-module network; the first was an LSTM network capturing sequential asset patterns, while the second employed dilated causal convolutions [250] and correlational convolutions for asset correlations. The output of both modules is concatenated for comprehensive analysis. Further studies using attention mechanisms are presented in [323, 326, 340].

Other Deep Learning-Based Feature Mechanisms: As discussed in Section 1, a significant challenge for RL is the need for extensive financial datasets. Addressing this, Yu et al. [345] proposed a data augmentation mechanism within their model-based Actor-Critic framework, using a Generative Adversarial Network (GAN) [127]. Specifically, they adopted a Recurrent GAN (RGAN) [95] variant, where the generator and discriminator are replaced by RNNs. They employed an RGAN to produce asset closing price returns at a frequency higher than the RL agent, which was then downsampled for synthetic High-Low-Close data. Moreover, AbdelKawy et al. [2] used a Deep Belief Network (DBN) [150] for feature extraction and dimensionality reduction.

4.3.4 **Other Feature Mechanisms.** Yuan et al. [347] use minute-candle data to train their RL model for daily stock trading, proposing a skewness and kurtosis-based selection process for trading stocks. Several works use OCHLV data in state representation without theoretical or practical explanations [165, 166]. However, Weng et al. [333] provide some clarity by revealing through XGBoost [62] that the closing price is the most important feature of all. Clustering methods for simplifying state representation are used in studies such as Fengqian and Chao [102], who use Japanese candlesticks for denoising. Other examples include [63, 175].

Various transformations and techniques are proposed to reduce the inherent randomness and uncertainty of financial data. Deng et al. [85] suggest a fuzzy transformation [210], while others apply sparse coding [84], and Carta et al. [57] use the Gramian Angular Field (GAF) [318] to transform time series into images. Lu et al. [218] recommend the Continuous Wavelet Transform (CWT) [268] to derive frequency-domain representations, which are seen as more suitable for time-varying signals, which they integrate with a CNN and LSTM to extract frequency- and time-domain features. Dimensionality reduction is also used to retain salient features [171, 205, 266]. Carapuço et al. [56] adhere to the data preprocessing guidelines established by LeCun et al. [190].

5 Action Modelling and Reward Functions in Finance

5.1 Introduction

In this section, we emphasise two pivotal facets of the RL framework: the action space and the reward functions, which are both contingent on the specific application. Actions, denoted as A_t, are the input of the agent into the environment at time t. Typically, in the QF-related RL literature, agents are assumed to have no impact on the environment, and actions may be discrete, such as buying or selling a stock, or continuous, such as portfolio weights at time t. Table 3 summarises the action and reward functions used in the fifteen most cited papers in this survey, which are also chronologically ordered and will be discussed further in this section.

The agent's objective, encapsulated by the reward signal $R_t \in \mathcal{R} \subset \mathbb{R}$, optimises the expected cumulative reward over a sequence of steps, underscoring the preference for long-term gains. The nature of these rewards is application-dependent, commonly aligned with financial objectives such as maximising the SR or cumulative wealth. This objective corresponds to the return G_t, defined as the discounted sum of sequential rewards $R_{t+1}, R_{t+2}, R_{t+3}...$, formalised as:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{8}$$

Publication	Asset	# of Act.	Cont.	Act. Func.	P. B.	Perf. Ratios	Util. Based	Other	Tr. Costs
Neuneier [244]	FX/Equities	2			x				x
Moody and Wu [235, 236, 238]	FX/Equities	2		Tanh/Softmax	x	x			x
Dempster et al. [81]	FX	2, 3			x				x
Nevmyvaka et al. [246]	Equities	11						x	x
Dempster and Leemans [83]	FX	2		Tanh			x		x
Lee et al. [194]	Equities	2-7			x			x	
Deng et al. [85]	Com./ Eq. Index	3		Tanh	x				x
Jiang and Liang [165, 166]	Crypto	12	x	Softmax	x				x
Almahdi and Yang [8]	Stock ETF's	5	x	Logsig/Softmax		x			x
Liang et al. [207]	Equities	5	x		x				x
Xiong et al. [339]	Equities	3			x				
Jeong and Kim [163]	Equity Index	3			x				

Table 3. Action and Reward types for the most cited papers in this survey. Asset = Asset Class, # of Act. = Number of Actions, Cont. = Continuous, Act. Func. = Activation Function, P. B. = Profit Based, Perf. Ratios = Performance Ratios, Util. Based = Utility-Based, Tr. Costs = Transaction Costs.

where $\gamma \in [0, 1)$ represents the discount rate determining the agent's foresight. With $\gamma = 0$, the agent prioritises immediate rewards, whereas γ close to 1 emphasises future rewards significantly. Notably, for infinite reward sequences, $\gamma \in (0, 1)$ ensures equation (8) remains finite and mathematically well-defined, which is crucial for continuing tasks.

Drawing parallels with finance, this approach conceptually resembles the discounting of the price of bonds, where γ is the period discount factor and R_i represents the coupon plus the face value at time i [96]. In finite-time horizons, such as modelling the expected return until a terminal event such as bankruptcy, using a discount factor $\gamma < 1$ is consistent with financial models such as the Gordon Growth Model [129] and bond pricing. These models inherently apply a discount factor of less than 1 to reflect the time value of money[18], ensuring that the present value of future cash flows is properly calculated. This method emphasises the financial principle that future rewards are typically less valuable than immediate rewards, even within a finite horizon. However, setting $\gamma = 1$ can be appropriate and meaningful in specific bounded scenarios where future rewards are equally critical.

5.2 Action

Action spaces in RL are typically either continuous or discrete, usually reflected in the agent's framework. While Value-based methods exclusively accommodate discrete action spaces, alternative methodologies exhibit greater flexibility, rendering them more realistic for financial trading applications. However, the utility of Value-based methods should not be discounted, as the following discussion in subsection 7.2 will reveal.

Most Value-based RL studies in QF allow for two to three actions (Buy/Hold/Sell) for a single stock. Exceptions exist, such as Sherstov and Stone's [284] model encompassing 1801 actions and Dempster and Romahi's [82] bitstring-coded state representation. Similarly, models with multiple tradable assets, such as Kaur's [172] model with three assets, demand a wider range of actions, thereby escalating computational time due to an expanded exploration scope. Jangmin et al. [161] and Park et al. [255] suggest alternatives to these limitations by using multiple local traders and creating an ETF portfolio, respectively. However, even these sophisticated models employ discrete action spaces. Recently, research has focused on addressing this limitation in DRL, such as Kalashnikov et al.'s [169] QT-Opt method, employing the Cross-Entropy Method (CEM) for Q-function optimisation.

Despite such advances, Value-based methods still lag behind Policy-based and Actor-Critic methods in terms of efficiency, especially when dealing with continuous action spaces. This ability is crucial in financial markets, where action spaces are naturally continuous. Consequently, using RL algorithms such as SARSA and Q-learning for continuous actions can lead to suboptimal solutions or even failure to converge, as observed in some simple cases [23].

In the RRL framework, the action space can be represented as an NN where the state inputs translate into actions [125]. In single-asset trading systems, typically a single output neuron (activated by a tanh function) is used, the output of which models long and short positions. Such output can be discretised into buy/hold/sell signals with defined thresholds. For multi-asset scenarios, such as in Jiang et al. [165, 166], the softmax activation function is used to ensure the sum of portfolio weights equals one. Variations of this approach are seen in the literature, where there is a distinction between long and short portfolios [323, 326].

---

[18]There are cases where the time value of money principle is violated, such as with an inverted yield curve, where short-term rates exceed long-term rates, making future cash flows more valuable. In RL, this could correspond to $\gamma > 1$, where future rewards are prioritised. This is uncommon in RL because it can lead to instability in learning algorithms and unbounded returns, complicating optimisation.

However, Coqueret and André [78] suggest being cautious against the indiscriminate use of the softmax function due to its potential violation of the budget constraint. Instead, they advocate for Dirichlet distributions, apt for long-only portfolios due to inherent budget adherence. In comparison, Actor-Critic methods align with Policy-based in their action set, albeit with some unique implementations. For example, Li et al. [200] harnessed agent action to improve SVM price prediction, while Bekiros [31] proposed a trading system that intertwines agent action with a fuzzy system. Qiu et al. [266] further innovated by introducing Quantum Price Levels (QPL) to model the action space.

5.3 Rewards

The selection of the reward function in RL is crucial, as it indicates the objective to be optimised [302]. The flexibility inherent in RL enables the creation of a wide range of reward functions tailored to specific tasks or problems. Tan et al. [307] demonstrate this by introducing a range of task-specific reward functions.

Several studies contrast different reward functions and frameworks, such as Du et al. [90], who analyse various reward functions, including cumulative wealth, utility, and SR in the context of Q-learning and RRL, highlighting their interconnection. Although utility-based rewards offer a simple way to infuse risk sensitivity into the objective function, Mihatsch and Neuneier [230] warn of inherent drawbacks. They suggest a risk-sensitive RL approach using transformed temporal differences during learning and propose a novel algorithmic variation of temporal difference and Q-learning. Lucarelli et al. [219] indicate that Sharpe Ratio (SR) reward functions outperform profit-based reward functions in DQN contexts. Finally, immediate rewards tend to be the primary focus in the literature, because not selecting the immediate reward will incur additional transaction costs in quantitative trading.

5.3.1 ***Profit and Performance-Based Rewards.*** Most academic research gravitates towards profit-based rewards [64, 93, 172, 276, 286, 294]. Risk Performance Measures (RPM), such as SR, comprise the second most common focus [40, 77, 311, 323]). An intriguing variation emerges in several studies [116, 205, 213, 225, 350], notably the use of Differential SR (DSR) as introduced and developed further by Moody and colleagues [235–238]. DSR, an online variant of standard SR, offers several benefits, including enabling efficient online optimisation and weighing more recent returns [235]. The generic derivation of the DSR suggests the feasibility of online versions for other performance measures [8, 9, 39, 235]. Alternative measures have been used in various contexts, for instance, Carapuço et al. [56] and Wu et al. [338] use the Sortino ratio[19]. The diversity of these measures highlights the potential to explore broader reward functions, such as incorporating higher moments into the classic SR [262].

Several studies have proposed unique risk-averse reward functions. For example, Jin and El-Saawy [167] define a reward function that incorporates both return and standard deviation, a structure echoed in Li and Chan [199], who instead use GARCH [46] volatility. Si et al. [286] and Bisht and Kumar [43] incorporate downside volatility, reflecting a more practitioner-orientated focus. Such risk-aware reward functions better capture the priorities of portfolio managers who balance risk and return.

5.3.2 ***Sparse and Conditional Rewards.*** Sparse rewards present a realistic model of many environments, typically characterising a reward function with predominantly zero values and few positive state-action pairs. Although sparse rewards offer a detailed description of various environments, they can pose challenges in identifying optimal policies due to the infrequent positive feedback [263]. Another form of reward function similar to sparse rewards is a conditional reward, in which the reward depends on specific conditions in the state or action. Usually, we observe sparse rewards or conditional-based rewards in cases where the action space is discrete, such as BUY, SELL, and HOLD in the Value-based framework.

Neuneier [244] first used conditional rewards within a foreign exchange context. The state vector s_t comprises the current exchange rate x_t, portfolio wealth c_t, and a binary variable b_t that indicates the investment in DM (Deutsch Mark) or USD at time t. The reward function, then defined at $t + 1$ given the action, integrates the transaction costs into the model, thus reducing the reward for a DM-USD transition. Numerous studies have expanded on this framework. Jia et al. [164] designed a sparse reward incorporating market price, volume, transaction costs, and the current amount. Kim and Kim [174] used conditional rewards for risk management in statistical arbitrage. Lucarelli et al. [219] introduced a hybrid reward function based on discretised and thresholded SR values, leading to specific Sharpe Ratio (SR) ranges that increase or decrease the reward. Lin and Beling [212] employed a sparse reward for optimal trade execution based on performance relative to Time-Weighted Average Price (TWAP). Similarly, Carta et al. [57] incorporated conditional rewards based on market prices, introducing a zero-reward state for idle agents. Taghian et al. [305] applied a similar framework in their trading systems.

---

[19]The Sortino ratio is similar to the SR but penalises those returns below a given investor threshold [295].

In this context, sparse and conditional rewards generally serve to control extreme reward values [174] or as a mechanism to alter trading signals [244], thus shaping the learning trajectory towards optimal policy discovery.

5.3.3 **Utility-Based Rewards.** Utility-based rewards have an established presence in the RL finance literature, initially presented by Dempster and Leemans [83], where they optimised an RL-based trading system according to a utility function. This utility function resembles the principles of classical portfolio construction, with the aim of generating higher returns for a given level of risk. The maximisation of the utility depends on several parameters, five in total. Subsequent studies expanded on these principles. Hens and Wöhrmann [149] applied the power utility function for dynamic allocation between bonds and equity indices. In a shift towards high-frequency market making, Lim and Gorse [209] used Constant Absolute Risk Aversion (CARA) to gauge an agent's attitude towards gains or losses on partial inventory sales.

García-Galicia et al. [120] proposed an Actor-Critic framework implementing a utility-based reward function for continuous-time PM. Bao and Liu [25], as well as Bao [26], employed the difference between two successive utility functions as a reward. Risk-averse agents and ES/CVAR, the average loss exceeding the worst $q\%$ of cases [269], are also adopted as a risk-adjusted performance measure [7, 51]. Finally, in market-making and trade execution literature, utility-based rewards are common [177, 248, 291].

5.3.4 **Composite Reward Signals.** Several studies employed multiple reward functions to optimise agent behaviour. An intuitive and straightforward approach is the linear combination of disparate reward functions. This practice can address the issue of reward noise, which is particularly prevalent in profit-based signals. When rewards contain high noise levels, they can induce substantial variation in returns and thus obstruct the training process. Furthermore, it becomes necessary to normalise the rewards when they span different scales. Employing composite rewards offers a natural avenue for integrating a multi-objective structure within our RL agent, although caution must be exercised with potentially conflicting rewards.

This type of reward system was first developed within Chan and Shelton's market-making framework [64]. They identified three distinct reward signals—changes in profit and inventory levels—and integrated them linearly. The training process determined the weights assigned to each signal. More recently, Zarkias et al. [348] and Tsantekidis et al. [313] proposed a combined reward signal based on three individual signals: profit and loss, price trailing (the extent to which the asset price is tracked) and fees. They found that their approach functioned as a potent regulariser, enhancing performance in comparison to the unadorned P&L reward. In Zhang et al. [353], the final reward was a linear combination of the risk-sensitive and the cost-sensitive reward mirroring Jin and El-Saawy [167]). Lastly, Koratamaddi et al. [180] produced a final reward encompassing a change in portfolio value from the previous day and market sentiment, marking the first instance of market sentiment being integrated into the reward function.

5.3.5 **Others.** Yang et al. [343] present a reward function incorporating the turbulence index [181]. In their methodology, if the turbulence index exceeds a particular threshold, agents act to forestall potential adverse shifts in portfolio value by liquidating all held stocks. Meanwhile, Wang et al. [328] have adopted a reward-shaping approach. Here, a penalty is appended to a profit-based reward if the agent's actions deviate from those of an expert, baseline, or any other valid comparator, where the expert policy serves as a threshold. The idea to integrate a dynamic strategy, such as an oracle, offers an intriguing prospect for further investigation.

6 Enhancing RL in Quantitative Finance with Advanced ML Techniques

6.1 Introduction

In this section, we explore salient ML themes evident in the current RL literature, highlighting their relevance and potential to significantly augment RL applications in QF. RL models can achieve enhanced performance, robustness, and adaptability in financial markets by integrating ensemble methods, transfer learning, imitation learning, policy distillation, and Multi-Agent Systems (MAS). These methods tackle challenges such as data scarcity and complex decision-making environments.

6.2 Ensemble approaches

Ensemble trading systems are conspicuously underrepresented in the reviewed literature. Notable studies include [57, 183, 196, 343]. This section highlights promising ideas for integrating ensemble methods to enhance RL-based trading systems in QF.

Yang et al. [343] constructed an ensemble consisting of the PPO, A2C, and DDPG algorithms. Specifically, they trained these three agents with an identical growing window of n months. Subsequently, the agent exhibiting the best performance (evaluated using the SR) during the validation test was employed to forecast the trade for the next quarter. This strategy was inspired by their observation that different agents excel under various market conditions. Notwithstanding its simplicity, this approach could be improved by implementing a forward-looking measure for agent selection or adopting multiple performance criteria. Furthermore,

understanding why specific agents outperform others under certain market conditions warrants further exploration. In their study, Leem and Kim [196] introduced a DQN ensemble trading system built on three trading experts, each associated with an individual action. A unique reward value was assigned to each action under specified conditions to drive specific investment behaviour. Carta et al. [57] developed a reward-based classifier to process outputs or features extracted from a preceding layer using CNN features. This process is based on stacking [337], which is an ensemble method. Kumar et al. [183] adopted a parallel framework, with their ensemble method using various feature extraction mechanisms such as CNNs and LSTMs. The resulting agents were then combined via a trio of methods: highest probability, average, and majority voting.

Ensemble methods enhance accuracy [140], generalisability [264], and robustness to outliers [251], making them ideal for RL-based trading systems in QF. However, they also decrease transparency, increase complexity, and increase computational costs [264, 358]. This trade-off between interpretability and benefits requires further investigation.

The construction of reward-based ensembles represents an intriguing frontier for future research. By facilitating the concurrent optimisation of multiple reward functions, this strategy can adapt to the intricate landscapes of RL environments found in QF applications, where performance is evaluated using distinct reward functions. This approach bears conceptual similarities with the multi-agent RL approach, which also centres on the optimisation of multiple agents, each guided by a unique reward function.

6.3 Transfer learning

In the past decade, ML has made significant strides in classification and regression tasks. However, these tasks often assume that training and testing datasets share the same distribution. When distributions shift, models typically need rebuilding with new data, which is inefficient. Transfer learning addresses this by using pre-trained models as starting points. In finance, transfer learning is crucial for improving RL agent performance by leveraging pre-trained models to address data scarcity as comprehensively discussed in seminal surveys by Pan et al. [253] and Zhuang et al. [356]. Transfer learning has profound implications in the realm of finance due to its potential to facilitate the generalisation of experience—a subject we discuss in this section. However, our review of the RL-based literature indicates that its adoption has been relatively limited [47, 48, 157, 163, 332].

Jeong et al. [163] applied DQN-based transfer learning to stock trading, focusing on equity stock indices. They used two techniques to filter the stocks: the correlation between two stocks and an NN to determine the relevance of the data pattern. They categorised the stock data into three groups based on their association with the underlying stock index (measured by lower mean square error (MSE) for NN). Half of the stocks with high correlation and low MSE were used to pre-train a model with shared weights, which was subsequently used to train the main model on the index. They argued that transfer learning compensates for financial data scarcity by using index constituents as proxies, although the opposite can also be anticipated since indices generally have more historical data. Motivated by the sample efficiency and transferability of model-based RL [263], Wei et al. [332] introduced a model-based RL agent for electronic trading, employing an autoencoder to extract latent LOB features and combining an RNN with a Mixture Density Network (MDN) [42]. This represents the first application of model-based RL in transfer learning. However, the challenges in modelling financial market environments, particularly state transition modelling, might explain the scarcity of such applications in the literature.

Huang and Tanaka [157] introduced transfer learning to avoid training multiple DQN agents from scratch. They trained a DQN agent on Apple Inc. prices before training others. Although this approach might help understand stock trends, it overlooks that Apple's success may not represent typical stock patterns, particularly regarding default risk, potentially leading to negative transfer [260]. Borrageiro et al. [47] used transfer learning with Radial Basis Functions for feature representation in an RRL framework. Later, they extended this to digital assets, using Echo State Networks [160] to represent feature space and then transferred this knowledge to RRL agents for Bitcoin trading.

Transfer learning, with its ability to leverage pre-existing knowledge, can significantly boost RL agents' performance in finance. However, some studies could have benefitted from a more strategic application of transfer learning. For instance, rather than using a single stock as in Huang and Tanaka [157], a sector-based approach might be more effective. Stocks within a sector tend to share similarities, and sector indices often provide more historical data. An optimal strategy might involve training initial models on sector indices and then applying transfer learning to individual stocks within the sector.

6.4 Policy distillation

In subsection 6.3, we examined transfer learning in QF. A related concept, Policy Distillation (PD), passes knowledge from a larger model to a smaller one for the same task, hence 'distillation' [151]. Despite its potential to improve the efficiency of the RL agent and mitigate overfitting, PD has been underused in finance, with notable applications by Tsantekidis et al. [314] and Fang et al. [101]. PD can help mitigate overfitting and streamline the deployment of RL agents by transferring knowledge from complex models to simpler and more efficient ones.

Tsantekidis et al. [314] proposed a PPO-based agent for foreign exchange trading using PD. They trained "teacher" agents across different currency groups, creating a diversified learning pool. The "students", learning a specific data subset, then used PD. To handle financial data noise, the authors employed diversified teacher ensembles. Similarly, Fang et al. [101] used PD under PPO for order execution [274]. The aim was to bridge the gap between imperfect market information and optimal order execution policy. Here, the teacher had access to perfect information, aiming to learn the optimal policy while the student mirrored the teacher's behaviour. Despite a few authors applying PD, its potential use cases mirror transfer learning's breadth. A potential comparison or integration with an agent using transfer learning could be beneficial, as PD can fill in missing values.

6.5 Imitation learning

Imitation Learning (IL), also known as learning from demonstrations, is a machine learning technique that allows an agent to learn a task by imitating expert behaviour. In the QF context, IL has gained attention due to its potential to replicate successful investment strategies. In RL, IL enables agents to mimic expert investors, thereby improving performance and enhancing the accuracy of financial models. The application of IL in QF was explored by Ding et al. [87], and later by Yu et al. [345], Liu et al. [213], and Kim et al. [175]. Finally, the current literature classifies IL methods into two categories: passive and active [1, 271].

Ding et al. [87] developed three investor imitation frameworks—Oracle, Collaborator, and Public Investor—each requiring a distinct reward structure. The Oracle framework leveraged evaluation scores to maximise historical rewards, but it lacked foresight. The Collaborator and Public Investor frameworks instead guided agent learning through a similarity score function and revenue curve approximation, respectively. Yu et al. [345] deployed an active IL algorithm within a model-based RL framework, employing one-step greedy actions to optimise immediate returns. This methodology involved slight weight perturbations of the actor based on the log loss between the expert's and the actor's actions. Liu et al. [213] created a trading-expert algorithm that operated long and short positions at optimal prices. They estimated the gap between agent and expert actions using Ross and Bagnell's method [270], followed by applying a modified policy gradient to the actor. Meanwhile, Kim et al. [175] used behaviour cloning in statistical arbitrage, facilitating learning from a prophetic expert.

IL has significant potential in QF, for example, enabling investors to replicate successful mutual fund performances. However, correct feature selection remains crucial. For instance, imitating a multifactor fund necessitates prioritising features such as value, momentum, low beta, size, and quality. Integrating IL with transfer learning offers exciting prospects for advancing finance models and addresses challenges with limited expert demonstrations. Using transfer learning, the knowledge gained from expert investors can be effectively transferred to different financial markets or scenarios, enabling faster adaptation and improved decision-making.

6.6 Multi-agent applications of RL in finance

A novel RL framework for Multi-Agent Systems (MAS) in QF was first proposed by Lee and Jagmin [191] and refined in subsequent studies [192–194], demonstrating how cooperative agent environments can optimise stock trading solutions. This framework structures the problem so that agents work together towards an optimal solution for stock trading. Lee et al. [194] proposed two types of agents: the signal agent and the order agent, each further divided into two agents, forming four agents:

(1) **Buy signal**: This agent assesses recent price movement to gauge potential price rise. If no buy signal, the episode ends.
(2) **Buy order**: Activated by a buy signal, this agent sets the limit price for buying based on intraday prices.
(3) **Sell signal**: This agent monitors profit and loss to decide when to close a position.
(4) **Sell order**: Activated by a sell signal, sets the limit price for selling based on intraday prices.

This approach recognises that buying and selling stocks involve unique considerations: buyers assess potential price rises or falls, while sellers consider price fluctuations and potential profits or losses – leading to distinct state representations for each agent. The buy agent's state is informed by long-term price changes, while the sell agent priorities the current profit or loss. Each agent is optimised through Q-learning, and it has been shown that cooperative agents can solve complex trading problems [194]

Lee et al. [195] proposed a Multi-agent Portfolio Management System (MAPS) to capture the dynamic nature of financial markets. Each MAPS agent operates independently, managing its portfolio. The challenge of identifying the best joint policy is addressed by training each agent separately with DQN, sharing the state space and experience replay. The objective is to maximise rewards while ensuring diverse actions from each agent. The loss function is a weighted average of global loss and local losses (individual agent losses), reflecting the goal of maximising rewards and promoting diversity among agents. This inspires further exploration, including investigating changes in agents' utility functions relative to risk aversion, adjusting agent weighting based on market conditions, and applying transfer learning to refine approaches by limiting the information each agent receives.

AbdelKawy et al. [2] introduced a synchronous-based multi-agent DQN/DDPG trading model. Each stock was initially trained with a DQN agent using a ResNet [145] network architecture, followed by transfer learning applied to each trading module to reduce computational time. Shavandi and Khedmati [282] developed the first multi-frame DQN multi-agent system with three agents operating at different time frequencies (1 hour, 15 min, and 5 min). The lowest-frequency agent's output serves as input for higher-frequency agents, reflecting the fractal nature of financial markets [261]. While the combined agent outperformed individual components, this was based on a single foreign exchange pair (EUR/USD), warranting cautious interpretation. Other studies have also explored various aspects of multi-agent learning in financial markets, including work by Casgrain et al. [61].

6.7 Model Interpretability

Wang et al. [323] developed an interpretable investment strategy to answer "What kind of stocks are selected as winners?" They used sensitivity analysis methods [3, 320, 322] to examine input factors' influence on final outputs. Their results showed that the algorithm selects stocks with favourable scores in long-term growth, low volatility, high intrinsic value, and recent undervaluation. Cong et al. [74] used a similar framework with polynomial sensitivity analysis. Despite the importance of model interpretability, explainability, and causality in financial research, the current RL literature has not addressed these themes extensively. Only Wang et al. [323] and Cong et al. [74] consider model interpretability, while Wang et al. [326] highlight the benefits of identifying causal structures among stocks using the PC [292]. They then employ this output as an input for graph convolution networks.

For RL in QF, model interpretability, explainability, and causality are crucial to building trust and ensuring that investment strategies are transparent and comprehensible to stakeholders. Investment managers must articulate the performance of the strategy to stakeholders. Without explanatory capacity, the adoption of the strategy may be limited. Incorporating causal inference methods to interpret relationships between input features and RL actions could be a promising research avenue [259].

7 RL Applications in Modern Finance – Promising Areas

7.1 Introduction

This section reviews common modern finance applications and RL-based solutions and examines RL framework customisation for classic QF issues. The literature illustrates RL's adaptability to the financial domain as can be seen in the range of popular applications listed in Table 4. In the following, we provide an examination of the salient ideas and potential extensions within the current literature on this topic and provide future directions in these areas. In addition, we delineate the pivotal concepts and thematic trends that have emerged. Although we acknowledge the extensive literature on PM, we have reserved some works for inclusion in different subsections as they emphasise distinct methods or themes relevant to PM. For example, the papers by Ding et al. [87] and Yu et al. [345], which use IL methods within the PM environment, are already discussed in subsection 6.5.

7.2 Portfolio Management Under the RL Framework

Markowitz's seminal work on portfolio theory [226] remains a foundational text in investment management, inspiring numerous contemporary Portfolio Management (PM) strategies. However, its utility relies on assumptions that, upon scrutiny, may seem overly simplistic. Among these assumptions are the rational behaviour of investors and the efficient operation of markets, both of which are often contradicted by empirical evidence. In particular, investors frequently exhibit herd-like behaviour towards speculative investments, culminating in asset bubbles or market crashes due to large-scale financial asset sell-offs [27, 221]. Against this backdrop, RL-based PM frameworks, devoid of these stringent assumptions, have attracted considerable research interest. Their promising results indicate the potential for superior performance compared to the classical Markowitz framework.

Initial applications of RL in QF, particularly PM, were scarce. Jagmin et al. [161] presented an RL agent using a Value-based-framework capable of dynamically allocating strategies across risky assets. The agent used four distinct trading patterns to predict share prices, followed by a meta-policy to distribute investments among these patterns optimally. However, the Value-based model's

Publication	Applications				
	Trading Systems	**Portfolio Management**	**Asset Allocation**	**End-to-End**	**Trade Execution**
Neuneier [244]	x		x		
Moody and Wu [235]	x				
Moody et al. [236]					
Moody and Safell [238]					
Dempster et al. [81]	x				
Nevmyvaka et al. [246]					x
Dempster and Leemans [83]	x			x	
Lee et al. [194]		x	x		
Deng et al. [85]	x				
Jiang and Liang [165]		x	x		
Jiang and Liang [166]					
Almahdi and Yang [8]		x	x		
Liang et al. [207]		x	x		
Xiong et al. [339]	x				
Jeong and Kim [163]	x				

Table 4. Key application categories for the top 15 cited publications in our list.

fundamental drawback was its dependence on a discrete action space, which misaligned with portfolio managers' preference for continuous weight allocation. Park et al. [255] applied the Value-based framework to an Exchange-Traded Funds (ETFs) portfolio, highlighting the potential shortcomings of a discrete action space and suggesting potential solutions. However, it should be noted that a discrete action space still lacks the necessary realism.

Arguably, the first attempt to develop a more realistic PM framework was made by Jiang et al. [165, 166]. They designed a model to manage a diverse cryptocurrency portfolio, with the agent aiming to maximise the average cumulative return R across 12 cryptocurrencies (with Bitcoin as a cash proxy) at a given time t. Despite its pioneering nature, Jiang et al.'s work leaves room for enhancements, including the incorporation of constraints in the optimisation. In response, building upon their earlier work [8], Almahdi and Yang [9] proposed the first constraint optimisation problem within the RRL framework. Their application of Particle Swarm Optimisation (PSO) [173] to incorporate cardinality, quantity, and round lot constraints [206] transformed the unconstrained RRL case into a constrained optimisation problem, effectively combining the RL and PSO frameworks. However, directly incorporating constraints within the RL framework might offer additional benefits. In a further advancement, Wang [324] and Wang and Zhou [325] proposed a continuous-time mean-variance portfolio framework with entropy regularisation to balance exploration and exploitation. Named Exploratory Mean-Variance (EMV), this framework includes policy evaluation, policy improvement, and a self-correcting mechanism for Lagrange multipliers. By avoiding deep structures, the authors sought to bypass issues related to low interpretability, extensive hyperparameter tuning, and unstable performance.

7.3 Financial Options-Based RL Applications

The evolution of option[20] pricing methodologies, stemming from seminal works by Black and Scholes [45] and Merton [229], has spawned extensive literature. These foundational papers, collectively known as the Black-Scholes-Merton (BSM) model[21], assume complete markets [14], where every financial instrument can be replicated without transaction costs. However, market frictions, such as transaction costs and liquidity constraints, violate this hypothesis. Furthermore, the BSM model presumes constant volatility, often an inaccurate assumption. RL, unbound by these simplified assumptions, provides a robust alternative.

The hedging process typically involves trading in the underlying asset of the option, such as stocks, as dictated by the first derivative of the option pricing formula [45, 229]. The majority of works in this domain explore optimal hedging amidst transaction costs in markets that do not adhere to the completeness principle, and there are differences between the European and American options. While the classic European option pricing employs the BSM model, the American option pricing requires a more intricate approach, relying on the Least-Squares Monte Carlo (LSMC) algorithm [216].

Li et al. [201] pioneered the pricing of American options using RL, using the Least-Squares Policy Iteration algorithm (LSPI) [184] and a variant of Q-learning, namely Fitted Q-learning [130], for learning exercise policies for American options. These methods were demonstrated to outperform the LSMC benchmark for both simulated and real data. Subsequently, Dubrov [92]

---

[20]An *option* is a financial contract where the owner has the right, but not the obligation to exercise, to buy or sell the reference instrument at a predefined strike.

[21]The BSM model estimates option pricing by considering factors like asset price and volatility, estimating the option premium based on contract execution probability.

expanded this framework to price Convertible bonds—corporate bonds that can be converted into the issuer's equity. Furthermore, a non-RL model using Random Forests was proposed, outperforming the RL-based benchmarks (LSPI and Fitted Q-learning) and the standard LSMC. However, no recent studies have applied RL to American option pricing.

Buehler et al. [51] applied RL to optimal hedging with transaction costs, testing their model on vanilla and barrier options. Similarly, Vittori et al. [317] applied Trust Region Volatility Optimisation [44], incorporating features such as Strike, Call price, and Delta hedge[22], as well as the preceding action in their environment. Conversely, Kolm and Ritter [177] omitted the Delta hedge, arguing that it is a non-linear function of state variables. Therefore, the agent should be able to identify such patterns. Not including such functional forms avoids relying on BSM assumptions. Du et al. [91] proposed a similar strategy with Deep Q-learning and Pop-Art [144]. Halperin [138] developed a discrete option pricing model using Q-learning, aiming to determine both the option price and the hedging strategy, without considering transaction costs. Cannelli et al. [55] demonstrated that a risk-averse contextual k-armed bandit is superior to DQL in sample-efficiency and hedging error reduction under the BSM framework. Lastly, Cao et al. [54] redefined the objective function of Almgren and Criss [10] (AC)[23], defining it as follows:

$$Y(t) = \mathbb{E}(C_t) + c\sqrt{\mathbb{E}(C_t^2) - \mathbb{E}(C_t)^2}, \tag{9}$$

where c is a constant, and C_t represents the total hedging cost from time t onwards. The objective is to minimise $Y(0)$. They subsequently fitted two separate Q-functions to track the expected value of the transaction cost $\mathbb{E}(C_t)$ and the expected value of the square of the cost $\mathbb{E}(C_t^2)$. As noted by the authors, this methodology offers various benefits over the baseline, such as the broader range of objective functions and the learning algorithm that supports continuous state and action space.

These applications represent intriguing attempts into option hedging and further testify to the adaptability of the RL framework. A compelling extension would be to apply such methodologies to more complex instruments, such as exotic options.

7.4 Order Execution

In the trading ecosystem, an asset manager, acting as a client, delegates a portfolio of trades, typically to a broker. This broker executes these trades, either selling or buying a predefined volume, with the central objective being transaction cost minimisation through strategic distribution across varied timeframes. Cartea et al. [58] highlight a key challenge - balancing market impact mitigation and price risk. Much of the existing RL literature in QF assumes negligible market impact, given the relatively small position size. Classic financial references such as Bertsimas and Lo [41] and AC offer valuable insight, but their assumptions are often too strict and usually ignore the dynamic interplay that defines financial markets, offering a relatively static framework.

The liquidity of the underlying stocks considerably influences the trading process, making the trading of less liquid stocks more difficult because of potentially higher transaction costs. The market impact of large block trades represents another substantial challenge. The current RL literature often presupposes immediate execution of trading positions, an assumption that may not accurately reflect real-world scenarios. Only a few researchers, such as Wang et al. [327], incorporate this aspect into their RL frameworks, evidenced by their inclusion of an execution module in a Hierarchical RL-based framework.

The large-scale implementation of RL in trade execution was pioneered by Nevmyvaka et al. [246], who used a modified Q-learning agent with 1.5 years of millisecond LOB data from NASDAQ. They classified state representation variables into private and market variables, a tactic subsequently adopted [212] and expanded by incorporating a risk-averse framework [283] and volume/spread attributes [148]. Bao and Liu [25] proposed the first multi-agent application of RL to optimal liquidation. They expanded the AC framework to the multi-agent environment, examining the interaction of cooperative and competitive behaviours and their implications for the overall system. Later, Bao [26] addressed the issue of fairness [240] in the multi-agent framework, exploring inequities that arise when differential execution strategies are applied to trades of the same assets with varying quantities and time horizons. Bao suggested adjusting each agent's reward by referencing it against the classic Gini or Bonferroni indices [334]. Despite their results being based on simulated data and a limited number of agents, the concept of fairness remains deeply relevant, not just to trade execution but to various facets of the financial world as well which remains as an open research area. Fang et al. [101] introduced a PPO-based framework with PD to handle imperfect information, integrating a quadratic penalty for market impact, following Ning et al. [248] and Lin and Beling [212]. Other notable studies include [79, 170, 248].

---

[22]Delta hedging in options is a strategy that aims to reduce the directional risk associated with the underlying asset's price movements.
[23]Almgren and Criss [10] proposed a quadratic utility as an objective function for the optimal execution problem. This transforms the optimisation problem into a mean-variance problem, with the expected hedging cost and the variance of the hedging cost being the two key components of the objective function.

Overall, although the AC model offers a convenient closed-form solution for order execution, its critical assumptions can lead to suboptimal outcomes when applied to real-world scenarios. In contrast, RL-based methodologies provide a more dynamic and flexible framework for trade execution, allowing for the adaptation to market conditions in real-time. This adaptability is crucial for managing the complexities of market impact and transaction costs, making RL a potentially more effective approach in the evolving landscape of order execution.

7.5 Market-Making

A market maker's function is to ensure liquidity for buyers and sellers by persistently offering bid and ask quotes alongside the respective market sizes. The profit avenue for the market maker is the spread, the difference between the bid and the ask. One substantial risk market makers face is the inventory risk, which arises when the current inventory remains unsold. This can occur when informed traders engage with market makers prior to a price drop, resulting in losses. Therefore, maintaining minimal inventory is a key goal for market makers, achievable either by initiating trades to reduce the inventory at a cost or by skewing asset prices to attract trades offsetting the inventory.

With the increase in trading frequencies over recent decades, human processing of the resulting data flood has become nearly impossible. This underscores the critical importance of electronic Limit Order Book (LOB). Given this, applications in market-making have the potential to attract significant interest. However, most current applications are based on simulated or artificial data, and few use real LOB data. However, there is a growing trend to use real-world data, as exemplified by Zhao and Linetsky's 2021 study using LOB data from the Chicago Mercantile Exchange (CME) on S&P 500 and 10-year Treasury note [354].

Chan and Shelton's pioneering work [64] marked the earliest application of RL to market-making. They proposed the first electronic market maker to quote bid and ask prices, drawing inspiration from Glosten and Milgrom's seminal work [124]. Following this, the field has seen notable advancements with the demonstration by Spooner et al. [290] that an asymmetrically dampened reward function improved learning stability, and Ganesh et al. [118] constructed a competitive multi-agent based on PPO. In addition, Guéant and Manziuk [135] proposed the first market-making framework for the optimisation of multiple corporate bonds under a model-based Actor-Critic RL framework, marking a significant step in the field.

Avellaneda and Stoikov's classic market-making model [20] inspired Spooner and Savani [291] to introduce a game-theoretic approach to market-making using adversarial RL. Gašperov and Kostanjčar [122] proposed a derivative-free adversarial neuroevolution-based RL [298] market-making agent. Zhao and Linetsky [354] included the Book Exhaustion Rate (BER) as part of the features to protect market makers from informed traders. The most recent publication by Zhong et al. [357] used a Q-learning framework to create a market-making agent with the aim of maximising the expected net profit. Their approach outperformed benchmark strategies using historical LOB data, garnering the interest of a company intending to implement this framework. However, more real-world implementation of such strategies and thorough examination of their effectiveness are necessary for future research.

7.6 End-to-End Trading Systems

Within the surveyed literature, most publications address a single facet of quantitative trading, such as PM or execution. However, a handful of studies have sought to incorporate multiple elements of trading, creating intriguing end-to-end systems. These systems showcase the adaptability of the RL framework across various aspects of QF, representing an initial step towards an autonomous trading system that minimises human intervention.

Lee and Jagmin's work [191] is notable for being one of the first attempts at creating such a system. In their research, two agents generate trading signals while the remaining two manage order placement (for more details, see 6.6). In a subsequent study, Dempster and Leemans [83] developed an RL trader that integrates with a broader trading system that emphasises risk management. The proposed system, designed for FX markets, is based on three pillars: an ML algorithm [235], risk and performance management, and utility-driven dynamic optimisation. The authors introduced several adjustments to the primary concept, using "one-at-a-time random search optimisation" to navigate the system's inherent complexity.

In recent efforts, Patel et al. [256] proposed a collaborative multi-agent market-making system that uses DQN. Here, the "Macro agent" operates on minute tick data to decide whether to buy, sell or hold an asset. Subsequently, a "Micro agent" leverages the order book data to determine the order placement within the LOB. Similarly, Wang et al. [327] addressed the common assumption in the existing literature that each portfolio allocation can be executed instantaneously, thus overlooking price slippage. They introduced a Hierarchical Reinforced Framework for Portfolio Management (HRPM), incorporating high-level portfolio management and

low-level trading execution policies. The former is optimised through REINFORCE with an additional entropy term to promote diversification, while the latter uses the Branching Dueling Q-Network [308].

These approaches of Patel et al. [256] and Wang et al. [327], which combine crucial elements of QF into a single trading system, serve as the first significant advancements since the groundbreaking work of Mnih et al. [231, 232]. One possible enhancement to these approaches could be the integration of a risk management module [83].

7.7 Business Cycles and Augmenting Existing Trading Strategies

RL has emerged as a powerful tool in financial trading, with its ability to learn and optimise from iterative interactions in a dynamic environment. In this subsection, we examine the innovative use of RL in the tuning and optimisation of trading strategies.

Inspired by seminal work on business cycles [176], Tan et al. [307] developed a trading algorithm that capitalises on stock price cycles. The core idea of their approach is to initiate and terminate trades as close as possible to each stock's turning point. This strategy first involves identifying asset-specific cycles optimised via RL within a Value-based framework. The authors proceeded to design a bespoke reward function suited to their requirements. Turning points, which signal when to open or close a position, were identified through an Adaptive Network Fuzzy Inference System. The final component is the trading agent, whose goal is to execute trades in close proximity to these determined turning points, once again harnessing RL within the Value-based framework for identification. The unique feature of this application is the deployment of RL as a constituent part of a more comprehensive trading strategy, underscoring its potential for fine-tuning various system components. Eilers et al. [93] provide another example of RL's application in refining trading strategy parameters by leveraging recognised seasonal effects [13, 114, 141]. Their RL-based strategy evaluates whether to generate long/short signals, holding periods, and leverage levels before event days.

Kim and Kim [174] implemented a DQN for pair trading, rewarding the agent when the spread between two similar companies reaches a predefined threshold and reverts to the mean. The agent is penalised if stop-loss limits are hit or the spread fails to revert. The environment focuses solely on the spread but could be enhanced by incorporating factors like liquidity. Wang et al. [328] applied a similar strategy to five-year data from 75 NASDAQ companies, outperforming the baseline in terms of SR. Cartea et al. [59] are perhaps the first to apply DDQN and Reinforced Deep Markov Models (RDMMs) [103] to FX triplets. In an FX triplet, one of the three FX pairs is redundant due to the no-arbitrage rule in frictionless markets[24]. Therefore, establishing an opposing position is expected to generate profit when the triplet deviates from the no-arbitrage relationship. Through a simulation study, the authors favoured the use of RDMMs. Kim et al. [175] proposed a hybrid RL framework in which the first component, based on the TD3 algorithm (Actor-Critic) [115], is responsible for trading actions in pairs trading, while the second component, based on a DDQN agent, is responsible for stop-loss boundaries. In recognition of the risk of a structural break in the trading pair, Lu et al. [218] introduced a hybrid RL-based on DQN in which structural break detection mechanisms are used as input to a DQN agent.

In summary, these studies illustrate the versatility and robustness of RL in enhancing trading strategies, demonstrating its ability to adapt to complex market dynamics and optimise execution in response to changes. Future research might explore the applicability of these methodologies, especially in markets with different characteristics or diverse economic conditions.

7.8 Asset Allocation

The versatility of the RL framework has been evidenced through various applications, including parameter tuning, exemplified by Tan et al. [307], who deployed RL to optimise parameters to discern stock-specific cycles. However, the potential role of the RL in strategic allocation - referring to long-term asset allocation - and market timing, which involves switching capital between asset classes or financial markets based on predictive methods, is not yet thoroughly explored in existing research. A notable exception is the work of Hens and Wöhrmann [149], who apply the RRL framework [236] for the dynamic allocation between bonds and equity indices. The allocation between the stock and bond indices is expressed as the portfolio return $R_t^P \in \mathbb{R}$, as follows:

$$R_t^P = w_t^\theta R_t^B + \left(1 - w_t^\theta\right) R_t^E, \tag{10}$$

Here, R_t^B and R_t^E denote bond and equity returns at time t respectively, and $w_t \in [0, 1]$ represents weights under an exponential parametric function based on θ. Similarly, Du et al. [90] switch allocations between risk-free and risky assets and a Value-based approach [258] was used to allocate between the S&P500 ETF and the AGG Bond Index or the 10-year US Treasury note.

---

[24]The no-arbitrage rule asserts that in an efficient market, it is impossible to make a risk-free profit through simultaneous buying and selling of related assets, as price discrepancies are quickly corrected.

Although the described framework appears simplistic, it uncovers novel avenues for applying RL in finance. The Hens and Wöhrmann framework [149] is a valuable tool for fund managers who frequently need to alternate between asset classes. However, there are opportunities to extend their work by incorporating more than two asset classes and integrating exogenous macroeconomic variables into the environment. Given that fund managers often consider macroeconomic conditions when strategising capital deployment for their clients, such enhancements could increase the relevance and application of this approach in the real world.

7.9 Human in the Loop – Robo-Advising

Most applications reviewed here aim to minimise human intervention and implement autonomous trading systems. However, Robo-advising stands out as an area where human input remains essential. Characterised as a class of algorithms, Robo-advisors use client-specific risk appetites for financial planning and investment advice, offering an innovative alternative to traditional investment management. Human input is necessary for establishing risk preferences. Robo-advisors confer multiple advantages over traditional investment managers, including cost-effectiveness due to reduced overheads, such as office maintenance. In addition, they mitigate human biases associated with asset allocation, recommendations, and data collection [109]. Moreover, Robo-advising has found acceptance among systemically significant entities, indicating its growing industry significance. Historically, risk preferences were determined through questionnaires [65], a method potentially subject to bias [154]. In contrast, the studies in this section use IRL and Inverse Portfolio Optimisation (IPO) to estimate client risk tolerance.

Central to the reviewed literature is the concept that Robo-advisors learn or infer an investor's risk preferences through iterative interactions. For instance, in the work of Alsabah et al. [12], the Robo-advisor allocates the investor's capital across several pre-constructed portfolios, each reflecting distinct levels of risk preference. These preferences can then be updated when the Robo-advisor requests the investor to modify portfolio allocations. This methodology, while innovative, incurs opportunity costs, as investors must spend time understanding the financial environment in which their wealth is invested. Furthermore, this framework introduces an exploration/exploitation trade-off. Robo-advisors must choose between investing based on current risk preference estimates or seeking updated risk preferences, which may be costly. The variability of investor preferences over time adds another layer of complexity. To address these challenges, the authors propose a planning algorithm similar to the IRL framework and demonstrate that convergence to the optimal policy correlates with the consistency of investor choices. Further innovation comes from Wang and Yu [329], who proposed a two-agent framework: the first agent uses IPO to infer investor preferences and expected returns from past allocations, while the second agent, based on DDPG, performs a multi-period mean-variance optimisation. Yu and Dong [346] also use IPO, with the DDPG agent serving as a benchmark case, while Dixon and Halperin [88] apply G-Learning [111] in an IRL context, with applications in wealth management and Robo-advising.

Despite the intrigue surrounding Robo-advising, the field remains under-researched. As one of the few RL applications in finance involving human feedback, future research could explore multi-agent Robo-advisors that compete or collaborate. This could illuminate how risk preferences operate in this context and reveal aggregate risk preferences.

8 Discussion

In this survey of 167 research articles, we discovered numerous compelling concepts for practical trading applications. RL agents generally exceeded their respective benchmarks, with notable exceptions such as Sherstov and Stone [284]. The robustness of these frameworks is manifested across various global regions and asset classes. For example, Maringer and Ramtohul [223] examined the performance of equity indices in the UK, France, Germany, and Switzerland, while Bekiros [31] focused on stock indices in Japan and the United States. Bisht and Kumar [43] honed in on Indian stock indices, with others, including Sawhney et al. [276] and Fang et al. [101], extending their frameworks to the Chinese and Hong Kong financial markets. The application of the RL framework to diverse asset classes is presented in the surveyed literature. Carapuço et al. [56] and Sornmayura [294] explored different currencies, Deng et al. [85] investigated commodities, and Gueant and Manziuk [135] delved into fixed income. Cryptocurrencies such as Bitcoin and Ethereum were the focus of Sattarov et al. [275], while Ye et al. [344] studied a combination of cryptocurrencies and stocks. Further extensions to multiple assets using different RL algorithms were made by authors such as Zhang et al. [352]. Yuan et al. [347] and Katongo and Bhattacharyya [171] performed comparative analyses across three main Actor-Critic algorithms. Notable contributions were also made by Lavko et al. [186], who evaluated the trading performance of various model-free agents, comparing them with the classical mean-variance framework [226] and equally weighted portfolios. RL agents consistently surpassed the respective benchmarks in these comparisons.

Despite the impressive performance of RL agents across regions and asset classes, concerns arise about the evaluation practices in the literature. Benhamou [35] compares an RL PM framework with classic strategies [226]. However, this comparison may be unbalanced, as the RL framework uses a different information set than the benchmarks. Hence, a comparison with the basic framework [226] could be incomplete. A fairer comparison with all models using identical input data could be ideal, thereby offering a genuine reflection of each method's data-leveraging ability and providing a more authentic evaluation of their merits.

Numerous studies use the SR as a performance metric. However, excessively high SR values warrant careful scrutiny. Since it can be a product, for example, of overfitting [22] and data snooping bias [299]. Moreover, empirical evidence suggests that an SR above 3 is exceptional. Thus, findings such as Deng et al. [85] with an SR of 21.2, or Xu et al. [340] astonishing SR of 217.68, raise questions. A possible explanation could be the forward-looking bias [35, 36], as the authors typically observe prices at time t and actions are taken at the same time. Introducing a lag could make these models more realistic, although more challenging.

In subsection 6.7, we highlighted the rare instances of model interpretability in the current literature. Although Markowitz's methodology [226] is simple, its transparency aids in understanding the portfolio weight output, contrasting with the complex frameworks often found in recent RL literature. For example, Gold [125] discovered that lagged returns were more significant than the count of hidden neurons, although these results varied by currency pair. This finding revealed complexities, as the optimal parameters varied by currency and exhibited interdependence. From a practical viewpoint, it is crucial to have a robust rationale for parameter selection and a clear explanation for any observed variances. When introducing a new algorithm into a trading system, the three primary considerations are the complexity it imposes, the explicability of its performance, and the investment rationale. Failing to justify these aspects can erode transparency and possibly lead to misrepresentation of investment products. In the reviewed literature, attempts to justify complex methodologies are sparse. Exceptions include Ponomarev et al. [265] grid-search strategy to optimise NN architecture. Recently, Aboussalah and Lee [5] proposed an RRL-based agent capable of hyper-parameter tuning via Bayesian optimisation. This automated Gaussian process, with Expected Improvement as the acquisition function, enables the trading system to select architectures optimised for maximising the selected reward.

There is a compelling need to expand the existing literature to incorporate a detailed examination of trading strategies based on conventional factors, especially in the context of PM, as introduced in subsection 4.2.3. These stylised investment factors, backed by a robust body of literature, call for a comprehensive understanding of their influence on the performance of an RL trading system. A truly effective alpha strategy[25] based on RL should exhibit significance after accounting for those factors [98–100]. Only recently, Cong et al. [74] evaluated their RL agent using a framework similar to the one aforementioned. They convincingly demonstrated the robustness of their strategy even after integrating several control variables.

One critique, inextricably linked to the preceding discourse, relates to the trend of treating RL in QF predominantly as an "engineering " issue. This is evident from the tendency of authors to draw heavily on ML and DL literature, applying these concepts to QF after making minor modifications or exploring alternative NN architectures and feature selection mechanisms. This approach can be enriched by incorporating conceptual and innovative developments, as advocated in Section 6. Future research should look beyond engineering solutions, striving for innovation relevant to the QF domain, as seen in seminal works by Moody and Wu [235]. The overengineering of solutions brings forth another concern – the vast array of alternative solutions based on varied network architectures makes it challenging to identify superior approaches and pinpoint the essential components within a given solution.

Another issue to address is the evident survivorship bias[26] in some publications, for example, Kaur [172] and Wu et al. [338]. These works often apply their frameworks to successful single stock names like Apple Inc., neglecting to test their methodology on stocks that are currently non-existent or bankrupt. Furthermore, the focus on a limited set of tradable assets introduces potential survivorship bias, and the success of these assets does not necessarily imply effectiveness across other available assets. To lend greater authenticity, we should scrutinise the proposed trading system on benchmark constituents over time. Recent studies have shown progress in this regard. For example, Lee et al. [195] extended their RL framework to the Russell 3000, while Wang et al. [323] used Wharton Research Data Services (WRDS) to select a quarter of valid stocks annually.

9 Future Directions

This section outlines promising research opportunities emerging from our comprehensive analysis of RL in QF:

---

[25]An investment strategy that aims to generate returns that exceed the performance of a benchmark index, after adjusting for systematic (market) risk.
[26]Refers to the selection of stocks that have performed well and survived throughout the sample period.

Exploration of Alternative Features: In subsection 4.2, we underscore the potential for investigating features derived from alternative data sources, such as ESG and macroeconomic-based features. Accurate environment modelling is crucial for successful RL, especially in the chaotic world of financial markets. Consequently, finding meaningful features can determine the success or failure of a trading system. Although exploring alternative data could present challenges like data retrieval and time-consuming preprocessing, the potential benefits merit further investigation.

Knowledge transfer in trading systems: Subsection 6.3 highlights the potential for applying transfer learning in financial markets. This method could expedite the training process and complete missing historical data for stocks with a brief history. The limited available literature [332] points to the potential of combining model-based RL with transfer learning, as it could lead to enhanced performance and faster convergence by reducing data dependency.

Meta-learning [278, 312], as a way of generalising knowledge from multiple tasks, presents another exciting research opportunity. This method could allow the agent to learn new tasks faster and better, while providing a natural regularisation method to prevent overfitting and enhance generalisation performance. A promising application of meta-learning in financial trading systems could be its use when an agent needs to adapt to a new market or set of assets. Moreover, multi-task learning—a related concept in which a single network is simultaneously trained on multiple related tasks—presents a compelling approach to accelerate the training process and improve regularisation [60]. Its potential is especially evident in contexts such as training on stocks within the same sector or analysing LOB data of similar stocks.

Multi-agent solutions: Subsection 6.6 examined multi-agent applications and identified future research directions. Recognising that financial markets are essentially MAS, their exploration is crucial for revealing hidden market structures and enhancing our understanding of these intricate systems. A key challenge is scalability: the existing literature often restricts its focus to a few agents, pointing to the need for wider research. MAS also present an opportunity to assess the market impact of individual agents, challenging the assumption of an unaffected environment. Another underexplored aspect is the establishment of coordination and communication protocols among different agents in the QF context [107]. Effective protocols can reduce information asymmetry, increase market transparency, and improve market efficiency. For instance, within a single firm, if one trading agent uncovers a flaw in a common strategy, this discovery could be shared quickly, allowing for strategic adjustments, improved overall performance, and minimised capital losses. Hence, this aspect warrants further examination. Finally, we eagerly anticipate further research exploring integrating MAS with various ML concepts and financial applications, as underscored in subsection 7.9.

Multi-objective RL: In subsection 7.2, we detailed RL's relevance in PM. Future research could target a more natural constraint of the action space, bypassing heuristic methods. This includes constraints at various levels such as sectors, countries, and individual stocks. This prompts a demand for a multi-objective RL framework addressing conflicting objectives (e.g., [234]). We might draw from Markowitz's PM approach [226], which balances conflicting objectives through a convex optimisation problem, enabling a tradeoff between returns and variance. Therefore, any model aspiring to supersede Markowitz's must offer superior performance and comparable flexibility. This observation logically moves us towards the promising idea of multi-objective RL in finance.

Different applications of Robo-adviser: In subsection 7.9, we examined Robo-advisers in PM and suggested future research. Human-in-the-loop methodology exploration should expand beyond this. An intriguing trajectory, outlined by Alsabah et al. [12], is Robo-advisers for tax loss harvesting[27] [296], offering the potential to decrease temporal and monetary expenditure.

Hierarchical RL: While Hierarchical RL (HRL) is a concept of significant potential within RL literature [198, 203], it is noticeably underexplored in QF apart from a few exceptions [327]. HRL leverages hierarchical relationships among agents, breaking down complex problems into smaller tasks, thus enabling faster resolutions. In QF, this could streamline processes like asset selection and execution or further subdivide asset selection into a hierarchy of steps for more efficient problem-solving.

Single vs. many stock trading systems: Trading systems typically target single assets or multiple assets. Future research might construct portfolios from several single-asset trading systems. This would harness the idiosyncratic detail of single-asset systems and the holistic perspective of multi-asset systems, underpinning a comprehensive asset management strategy [6].

Human in the loop: In subsection 7.9, we discussed instances requiring human feedback as an integral part of the RL framework. Another potential application arises within the context of PD or IL. Notably, when the expert or teacher lacks access to an optimal

---

[27]Tax loss harvesting refers to offsetting capital gains tax liabilities on appreciated investments by selling those that have experienced a loss.

solution but has a preferred strategy, the process of distillation or IL could aim to replicate this. This approach could be especially valuable when the goal is to integrate human expertise or specific strategic intentions into the automated decision-making process.

10 Conclusion

We conducted a comprehensive survey of 167 publications exploring RL applications in QF. Our findings reveal a promising alternative to classic QF methodologies, particularly in areas such as PM and option hedging. We also examined the practical use of ML concepts, including transfer learning, imitation learning, and multi-agent RL, underscoring their potential impact on future research. Throughout our analysis, we emphasise the fundamental components of RL: environment, rewards, and action. We discussed key advancements in these areas from a QF viewpoint, scrutinising their contribution to RL development in QF.

Although acknowledging the progress made in recent publications, we also identified challenges and limitations in earlier stages of the literature, particularly with respect to results and methodologies. As QF is an inherently applied field, it is crucial that the proposed solutions align with real-world conditions and meet rigorous standards closely. Looking ahead, we put forward several ideas for future research in various sections of our survey. In conclusion, our survey provides valuable insights into the current landscape of RL in QF and presents a roadmap for future research in this rapidly evolving field. By addressing the identified challenges and pursuing these research directions, we can further enhance the effectiveness and applicability of RL methods in QF, ultimately advancing the understanding and practice of intelligent decision-making in financial markets.

Acknowledgments

References

[1] Abbeel, P. and Ng, A.Y., 2004, July. Apprenticeship learning via inverse reinforcement learning. In Proceedings of the twenty-first international conference on Machine learning (p. 1).

[2] AbdelKawy, R., Abdelmoez, W.M. and Shoukry, A., 2021. A synchronous deep reinforcement learning model for automated multi-stock trading. Progress in Artificial Intelligence, 10(1), pp.83-97.

[3] Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M. and Kim, B., 2018. Sanity checks for saliency maps. Advances in neural information processing systems, 31.

[4] Abecasis, S.M., Lapenta, E.S. and Pedreira, C.E., 1999. Performance metrics for financial time series forecasting. J. Comput. Intell. in Finance, 7(4), pp.5-22.

[5] Aboussalah, A.M. and Lee, C.G., 2020. Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization. Expert Systems with Applications, 140, p.112891.

[6] Aboussalah, A.M., Xu, Z. and Lee, C.G., 2022. What is the value of the cross-sectional approach to deep reinforcement learning?. Quantitative Finance, 22(6), pp.1091-1111.

[7] Alameer, A. and Al Shehri, K., 2022, August. Conditional value-at-risk for quantitative trading: A direct reinforcement learning approach. In 2022 IEEE Conference on Control Technology and Applications (CCTA) (pp. 1208-1213). IEEE.

[8] Almahdi, S. and 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, pp.267-279.

[9] Almahdi, S. and Yang, S.Y., 2019. A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning. Expert Systems with Applications, 130, pp.145-156.

[10] Almgren, R. and Chriss, N., 2001. Optimal execution of portfolio transactions. Journal of Risk, 3, pp.5-40.

[11] Antweiler, W. and Frank, M.Z., 2004. Is all that talk just noise? The information content of internet stock message boards. The Journal of finance, 59(3), pp.1259-1294.

[12] Alsabah, H., Capponi, A., Ruiz Lacedelli, O. and Stern, M., 2021. Robo-advising: Learning investors' risk preferences via portfolio choices. Journal of Financial Econometrics, 19(2), pp.369-392.

[13] Ariel, R.A., 1987. A monthly effect in stock returns. Journal of financial economics, 18(1), pp.161-174.

[14] Arrow, K.J. and Debreu, G., 1954. Existence of an equilibrium for a competitive economy. Econometrica: Journal of the Econometric Society, pp.265-290.

[15] Asiain, E., Clempner, J.B. and Poznyak, A.S., 2019. Controller exploitation-exploration reinforcement learning architecture for computing near-optimal policies. Soft Computing, 23(11), pp.3591-3604.

[16] Åström, K.J., 1965. Optimal control of Markov processes with incomplete state information. Journal of mathematical analysis and applications, 10(1), pp.174-205.

[17] Atsalakis, G.S. and Valavanis, K.P., 2009. Surveying stock market forecasting techniques–Part II: Soft computing methods. Expert Systems with applications, 36(3), pp.5932-5941.

[18] Audibert, J.Y., Munos, R. and Szepesvári, C., 2009. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. Theoretical Computer Science, 410(19), pp.1876-1902.

[19] Auer, P., Cesa-Bianchi, N. and Fischer, P., 2002. Finite-time analysis of the multiarmed bandit problem. Machine learning, 47, pp.235-256.

[20] Avellaneda, M. and Stoikov, S., 2008. High-frequency trading in a limit order book. Quantitative Finance, 8(3), pp.217-224.

[21] Babcock, B.A., Choi, E.K. and Feinerman, E., 1993. Risk and probability premiums for CARA utility functions. Journal of Agricultural and Resource Economics, pp.17-24.

[22] Bailey, D.H., Borwein, J.M., de Prado, M.L. and Zhu, Q.J., 2014. Pseudomathematics and financial charlatanism: The effects of backtest over fitting on out-of-sample performance. Notices of the AMS, 61(5), pp.458-471.

[23] Baird, L. and Moore, A., 1998. Gradient descent for general reinforcement learning. Advances in neural information processing systems, 11.

[24] Banz, R.W., 1981. The relationship between return and market value of common stocks. Journal of financial economics, 9(1), pp.3-18.

[25] Bao, W. and Liu, X.Y., 2019. Multi-agent deep reinforcement learning for liquidation strategy analysis. arXiv preprint arXiv:1906.11046.

[26] Bao, W., 2019. Fairness in multi-agent reinforcement learning for stock trading. arXiv preprint arXiv:2001.00918.

[27] Barberis, N. and Thaler, R., 2003. A survey of behavioral finance. Handbook of the Economics of Finance, 1, pp.1053-1128.

[28] Barto, A.G. and Anandan, P., 1985. Pattern-recognizing stochastic learning automata. IEEE Transactions on Systems, Man, and Cybernetics, (3), pp.360-375.

[29] Bates, R.G., Dempster, M.A. and Romahi, Y.S., 2003, March. Evolutionary reinforcement learning in FX order book and order flow analysis. In 2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings. (pp. 355-362). IEEE.

[30] Baytas, I.M., Xiao, C., Zhang, X., Wang, F., Jain, A.K. and Zhou, J., 2017, August. Patient subtyping via time-aware LSTM networks. In Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 65-74).

[31] Bekiros, S.D., 2010. Heterogeneous trading strategies with adaptive fuzzy actor–critic reinforcement learning: A behavioral approach. Journal of Economic Dynamics and Control, 34(6), pp.1153-1170.

[32] Bellemare, M.G., Dabney, W. and Munos, R., 2017, July. A distributional perspective on reinforcement learning. In International Conference on Machine Learning (pp. 449-458). PMLR.

[33] Bellman R., Dynamic Programming. Courier Corporation, 1957, 2013. 356

[34] Bengio, Y., Simard, P. and Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), pp.157-166.

[35] Benhamou, E., Saltiel, D., Ungari, S. and Mukhopadhyay, A., 2020. Bridging the gap between Markowitz planning and deep reinforcement learning. arXiv preprint arXiv:2010.09108.

[36] Benhamou, E., Saltiel, D., Ungari, S., Mukhopadhyay, A. and Atif, J., 2020. Aamdrl: Augmented asset management with deep reinforcement learning. arXiv preprint arXiv:2010.08497.

[37] Benhamou, E., Saltiel, D., Ohana, J.J. and Atif, J., 2021, January. Detecting and adapting to crisis pattern with context based Deep Reinforcement Learning. In 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 10050-10057). IEEE.

[38] Benhamou, E., Saltiel, D., Tabachnik, S., Wong, S.K. and Chareyron, F., 2021. Adaptive learning for financial markets mixing model-based and model-free RL for volatility targeting. arXiv preprint arXiv:2104.10483.

[39] Bertoluzzo, F. and Corazza, M., 2007, September. Making financial trading by recurrent reinforcement learning. In International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (pp. 619-626). Springer, Berlin, Heidelberg.

[40] Bertoluzzo, F. and Corazza, M., 2012. Testing different reinforcement learning configurations for financial trading: Introduction and applications. Procedia Economics and Finance, 3, pp.68-77.

[41] Bertsimas, D. and Lo, A.W., 1998. Optimal control of execution costs. Journal of financial markets, 1(1), pp.1-50.

[42] Bishop, C.M., 1994. Mixture density networks.

[43] Bisht, K. and Kumar, A., 2020, December. Bisht, K. and Kumar, A., 2020, December. Deep Reinforcement Learning based Multi-Objective Systems for Financial Trading. In 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE) (pp. 1-6). IEEE.

[44] Bisi, L., Sabbioni, L., Vittori, E., Papini, M. and Restelli, M., 2019. Risk-averse trust region optimization for reward-volatility reduction. arXiv preprint arXiv:1912.03193.

[45] Black, F. and Scholes, M., 1973. The pricing of options and corporate liabilities. Journal of political economy, 81(3), pp.637-654.

[46] Bollerslev, T.,1986. Generalized Autoregressive Conditional Heteroskedasticity. Journal of Econometrics. 31 (3), 307−327. Doi: 10.1016/0304-4076(86)90063-1.

[47] Borrageiro, G., Firoozye, N. and Barucca, P., 2021. Reinforcement learning for systematic FX trading. IEEE Access, 10, pp.5024-5036.

[48] Borrageiro, G., Firoozye, N. and Barucca, P., 2022. The Recurrent Reinforcement Learning Crypto Agent. IEEE Access, 10, pp.38590-38599.

[49] Briola, A., Turiel, J., Marcaccioli, R. and Aste, T., 2021. Deep reinforcement learning for active high frequency trading. arXiv preprint arXiv:2101.07107.

[50] Broomhead, D.S. and Lowe, D., 1988. Radial basis functions, multi-variable functional interpolation and adaptive networks. Royal Signals and Radar Establishment Malvern (United Kingdom).

[51] Buehler, H., Gonon, L., Teichmann, J., Wood, B., Mohan, B. and Kochems, J., 2019. Deep hedging: hedging derivatives under generic market frictions using reinforcement learning. Swiss Finance Institute Research Paper, (19-80).

[52] Busoniu, L., Babuška, R. and 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), pp.156-172.

[53] Buşoniu, L., Babuška, R. and De Schutter, B., 2010. Multi-agent reinforcement learning: An overview. Innovations in multi-agent systems and applications-1, pp.183-221.

[54] Cao, J., Chen, J., Hull, J. and Poulos, Z., 2021. Deep hedging of derivatives using reinforcement learning. The Journal of Financial Data Science, 3(1), pp.10-27.

[55] Cannelli, L., Nuti, G., Sala, M. and Szehr, O., 2020. Hedging using reinforcement learning: Contextual $ k $-Armed Bandit versus $ Q $-learning. arXiv preprint arXiv:2007.01623.

[56] Carapuço, J., Neves, R. and Horta, N., 2018. Reinforcement learning applied to Forex trading. Applied Soft Computing, 73, pp.783-794.

[57] Carta, S., Corriga, A., Ferreira, A., Podda, A.S. and Recupero, D.R., 2021. A multi-layer and multi-ensemble stock trader using deep learning and deep reinforcement learning. Applied Intelligence, 51(2), pp.889-905.

[58] Cartea, Á., Jaimungal, S. and Penalva, J., 2015. Algorithmic and high-frequency trading. Cambridge University Press.

[59] Cartea, Á., Jaimungal, S. and Sánchez-Betancourt, L., 2021. Deep reinforcement learning for algorithmic trading. Available at SSRN 3812473.

[60] Caruana, R., 1998. Multitask learning (pp. 95-133). Springer US.

[61] Casgrain, P., Ning, B. and Jaimungal, S., 2022. Deep Q-learning for Nash equilibria: Nash-DQN. Applied Mathematical Finance, 29(1), pp.62-78.

[62] Chen, T. and Guestrin, C., 2016, August. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).

[63] Chakole, J.B., Kolhe, M.S., Mahapurush, G.D., Yadav, A. and Kurhekar, M.P., 2021. A Q-learning agent for automated trading in equity stock markets. Expert Systems with Applications, 163, p.113761.

[64] Chan, N.T. and Shelton, C., 2001. An electronic market-maker.

[65] Charness, G., Gneezy, U. and Imas, A., 2013. Experimental methods: Eliciting risk preferences. Journal of economic behavior and organization, 87, pp.43-51.

[66] Charpentier, A., Elie, R. and Remlinger, C., 2021. Reinforcement learning in economics and finance. Computational Economics, pp.1-38.

[67] Chen, Y., Mabu, S., Hirasawa, K. and Hu, J., 2007, July. Trading rules on stock markets using genetic network programming with sarsa learning. In Proceedings of the 9th annual conference on Genetic and evolutionary computation (pp. 1503-1503).

[68] Chen, L. and Gao, Q., 2019, October. Application of deep reinforcement learning on automated stock trading. In 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS) (pp. 29-33). IEEE.

[69] Cho, K., Van Merriënboer, B., Bahdanau, D. and Bengio, Y., 2014. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

[70] Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).

[71] Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[72] Cochrane, J.H., 2005. Asset pricing: Revised edition. Princeton university press.

[73] Cohen, L. and Frazzini, A., 2008. Economic links and predictable returns. The Journal of Finance, 63(4), pp.1977-2011.

[74] Cong, L.W., Tang, K., Wang, J. and Zhang, Y., 2020. AlphaPortfolio: Direct construction through reinforcement learning and interpretable AI. Capital Markets: Asset Pricing and Valuation eJournal.

[75] Cont, R., 2001. Empirical properties of asset returns: stylized facts and statistical issues. Quantitative finance, 1(2), p.223.

[76] Corazza, M. and Bertoluzzo, F., 2014. Q-learning-based financial trading systems with applications. University Ca'Foscari of Venice, Dept. of Economics Working Paper Series No, 15.

[77] Corazza, M. and Sangalli, A., 2015. Q-Learning and SARSA: a comparison between two intelligent stochastic control approaches for financial trading. University Ca'Foscari of Venice, Dept. of Economics Research Paper Series No, 15.

[78] Coqueret, G. and André, E., 2022. Factor investing with reinforcement learning. Available at SSRN 4103045.

[79] Dabérius, K., Granat, E. and Karlsson, P., 2019. Deep execution-value and policy based reinforcement learning for trading and beating market benchmarks. Available at SSRN 3374766.

[80] Dechter, R., 1986. Learning while searching in constraint-satisfaction problems.

[81] Dempster, M.A., Payne, T.W., Romahi, Y. and Thompson, G.W., 2001. Computational learning techniques for intraday FX trading using popular technical indicators. IEEE Transactions on neural networks, 12(4), pp.744-754.

[82] Dempster, M.A.H. and Romahi, Y.S., 2002, August. Intraday FX trading: An evolutionary reinforcement learning approach. In International Conference on Intelligent Data Engineering and Automated Learning (pp. 347-358). Springer, Berlin, Heidelberg.

[83] Dempster, M.A. and Leemans, V., 2006. An automated FX trading system using adaptive reinforcement learning. Expert Systems with Applications, 30(3), pp.543-552.

[84] Deng, Y., Kong, Y., Bao, F. and Dai, Q., 2015. Sparse coding-inspired optimal trading system for HFT industry. IEEE Transactions on Industrial Informatics, 11(2), pp.467-475.

[85] Deng, Y., Bao, F., Kong, Y., Ren, Z. and Dai, Q., 2016. Deep direct reinforcement learning for financial signal representation and trading. IEEE transactions on neural networks and learning systems, 28(3), pp.653-664.

[86] de Oliveira, R.A., Ramos, H.S., Dalip, D.H. and Pereira, A.C.M., 2020, October. A tabular sarsa-based stock market agent. In Proceedings of the First ACM International Conference on AI in Finance (pp. 1-8).

[87] Ding, Y., Liu, W., Bian, J., Zhang, D. and Liu, T.Y., 2018, July. Investor-imitator: A framework for trading knowledge extraction. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 1310-1319).

[88] Dixon, M. and Halperin, I., 2020. G-learner and girl: Goal based wealth management with reinforcement learning. arXiv preprint arXiv:2002.10990.

[89] Doshi-Velez, F. and Kim, B., 2017. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.

[90] Du, X., Zhai, J. and Lv, K., 2016. Algorithm trading using q-learning and recurrent reinforcement learning. positions, 1(1).

[91] Du, J., Jin, M., Kolm, P.N., Ritter, G., Wang, Y. and Zhang, B., 2020. Deep reinforcement learning for option replication and hedging. The Journal of Financial Data Science, 2(4), pp.44-57.

[92] Dubrov, B., 2015. Monte Carlo simulation with machine learning for pricing American options and convertible bonds. Available at SSRN 2684523.

[93] Eilers, D., Dunis, C.L., von Mettenheim, H.J. and Breitner, M.H., 2014. Intelligent trading of seasonal effects: A decision support algorithm based on reinforcement learning. Decision support systems, 64, pp.100-108.

[94] Engle, R.F. and Granger, C.W., 1987. Co-integration and error correction: representation, estimation, and testing. Econometrica: journal of the Econometric Society, pp.251-276.

[95] Esteban, C., Hyland, S.L. and Rätsch, G., 2017. Real-valued (medical) time series generation with recurrent conditional gans. arXiv preprint arXiv:1706.02633.

[96] Fabozzi, F.J. and Mann, S.V., 2012. The handbook of fixed income securities. McGraw-Hill Education.

[97] Fama, E.F., 1970. Efficient capital markets: A review of theory and empirical work. Journal of Finance, 25(2), pp.383-417.

[98] Fama, E.F. and French, K.R., 1992. The cross-section of expected stock returns. Journal of Finance, 47(2), pp.427-465.

[99] Fama, E.F. and French, K.R., 1993. Common risk factors in the returns on stocks and bonds. Journal of Financial Economics, 33, pp.3-56.

[100] Fama, E.F. and French, K.R., 1996. Multifactor explanations of asset pricing anomalies. Journal of Finance, 5, pp. 55-84

[101] Fang, Y., Ren, K., Liu, W., Zhou, D., Zhang, W., Bian, J., Yu, Y. and Liu, T.Y., 2021, May. Universal trading for order execution with oracle policy distillation. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 1, pp. 107-115).

[102] Fengqian, D. and Chao, L., 2020. An adaptive financial trading system using deep reinforcement learning with candlestick decomposing features. IEEE Access, 8, pp.63666-63678.

[103] Ferreira, T.A., 2020. Reinforced deep Markov models with applications in automatic trading. arXiv preprint arXiv:2011.04391.

[104] Feuerriegel, S. and Prendinger, H., 2016. News-based trading strategies. Decision Support Systems, 90, pp.65-74.

[105] Fischer, T.G., 2018. Reinforcement learning in the financial markets-a survey (No. 12/2018). FAU Discussion Papers in Economics.

[106] Fischer, T. and Krauss, C., 2018. Deep learning with long short-term memory networks for financial market predictions. European journal of operational research, 270(2), pp.654-669.

[107] Foerster, J., Assael, I.A., De Freitas, N. and Whiteson, S., 2016. Learning to communicate with deep multi-agent reinforcement learning. Advances in neural information processing systems, 29.

[108] Foerster, J.N., Chen, R.Y., Al-Shedivat, M., Whiteson, S., Abbeel, P. and Mordatch, I., 2017. Learning with opponent-learning awareness. arXiv preprint arXiv:1709.04326.

[109] Foerster, S., Linnainmaa, J.T., Melzer, B.T. and Previtero, A., 2017. Retail financial advice: does one size fit all?. The Journal of Finance, 72(4), pp.1441-1482.

[110] Fortunato, M., Azar, M.G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O. and Blundell, C., 2017. Noisy networks for exploration. arXiv preprint arXiv:1706.10295.

[111] Fox, R., Pakman, A. and Tishby, N., 2015. Taming the noise in reinforcement learning via soft updates. arXiv preprint arXiv:1512.08562.

[112] Fox, R. and Ludvig, E.A., 2024. Assimilating human feedback from autonomous vehicle interaction in reinforcement learning models. Autonomous Agents and Multi-Agent Systems, 38(2), p.26.

[113] Frazzini, A. and Pedersen, L.H., 2014. Betting against beta. Journal of Financial Economics, 111(1), pp.1-25.

[114] French, K.R., 1980. Stock returns and the weekend effect. Journal of financial economics, 8(1), pp.55-69.

[115] Fujimoto, S., Hoof, H. and Meger, D., 2018, July. Addressing function approximation error in actor-critic methods. In International conference on machine learning (pp. 1587-1596). PMLR.

[116] Gabrielsson, P. and Johansson, U., 2015, December. High-frequency equity index futures trading using recurrent reinforcement learning with candlesticks. In 2015 IEEE Symposium Series on Computational Intelligence (pp. 734-741). IEEE.

[117] Ganesh, P. and Rakheja, P., 2018. Deep reinforcement learning in high frequency trading. arXiv preprint arXiv:1809.01506.

[118] Ganesh, S., Vadori, N., Xu, M., Zheng, H., Reddy, P. and Veloso, M., 2019. Reinforcement learning for market making in a multi-agent dealer market. arXiv preprint arXiv:1911.05892.

[119] Gao, X. and Chan, L., 2000. An algorithm for trading and portfolio management using q-learning and sharpe ratio maximization. In Proceedings of the international conference on neural information processing (pp. 832-837).

[120] García-Galicia, M., Carsteanu, A.A. and Clempner, J.B., 2019. Continuous-time reinforcement learning approach for portfolio management with time penalization. Expert Systems with Applications, 129, pp.27-36.

[121] Gârleanu, N. and Pedersen, L.H., 2013. Dynamic trading with predictable returns and transaction costs. The Journal of Finance, 68(6), pp.2309-2340.

[122] Gašperov, B. and Kostanjčar, Z., 2021. Market making with signals through deep reinforcement learning. IEEE Access, 9, pp.61611-61622.

[123] Gers, F.A., Schmidhuber, J. and Cummins, F., 2000. Learning to forget: Continual prediction with LSTM. Neural computation, 12(10), pp.2451-2471.

[124] Glosten, L.R. and Milgrom, P.R., 1985. Bid, ask and transaction prices in a specialist market with heterogeneously informed traders. Journal of financial economics, 14(1), pp.71-100.

[125] Gold, C., 2003, March. FX trading via recurrent reinforcement learning. In 2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings. (pp. 363-370). IEEE.

[126] Gompers, P., Ishii, J. and Metrick, A., 2003. Corporate governance and equity prices. The quarterly journal of economics, 118(1), pp.107-156.

[127] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial networks. Advances in Neural Information Processing Systems, 27, 2672-2680

[128] Goodfellow, I., Bengio, Y. and Courville, A., 2016. Deep learning. MIT press.

[129] Gordon, M.J., 1959. Dividends, earnings, and stock prices. The review of economics and statistics, pp.99-105.

[130] Gordon, G.J., 1995. Stable fitted reinforcement learning. Advances in neural information processing systems, 8.

[131] Gorse, D., 2011, April. Application of stochastic recurrent reinforcement learning to index trading. ESANN.

[132] Graham, B. (1973). The Intelligent Investor (4th rev. ed.). Harpers & Row, New York.

[133] Grondman, I., Busoniu, L., Lopes, G.A. and Babuska, R., 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(6), pp.1291-1307.

[134] Gu, Y., Mabu, S., Yang, Y., Li, J. and Hirasawa, K., 2011, September. Trading rules on stock markets using Genetic Network Programming-Sarsa learning with plural subroutines. In SICE Annual Conference 2011 (pp. 143-148). IEEE.

[135] Guéant, O. and Manziuk, I., 2019. Deep reinforcement learning for market making in corporate bonds: beating the curse of dimensionality. Applied Mathematical Finance, 26(5), pp.387-452.

[136] Guestrin, C., Koller, D. and Parr, R., 2001. Multiagent planning with factored MDPs. Advances in neural information processing systems, 14.

[137] Haarnoja, T., Zhou, A., Abbeel, P. and Levine, S., 2018, July. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In International conference on machine learning (pp. 1861-1870). PMLR.

[138] Halperin, I., 2020. Qlbs: Q-learner in the black-scholes (-merton) worlds. The Journal of Derivatives, 28(1), pp.99-122.

[139] Hambly, B., Xu, R. and Yang, H., 2021. Recent advances in reinforcement learning in finance. arXiv preprint arXiv:2112.04553.

[140] Hansen, L.K. and Salamon, P., 1990. Neural network ensembles. IEEE transactions on pattern analysis and machine intelligence, 12(10), pp.993-1001.

[141] Hansen, P.R., Lunde, A. and Nason, J.M., 2005. Testing the significance of calendar effects. Federal Reserve Bank of Atlanta Working Paper, (2005-02).

[142] Harvey, C.R. and Liu, Y., 2019. A census of the factor zoo. Available at SSRN 3341728.

[143] Hasselt, H., Guez, A. and Silver, D., 2016, March. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI conference on artificial intelligence (Vol. 30, No. 1).

[144] Hasselt, H.P., Guez, A., Hessel, M., Mnih, V. and Silver, D., 2016. Learning values across many orders of magnitude. Advances in neural information processing systems, 29.

[145] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[146] Heess, N., Hunt, J.J., Lillicrap, T.P. and Silver, D., 2015. Memory-based control with recurrent neural networks. arXiv preprint arXiv:1512.04455.

[147] Henaff, M., Bruna, J. and LeCun, Y., 2015. Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163.

[148] Hendricks, D. and Wilcox, D., 2014, March. A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution. In 2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr) (pp. 457-464). IEEE.

[149] Hens, T. and Wöhrmann, P., 2007. Strategic asset allocation and market timing: a reinforcement learning approach. Computational Economics, 29(3), pp.369-381.

[150] Hinton, G.E., Osindero, S. and Teh, Y.W., 2006. A fast learning algorithm for deep belief nets. Neural computation, 18(7), pp.1527-1554.

[151] Hinton, G., Vinyals, O. and Dean, J., 2015. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2(7).

[152] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.

[153] Holland, J.H., 1992. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press.

[154] Holt, C.A. and Laury, S.K., 2002. Risk aversion and incentive effects. American economic review, 92(5), pp.1644-1655.

[155] Hryshko, A. and Downs, T., 2004. System for foreign exchange trading using genetic algorithms and reinforcement learning. International journal of systems science, 35(13-14), pp.763-774.

[156] Huang, C.Y., 2018. Financial trading as a game: A deep reinforcement learning approach. arXiv preprint arXiv:1807.02787.

[157] Huang, Z. and Tanaka, F., 2022. MSPM: A modularized and scalable multi-agent reinforcement learning-based system for financial portfolio management. Plos one, 17(2), p.e0263689.

[158] Huo, X. and Fu, F., 2017. Risk-aware multi-armed bandit problem with application to portfolio selection. Royal Society open science, 4(11), p.171377.

[159] Hutter, M., 2005. Universal artificial intelligence: Sequential decisions based on algorithmic probability. Springer Science and Business Media.

[160] Jaeger, H., 2002. Adaptive nonlinear system identification with echo state networks. Advances in neural information processing systems, 15.

[161] Jangmin, O., Lee, J., Lee, J.W. and Zhang, B.T., 2006. Adaptive stock trading with dynamic asset allocation using reinforcement learning. Information Sciences, 176(15), pp.2121-2147.

[162] Jegadeesh, N. and Titman, S., 1993. Returns to buying winners and selling losers: Implications for stock market efficiency. The Journal of finance, 48(1), pp.65-91.

[163] Jeong, G. and Kim, H.Y., 2019. Improving financial trading decisions using deep Q-learning: Predicting the number of shares, action strategies, and transfer learning. Expert Systems with Applications, 117, pp.125-138.

[164] Jia, W.U., Chen, W.A.N.G., Xiong, L. and Hongyong, S.U.N., 2019, July. Quantitative trading on stock market based on deep reinforcement learning. In 2019 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[165] Jiang, Z., Xu, D. and Liang, J., 2017. A deep reinforcement learning framework for the financial portfolio management problem. arXiv preprint arXiv:1706.10059.

[166] Jiang, Z. and Liang, J., 2017, September. Cryptocurrency portfolio management with deep reinforcement learning. In 2017 Intelligent Systems Conference (IntelliSys) (pp. 905-913). IEEE.

[167] Jin, O. and El-Saawy, H., 2016. Portfolio management using reinforcement learning. Stanford University.

[168] Johannes, M., Korteweg, A. and Polson, N., 2014. Sequential learning, predictability, and optimal portfolio returns. The Journal of Finance, 69(2), pp.611-644.

[169] Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V. and Levine, S., 2018. QF-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. arXiv preprint arXiv:1806.10293.

[170] Karpe, M., Fang, J., Ma, Z. and Wang, C., 2020, October. Multi-agent reinforcement learning in a realistic limit order book market simulation. In Proceedings of the First ACM International Conference on AI in Finance (pp. 1-7).

[171] Katongo, M. and Bhattacharyya, R., 2021. The use of deep reinforcement learning in tactical asset allocation. Available at SSRN 3812609.

[172] Kaur, S., 2017. Algorithmic trading using sentiment analysis and reinforcement learning. positions.

[173] Kennedy, J. and Eberhart, R., 1995, November. Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). IEEE.

[174] Kim, T. and Kim, H.Y., 2019. Optimizing the pairs-trading strategy using deep reinforcement learning with trading and stop-loss boundaries. Complexity, 2019.

[175] Kim, S.H., Park, D.Y. and Lee, K.H., 2022. Hybrid Deep Reinforcement Learning for Pairs Trading. Applied Sciences, 12(3), p.944.

[176] Kitchin, J., 1923. Cycles and trends in economic factors. The Review of economic statistics, pp.10-16.

[177] Kolm, P.N. and Ritter, G., 2019. Dynamic replication and hedging: A reinforcement learning approach. The Journal of Financial Data Science, 1(1), pp.159-171.

[178] Konda, V. and Tsitsiklis, J., 1999. Actor-critic algorithms. Advances in neural information processing systems, 12.

[179] Konda, V.R. and Tsitsiklis, J.N., 2004. Convergence rate of linear two-time-scale stochastic approximation. The Annals of Applied Probability, 14(2), pp.796-819.

[180] Koratamaddi, P., Wadhwani, K., Gupta, M. and Sanjeevi, S.G., 2021. Market sentiment-aware deep reinforcement learning approach for stock portfolio allocation. Engineering Science and Technology, an International Journal, 24(4), pp.848-859.

[181] Kritzman, M. and Li, Y., 2010. Skulls, financial turbulence, and risk management. Financial Analysts Journal, 66(5), pp.30-41.

[182] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2017. Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), pp.84-90.

[183] Kumar, B., Roshan, A., Baranwal, A., Rajendran, S., Sharma, S., Mishra, A. and Vyas, O.P., 2022, August. Optimised Forex Trading using Ensemble of Deep Q-Learning Agents. In Proceedings of the 2022 Fourteenth International Conference on Contemporary Computing (pp. 417-428).

[184] Lagoudakis, M.G. and Parr, R., 2003. Least-squares policy iteration. The Journal of Machine Learning Research, 4, pp.1107-1149.

[185] Lample, G. and Chaplot, D.S., 2017, February. Playing FPS games with deep reinforcement learning. In Thirty-First AAAI Conference on Artificial Intelligence.

[186] Lavko, M., Klein, T. and Walther, T., 2023. Reinforcement Learning and Portfolio Allocation: Challenging Traditional Allocation Methods. Queen's Management School Working Paper, 1.

[187] Lazaric, A., 2012. Transfer in reinforcement learning: a framework and a survey. In Reinforcement Learning: State-of-the-Art (pp. 143-173). Berlin, Heidelberg: Springer Berlin Heidelberg.

[188] LeBaron, B., 1992. Some relations between volatility and serial correlations in stock market returns. Journal of Business, pp.199-219.

[189] LeCun, Y. and Bengio, Y., 1995. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361(10), p.1995.

[190] LeCun, Y., Bottou, L., Orr, G.B. and Müller, K.R., 2002. Efficient backprop. In Neural networks: Tricks of the trade (pp. 9-50). Berlin, Heidelberg: Springer Berlin Heidelberg

[191] Lee, J.W. and Jangmin, O., 2002, September. A multi-agent Q-learning framework for optimizing stock trading systems. In International Conference on Database and Expert Systems Applications (pp. 153-162). Springer, Berlin, Heidelberg.

[192] Lee, J.W. and Zhang, B.T., 2002. Stock trading system using reinforcement learning with cooperative agents. In Proceedings of the Nineteenth International Conference on Machine Learning (pp. 451-458).

[193] Lee, J.W., Kim, S.D., Lee, J. and Chae, J., 2003. An intelligent stock trading system based on reinforcement learning. IEICE TRANSACTIONS on Information and Systems, 86(2), pp.296-305.

[194] Lee, J.W., Park, J., Jangmin, O., Lee, J. and Hong, E., 2007. A multiagent approach to Q-learning for daily stock trading. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 37(6), pp.864-877.

[195] Lee, J., Kim, R., Yi, S.W. and Kang, J., 2020. MAPS: Multi-Agent reinforcement learning-based Portfolio management System. arXiv preprint arXiv:2007.05402.

[196] Leem, J. and Kim, H.Y., 2020. Action-specialized expert ensemble trading system with extended discrete action space using deep reinforcement learning. Plos one, 15(7), p.e0236178.

[197] Lei, K., Zhang, B., Li, Y., Yang, M. and Shen, Y., 2020. Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. Expert Systems with Applications, 140, p.112872.

[198] Levy, A., Konidaris, G., Platt, R. and Saenko, K., 2017. Learning multi-level hierarchies with hindsight. arXiv preprint arXiv:1712.00948.

[199] Li, J. and Chan, L., 2006, July. Reward adjustment reinforcement learning for risk-averse asset allocation. In The 2006 IEEE International Joint Conference on Neural Network Proceedings (pp. 534-541). IEEE.

[200] Li, H., Dagli, C.H. and Enke, D., 2007, April. Short-term stock market timing prediction under reinforcement learning schemes. In 2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (pp. 233-240). IEEE.

[201] Li, Y., Szepesvari, C. and Schuurmans, D., 2009, April. Learning exercise policies for American options. In Artificial Intelligence and Statistics (pp. 352-359). PMLR.

[202] Li, X., Li, Y., Zhan, Y. and Liu, X.Y., 2019. Optimistic bull or pessimistic bear: Adaptive deep reinforcement learning for stock portfolio allocation. arXiv preprint arXiv:1907.01503.

[203] Li, S., Wang, R., Tang, M. and Zhang, C., 2019. Hierarchical reinforcement learning with advantage-based auxiliary rewards. Advances in Neural Information Processing Systems, 32.

[204] Li, Y., Ni, P. and Chang, V., 2020. Application of deep reinforcement learning in stock trading strategies and stock forecasting. Computing, 102(6), pp.1305-1322.

[205] Li, L., 2021, November. An automated portfolio trading system with feature preprocessing and recurrent reinforcement learning. In Proceedings of the Second ACM International Conference on AI in Finance (pp. 1-8).

[206] Liagkouras, K. and Metaxiotis, K., 2018. A new efficiently encoded multiobjective algorithm for the solution of the cardinality constrained portfolio optimization problem. Annals of Operations Research, 267(1), pp.281-319.

[207] Liang, Z., Chen, H., Zhu, J., Jiang, K. and Li, Y., 2018. Adversarial deep reinforcement learning in portfolio management. arXiv preprint arXiv:1808.09940.

[208] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. and Wierstra, D., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

[209] Lim, Y.S. and Gorse, D., 2018, April. Reinforcement learning for high-frequency market making. In ESANN 2018-Proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (pp. 521-526). ESANN.

[210] Lin, C.T. and Lee, C.S.G., 1991. Neural-network-based fuzzy logic control and decision system. IEEE Transactions on computers, 40(12), pp.1320-1336.

[211] Lin, L.J., 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning, 8(3-4), pp.293-321.

[212] Lin, S. and Beling, P.A., 2020. An End-to-End Optimal Trade Execution Framework based on Proximal Policy Optimization. In IJCAI (pp. 4548-4554).

[213] Liu, Y., Liu, Q., Zhao, H., Pan, Z. and Liu, C., 2020, April. Adaptive quantitative trading: An imitative deep reinforcement learning approach. In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 02, pp. 2128-2135).

[214] Lo, A.W. and MacKinlay, A.C., 1988. Stock market prices do not follow random walks: Evidence from a simple specification test. The review of financial studies, 1(1), pp.41-66.

[215] Lo, Andrew W. "Long-term memory in stock market prices." Econometrica: Journal of the Econometric Society (1991): 1279-1313.

[216] Longstaff, F.A. and Schwartz, E.S., 2001. Valuing American options by simulation: a simple least-squares approach. The review of financial studies, 14(1), pp.113-147.

[217] Lu, D.W., 2017. Agent inspired trading using recurrent reinforcement learning and lstm neural networks. arXiv preprint arXiv:1707.07338.

[218] Lu, J.Y., Lai, H.C., Shih, W.Y., Chen, Y.F., Huang, S.H., Chang, H.H., Wang, J.Z., Huang, J.L. and Dai, T.S., 2022. Structural break-aware pairs trading strategy using deep reinforcement learning. The Journal of Supercomputing, 78(3), pp.3843-3882.

[219] Lucarelli, Giorgio, and Matteo Borrotti. "A deep reinforcement learning approach for automated cryptocurrency trading." In Artificial Intelligence Applications and Innovations: 15th IFIP WG 12.5 International Conference, AIAI 2019, Hersonissos, Crete, Greece, May 24–26, 2019, Proceedings 15, pp. 247-258. Springer International Publishing, 2019.

[220] Lucca, D.O. and Moench, E., 2015. The pre-FOMC announcement drift. The Journal of finance, 70(1), pp.329-371.

[221] Lux, T., 1995. Herd behaviour, bubbles and crashes. The economic journal, 105(431), pp.881-896.

[222] Mabu, S., Chen, Y., Hirasawa, K. and Hu, J., 2007, September. Stock trading rules using genetic network programming with actor-critic. In 2007 IEEE Congress on Evolutionary Computation (pp. 508-515). IEEE.

[223] Maringer, D. and Ramtohul, T., 2010, April. Threshold recurrent reinforcement learning model for automated trading. In European Conference on the Applications of Evolutionary Computation (pp. 212-221). Springer, Berlin, Heidelberg.

[224] Maringer, D. and Ramtohul, T., 2012. Regime-switching recurrent reinforcement learning for investment decision making. Computational Management Science, 9(1), pp.89-107.

[225] Maringer, D. and Zhang, J., 2014, March. Transition variable selection for regime switching recurrent reinforcement learning. In 2014 IEEE Conference on Computational Intelligence for Financial Engineering & Economics (CIFEr) (pp. 407-413). IEEE.

[226] Markowitz, H. M., 1952. Portfolio Selection. The Journal of Finance. 7 (1), 77–91. Doi: 10.2307/2975974. JSTOR 2975974.

[227] Martinez, L.C., da Hora, D.N., Palotti, J.R.D.M., Meira, W. and Pappa, G.L., 2009, June. From an artificial neural network to a stock market day-trading system: A case study on the bm&f bovespa. In 2009 International Joint Conference on Neural Networks (pp. 2006-2013). IEEE.

[228] Menzly, L. and Ozbas, O., 2010. Market segmentation and cross-predictability of returns. The Journal of Finance, 65(4), pp.1555-1580.

[229] Merton, R.C., 1973. Theory of rational option pricing. The Bell Journal of economics and management science, pp.141-183.

[230] Mihatsch, O. and Neuneier, R., 2002. Risk-sensitive reinforcement learning. Machine learning, 49, pp.267-290.

[231] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M., 2013. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

[232] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G. and Petersen, S., 2015. Human-level control through deep reinforcement learning. nature, 518(7540), pp.529-533.

[233] Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D. and Kavukcuoglu, K., 2016, June. Asynchronous methods for deep reinforcement learning. In International conference on machine learning (pp. 1928-1937). PMLR.

[234] Van Moffaert, K. and Nowé, A., 2014. Multi-objective reinforcement learning using sets of pareto dominating policies. The Journal of Machine Learning Research, 15(1), pp.3483-3512.

[235] Moody, J. and Wu, L., 1997, March. Optimization of trading systems and portfolios. In Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr) (pp. 300-307). IEEE.

[236] Moody, J., Wu, L., Liao, Y. and Saffell, M., 1998. Performance functions and reinforcement learning for trading systems and portfolios. Journal of Forecasting, 17(5-6), pp.441-470.

[237] Moody, J., Saffell, M., Liao, Y. and Wu, L., 1998. Reinforcement learning for trading systems and portfolios: Immediate vs future rewards. In Decision technologies for computational finance (pp. 129-140). Springer, Boston, MA.

[238] Moody, J. and Saffell, M., 2001. Learning to trade via direct reinforcement. IEEE transactions on neural Networks, 12(4), pp.875-889.

[239] Mosavi, A., Faghan, Y., Ghamisi, P., Duan, P., Ardabili, S.F., Salwana, E. and Band, S.S., 2020. Comprehensive review of deep reinforcement learning methods and applications in economics. Mathematics, 8(10), p.1640.

[240] Moulin, H., 2004. Fair division and collective welfare. MIT press.

[241] Murphy, J.J., 1999. Technical analysis of the financial markets: A comprehensive guide to trading methods and applications. Penguin.

[242] Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W. and Abbeel, P., 2018, May. Overcoming exploration in reinforcement learning with demonstrations. In 2018 IEEE international conference on robotics and automation (ICRA) (pp. 6292-6299). IEEE.

[243] Nan, A., Perumal, A. and Zaiane, O.R., 2022. Sentiment and knowledge based algorithmic trading with deep reinforcement learning. In International Conference on Database and Expert Systems Applications (pp. 167-180). Springer, Cham.

[244] Neuneier, R., 1996. Optimal asset allocation using adaptive dynamic programming. Advances in Neural Information Processing Systems, pp.952-958.

[245] Neuneier, R., 1998. Enhancing Q-learning for optimal asset allocation. In Advances in neural information processing systems (pp. 936-942).

[246] Nevmyvaka, Y., Feng, Y. and Kearns, M., 2006, June. Reinforcement learning for optimised trade execution. In Proceedings of the 23rd international conference on Machine learning (pp. 673-680).

[247] Ng, A.Y. and Russell, S., 2000, June. Algorithms for inverse reinforcement learning. In Icml (Vol. 1, p. 2).

[248] Ning, B., Lin, F.H.T. and Jaimungal, S., 2021. Double deep q-learning for optimal execution. Applied Mathematical Finance, 28(4), pp.361-380.

[249] Oikonomou, I., Brooks, C. and Pavelin, S., 2012. The impact of corporate social performance on financial risk and utility: A longitudinal analysis. Financial Management, 41(2), pp.483-515.

[250] Oord, A.V.D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K., 2016. Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.

[251] Opitz, D. and Maclin, R., 1999. Popular ensemble methods: An empirical study. Journal of artificial intelligence research, 11, pp.169-198.

[252] Ozbayoglu, A.M., Gudelek, M.U. and Sezer, O.B., 2020. Deep learning for financial applications: A survey. Applied Soft Computing, 93, p.106384.

[253] Pan, S.J. and Yang, Q., 2010. A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10), pp.1345-1359.

[254] Park, C.H. and Irwin, S.H., 2007. What do we know about the profitability of technical analysis?. Journal of Economic surveys, 21(4), pp.786-826.

[255] Park, H., Sim, M.K. and Choi, D.G., 2020. An intelligent financial portfolio trading strategy using deep Q-learning. Expert Systems with Applications, 158, p.113573.

[256] Patel, Y., 2018. Optimizing market making using multi-agent reinforcement learning. arXiv preprint arXiv:1812.10252.

[257] Pei, W., Baltrusaitis, T., Tax, D.M. and Morency, L.P., 2017. Temporal attention-gated model for robust sequence classification. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 6730-6739).

[258] Pendharkar, P.C. and Cusatis, P., 2018. Trading financial indices with reinforcement learning agents. Expert Systems with Applications, 103, pp.1-13.

[259] Peters, J., Janzing, D. and Schölkopf, B., 2017. Elements of causal inference: foundations and learning algorithms (p. 288). The MIT Press.

[260] Perkins, D.N. and Salomon, G., 1992. Transfer of learning. International encyclopedia of education, 2, pp.6452-6457.

[261] Peters, E.E., 1994. Fractal market analysis: applying chaos theory to investment and economics (Vol. 24). John Wiley & Sons.

[262] Pézier, J. and White, A., 2006. The relative merits of investable hedge fund indices and of funds of hedge funds in optimal passive portfolios (No. icma-dp2006-10). Henley Business School, Reading University.

[263] Plaat, A., 2022. Deep Reinforcement Learning. arXiv preprint arXiv:2201.02135.

[264] Polikar, R., 2006. Ensemble based systems in decision making. IEEE Circuits and systems magazine, 6(3), pp.21-45.

[265] Ponomarev, E.S., Oseledets, I.V. and Cichocki, A.S., 2019. Using reinforcement learning in the algorithmic trading problem. Journal of Communications Technology and Electronics, 64(12), pp.1450-1457.

[266] Qiu, Y., Qiu, Y., Yuan, Y., Chen, Z. and Lee, R., 2021. QF-TraderNet: Intraday Trading via Deep Reinforcement With Quantum Price Levels Based Profit-And-Loss Control. Frontiers in Artificial Intelligence, 4.

[267] Rafailov, R., Hatch, K.B., Kolev, V., Martin, J.D., Phielipp, M. and Finn, C., 2023, December. MOTO: Offline pre-training to online fine-tuning for model-based robot learning. In Conference on Robot Learning (pp. 3654-3671). PMLR.

[268] Rioul, O. and Duhamel, P., 1992. Fast algorithms for discrete and continuous wavelet transforms. IEEE transactions on information theory, 38(2), pp.569-586.

[269] , R.T. and Uryasev, S., 2000. Optimization of conditional value-at-risk. Journal of risk, 2, pp.21-42.

[270] Ross, S. and Bagnell, D., 2010, March. Efficient reductions for imitation learning. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 661-668). JMLR Workshop and Conference Proceedings.

[271] Ross, S., Gordon, G. and Bagnell, D., 2011, June. A reduction of imitation learning and structured prediction to no-regret online learning. In Proceedings of the fourteenth international conference on artificial intelligence and statistics (pp. 627-635). JMLR Workshop and Conference Proceedings.

[272] Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1985. Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science.

[273] Rummery, G.A. and Niranjan, M., 1994. On-line Q-learning using connectionist systems (Vol. 37, p. 20). Cambridge, England: University of Cambridge, Department of Engineering.

[274] Rusu, A.A., Colmenarejo, S.G., Gulcehre, C., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V., Kavukcuoglu, K. and Hadsell, R., 2015. Policy distillation. arXiv preprint arXiv:1511.06295.

[275] Sattarov, O., Muminov, A., Lee, C.W., Kang, H.K., Oh, R., Ahn, J., Oh, H.J. and Jeon, H.S., 2020. Recommending cryptocurrency trading points with deep reinforcement learning approach. Applied Sciences, 10(4), p.1506.

[276] Sawhney, R., Wadhwa, A., Agarwal, S. and Shah, R., 2021, June. Quantitative day trading from natural language using reinforcement learning. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 4018-4030).

[277] Schaul, T., Quan, J., Antonoglou, I. and Silver, D., 2015. Prioritized experience replay. arXiv preprint arXiv:1511.05952.

[278] Schmidhuber, J., 1987. Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook (Doctoral dissertation, Technische Universität München).

[279] Schulman, J., Levine, S., Abbeel, P., Jordan, M. and Moritz, P., 2015, June. Trust region policy optimization. In International conference on machine learning (pp. 1889-1897). PMLR.

[280] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O., 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[281] Sharpe, W. F.,1994. The Sharpe Ratio. The Journal of Portfolio Management. 21 (1), pp.49–58.

[282] Shavandi, A. and Khedmati, M., 2022. A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets. Expert Systems with Applications, 208, p.118124.

[283] Shen, Y., Huang, R., Yan, C. and Obermayer, K., 2014, March. Risk-averse reinforcement learning for algorithmic trading. In 2014 IEEE Conference on Computational Intelligence for Financial Engineering and Economics (CIFEr) (pp. 391-398). IEEE.

[284] Sherstov, A.A. and Stone, P., 2004, July. Three automated stock-trading agents: A comparative study. In International Workshop on Agent-Mediated Electronic Commerce (pp. 173-187). Springer, Berlin, Heidelberg.

[285] Shi, S., Li, J., Li, G. and Pan, P., 2019, November. A multi-scale temporal feature aggregation convolutional neural network for portfolio management. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 1613-1622).

[286] Si, W., Li, J., Ding, P. and Rao, R., 2017, December. A multi-objective deep reinforcement learning approach for stock index future's intraday trading. In 2017 10th International symposium on computational intelligence and design (ISCID) (Vol. 2, pp. 431-436). IEEE.

[287] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D. and Riedmiller, M., 2014, January. Deterministic policy gradient algorithms. In International conference on machine learning (pp. 387-395). PMLR.

[288] Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. and Dieleman, S., 2016. Mastering the game of Go with deep neural networks and tree search. nature, 529(7587), pp.484-489.

[289] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T. and Lillicrap, T., 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science, 362(6419), pp.1140-1144.

[290] Spooner, T., Fearnley, J., Savani, R. and Koukorinis, A., 2018. Market making via reinforcement learning. arXiv preprint arXiv:1804.04216.

[291] Spooner, T. and Savani, R., 2020. Robust market making via adversarial reinforcement learning. arXiv preprint arXiv:2003.01820.

[292] Spirtes, P., Glymour, C.N., Scheines, R. and Heckerman, D., 2000. Causation, prediction, and search. MIT press.

[293] Soleymani, F. and Paquet, E., 2021. Deep graph convolutional reinforcement learning for financial portfolio management–DeepPocket. Expert Systems with Applications, 182, p.115127.

[294] Sornmayura, S., 2019. Robust forex trading with deep q network (dqn). ABAC Journal, 39(1).

[295] Sortino, F.A. and Price, L.N., 1994. Performance measurement in a downside risk framework. the Journal of Investing, 3(3), pp.59-64.

[296] Stein, D.M. and Narasimhan, P., 1999. Of passive and active equity portfolios in the presence of taxes. The Journal of Wealth Management, 2(2), pp.55-63.

[297] Stiglitz, J.E. and Weiss, A., 1981. Credit rationing in markets with imperfect information. The American economic review, 71(3), pp.393-410.

[298] Such, F.P., Madhavan, V., Conti, E., Lehman, J., Stanley, K.O. and Clune, J., 2017. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. arXiv preprint arXiv:1712.06567.

[299] Sullivan, R., Timmermann, A. and White, H., 1999. Data-snooping, technical trading rule performance, and the bootstrap. The journal of Finance, 54(5), pp.1647-1691.

[300] Sutton, R.S., 1984. Temporal credit assignment in reinforcement learning. University of Massachusetts Amherst.

[301] Sutton, R.S., 1988. Learning to predict by the methods of temporal differences. Machine learning, 3(1), pp.9-44.

[302] Sutton, R.S. and Barto, A.G., 2018. Reinforcement learning: An introduction. MIT press.

[303] Sutskever, I., Vinyals, O. and Le, Q.V., 2014. Sequence to sequence learning with neural networks. Advances in neural information processing systems, 27.

[304] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[305] Taghian, M., Asadi, A. and Safabakhsh, R., 2022. Learning financial asset-specific trading rules via deep reinforcement learning. Expert Systems with Applications, 195, p.116523.

[306] Tan, M., 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In Proceedings of the tenth international conference on machine learning (pp. 330-337).

[307] Tan, Z., Quek, C. and Cheng, P.Y., 2011. Stock trading with cycles: A financial application of ANFIS and reinforcement learning. Expert Systems with Applications, 38(5), pp.4741-4755.

[308] Tavakoli, A., Pardo, F. and Kormushev, P., 2018, April. Action branching architectures for deep reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

[309] Tetlock, P.C., 2007. Giving content to investor sentiment: The role of media in the stock market. The Journal of finance, 62(3), pp.1139-1168.

[310] Tetlock, P.C., Saar-Tsechansky, M. and Macskassy, S., 2008. More than words: Quantifying language to measure firms' fundamentals. The journal of finance, 63(3), pp.1437-1467.

[311] Théate, T. and Ernst, D., 2021. An application of deep reinforcement learning to algorithmic trading. Expert Systems with Applications, 173, p.114632.

[312] Thrun, S., 1995. Is learning the n-th thing any easier than learning the first?. Advances in neural information processing systems, 8.

[313] Tsantekidis, A., Passalis, N., Toufa, A.S., Saitas-Zarkias, K., Chairistanidis, S. and Tefas, A., 2020. Price trailing for financial trading using deep reinforcement learning. IEEE Transactions on Neural Networks and Learning Systems, 32(7), pp.2837-2846.

[314] Tsantekidis, A., Passalis, N. and Tefas, A., 2021. Diversity-driven knowledge distillation for financial trading using Deep Reinforcement Learning. Neural Networks, 140, pp.193-202.

[315] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems, 30.

[316] Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P. and Oh, J., 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature, 575(7782), pp.350-354.

[317] Vittori, E., Trapletti, M. and Restelli, M., 2020, October. Option hedging with risk averse reinforcement learning. In Proceedings of the First ACM International Conference on AI in Finance (pp. 1-8).

[318] Wang, Z. and Oates, T., 2015, April. Encoding time series as images for visual inspection and classification using tiled convolutional neural networks. In Workshops at the twenty-ninth AAAI conference on artificial intelligence.

[319] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M. and Freitas, N., 2016, June. Dueling network architectures for deep reinforcement learning. In International conference on machine learning (pp. 1995-2003). PMLR.

[320] Wang, J., Gu, Q., Wu, J., Liu, G. and Xiong, Z., 2016, December. Traffic speed prediction and congestion source exploration: A deep learning method. In 2016 IEEE 16th international conference on data mining (ICDM) (pp. 499-508). IEEE.

[321] Wang, Y., Wang, D., Zhang, S., Feng, Y., Li, S. and Zhou, Q., 2017. Deep Q-trading. cslt. riit. tsinghua. edu. cn.

[322] Wang, J., Wang, Z., Li, J. and Wu, J., 2018, July. Multilevel wavelet decomposition network for interpretable time series analysis. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 2437-2446).

[323] Wang, J., Zhang, Y., Tang, K., Wu, J. and Xiong, Z., 2019, July. Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks. In Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 1900-1908).

[324] Wang, H., 2019. Large scale continuous-time mean-variance portfolio allocation via reinforcement learning. arXiv preprint arXiv:1907.11718.

[325] Wang, H. and Zhou, X.Y., 2020. Continuous-time mean–variance portfolio selection: A reinforcement learning framework. Mathematical Finance, 30(4), pp.1273-1308.

[326] Wang, Z., Huang, B., Tu, S., Zhang, K. and Xu, L., 2021, May. DeepTrader: A Deep Reinforcement Learning Approach for Risk-Return Balanced Portfolio Management with Market Conditions Embedding. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 1, pp. 643-650).

[327] Wang, R., Wei, H., An, B., Feng, Z. and Yao, J., 2021, May. Commission fee is not enough: A hierarchical reinforced framework for portfolio management. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 1, pp. 626-633).

[328] Wang, C., Sandås, P. and Beling, P., 2021, May. Improving pairs trading strategies via reinforcement learning. In 2021 International Conference on Applied Artificial Intelligence (ICAPAI) (pp. 1-7). IEEE.

[329] Wang, H. and Yu, S., 2021, December. Robo-advising: Enhancing investment with inverse optimization and deep reinforcement learning. In 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 365-372). IEEE.

[330] Watkins, C.J.C.H., 1989. Learning from delayed rewards.

[331] Watkins, C.J. and Dayan, P., 1992. Q-learning. Machine learning, 8(3-4), pp.279-292.

[332] Wei, H., Wang, Y., Mangu, L. and Decker, K., 2019. Model-based reinforcement learning for predictions and control for limit order books. arXiv preprint arXiv:1910.03743.

[333] Weng, L., Sun, X., Xia, M., Liu, J. and Xu, Y., 2020. Portfolio trading system of digital currencies: A deep reinforcement learning with multidimensional attention gating mechanism. Neurocomputing, 402, pp.171-182.

[334] Weymark, J.A., 1981. Generalized Gini inequality indices. Mathematical Social Sciences, 1(4), pp.409-430.

[335] Williams, R.J., 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3), pp.229-256.

[336] Wohlin, C., 2014, May. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In Proceedings of the 18th international conference on evaluation and assessment in software engineering (pp. 1-10).

[337] Wolpert, D.H., 1992. Stacked generalization. Neural networks, 5(2), pp.241-259.

[338] Wu, X., Chen, H., Wang, J., Troiano, L., Loia, V. and Fujita, H., 2020. Adaptive stock trading strategies with deep reinforcement learning methods. Information Sciences, 538, pp.142-158.

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

[340] Xu, K., Zhang, Y., Ye, D., Zhao, P. and Tan, M., 2021, January. Relation-aware transformer for portfolio policy learning. In Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence (pp. 4647-4653).

[341] Yang, D. and Zhang, Q., 2000. Drift-independent volatility estimation based on high, low, open, and close prices. The Journal of Business, 73(3), pp.477-492.

[342] Yang, S.Y., Yu, Y. and Almahdi, S., 2018. An investor sentiment reward-based trading system using Gaussian inverse reinforcement learning algorithm. Expert Systems with Applications, 114, pp.388-401.

[343] Yang, H., Liu, X.Y., Zhong, S. and 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).

[344] Ye, Y., Pei, H., Wang, B., Chen, P.Y., Zhu, Y., Xiao, J. and Li, B., 2020, April. Reinforcement-learning based portfolio management with augmented asset movement prediction states. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 01, pp. 1112-1119).

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

[346] Yu, S., Chen, Y. and Dong, C., 2020. Learning time varying risk preferences from investment portfolios using inverse optimization with applications on mutual funds. arXiv preprint arXiv:2010.01687.

[347] Yuan, Y., Wen, W. and Yang, J., 2020. Using data augmentation based reinforcement learning for daily stock trading. Electronics, 9(9), p.1384.

[348] Zarkias, K.S., Passalis, N., Tsantekidis, A. and Tefas, A., 2019, May. Deep reinforcement learning for financial trading using price trailing. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 3067-3071). IEEE.

[349] Zhang, J. and Maringer, D., 2013, July. Indicator selection for daily equity trading with recurrent reinforcement learning. In Proceedings of the 15th annual conference companion on Genetic and evolutionary computation (pp. 1757-1758).

[350] Zhang, J. and Maringer, D., 2014, July. Two parameter update schemes for recurrent reinforcement learning. In 2014 IEEE Congress on Evolutionary Computation (CEC) (pp. 1449-1453). IEEE.

[351] Zhang, J. and Maringer, D., 2016. Using a genetic algorithm to improve recurrent reinforcement learning for equity trading. Computational Economics, 47(4), pp.551-567.

[352] Zhang, Z., Zohren, S. and Roberts, S., 2020. Deep reinforcement learning for trading. The Journal of Financial Data Science, 2(2), pp.25-40.

[353] Zhang, Y., Zhao, P., Li, B., Wu, Q., Huang, J. and Tan, M., 2020. Cost-sensitive portfolio selection via deep reinforcement learning. IEEE Transactions on knowledge and data engineering.

[354] Zhao, M. and Linetsky, V., 2021, November. High frequency automated market making algorithms with adverse selection risk control via reinforcement learning. In Proceedings of the Second ACM International Conference on AI in Finance (pp. 1-9).

[355] Zhao, K., Ma, Y., Liu, J., Jianye, H.A.O., Zheng, Y. and Meng, Z., 2023, July. Improving Offline-to-Online Reinforcement Learning with Q-Ensembles. In ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems.

[356] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H. and He, Q., 2020. A comprehensive survey on transfer learning. Proceedings of the IEEE, 109(1), pp.43-76.

[357] Zhong, Y., Bergstrom, Y.M. and Ward, A., 2021, January. Data-driven market-making via model-free learning. In Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence (pp. 4461-4468).

[358] Zhou, Z.H., 2012. Ensemble methods: foundations and algorithms. CRC press.

[359] Zhu, Y., Yang, H., Jiang, J. and Huang, Q., 2018, December. An adaptive box-normalization stock index trading strategy based on reinforcement learning. In International Conference on Neural Information Processing (pp. 335-346). Springer, Cham.