International Journal of Information Management Data Insights 2 (2022) 100094

Contents lists available at ScienceDirect

International Journal of Information Management Data Insights

journal homepage: www.elsevier.com/locate/jjimei

How are reinforcement learning and deep learning algorithms used for big data based decision making in financial industries–A review and research agenda

Vinay Singh [a,b,e,*], Shiuann-Shuoh Chen [b], Minal Singhania [a], Brijesh Nanavati [c], Arpan kumar kar [d], Agam Gupta [d]

[a] BASF SE, Pfalzgrafenstraße 1, 67061 Ludwigshafen am Rhein, Germany
[b] Department of Business Administration, National Central University, No. 300, Zhongda Road, Zhongli District, Taoyuan City 320, Taiwan
[c] BASF Services Europe GmbH, Rotherstrasse 11, 10245 Berlin, Germany
[d] Department of Management Studies and School of Artificial Intelligence, Indian Institute of Technology Delhi, New Delhi 110016, India
[e] Universität Siegen, Adolf-Reichwein-Straße 2, 57076 Siegen

ARTICLE INFO

Keywords:
Big data
Markov decision process
Online learning
Reinforcement learning
Financial applications
Deep reinforcement learning

ABSTRACT

Data availability and accessibility have brought in unseen changes in the finance systems and new theoretical and computational challenges. For example, in contrast to classical stochastic control theory and other analytical approaches for solving financial decision-making problems that rely heavily on model assumptions, new developments from reinforcement learning (RL) can make full use of a large amount of financial data with fewer model assumptions and improve decisions in complex economic environments. This paper reviews the developments and use of Deep Learning(DL), RL, and Deep Reinforcement Learning (DRL)methods in information-based decision-making in financial industries. Therefore, it is necessary to understand the variety of learning methods, related terminology, and their applicability in the financial field. First, we introduce Markov decision processes, followed by Various algorithms focusing on value and policy-based methods that do not require any model assumptions. Next, connections are made with neural networks to extend the framework to encompass deep RL algorithms. Finally, the paper concludes by discussing the application of these RL and DRL algorithms in various decision-making problems in finance, including optimal execution, portfolio optimization, option pricing, hedging, and market-making. The survey results indicate that RL and DRL can provide better performance and higher efficiency than traditional algorithms while facing real economic problems in risk parameters and ever-increasing uncertainties. Moreover, it offers academics and practitioners insight and direction on the state-of-the-art application of deep learning models in finance.

1. Introduction

Machine learning(ML) based application has exploded in the past decade; almost everyone interacts with modern artificial intelligence many times every day. ML methods enable machines to conduct complex tasks such as detecting faces, understanding speech, recommending alternatives, targeted marketing or finding anomalies in business processes (Sharma et al., 2021; Singh et al., 2022; Verma et al., 2021). However, financial institutions cannot comprehend medium and large enterprises' static and dynamic economic development business information (Theate &, Ernst,2021). As a result, they are not always aware of the decisions made by investors for non-entirely rational people. Therefore, machine learning methods generally decide based on known properties learned from the training data. Using many principles and tools from statistics.

However, machine learning models aspire to a generalized predictive pattern. For example,most learning problems could be seen as optimizing a cost: minimizing a loss or maximizing a reward. But learning algorithms seek to optimize a criterion (loss, reward, regret) on training and unseen samples(Khadjeh et al., 2014). Financial decision-making problems have traditionally been modeled using stochastic processes and methods arising from stochastic control. However, the availability of large amounts of financial data has revolutionized data processing and statistical modeling techniques in Finance and brought new theoretical and computational challenges. In contrast, the classical stochastic control approach describes how agents acting within some system might

* Corresponding author: Vinay Singh, BASF SE, Pfalzgrafenstraße 1, 67061 Ludwigshafen am Rhein, Germany.
E-mail address: vinaysingh.bvp@gmail.com (V. Singh).

https://doi.org/10.1016/j.jjimei.2022.100094
Received 8 February 2022; Received in revised form 1 June 2022; Accepted 19 June 2022

learn to make optimal decisions through repeated experience gained by interacting with the system.

RL is a robust mathematical framework where the agents interact directly with the environment. It is experience-driven autonomous learning where the agent enhances its efficiency by trial-and-error to optimize the cumulative reward. It does not require labeled data to do so. For Autonomous learning policy, search and value function approximation are vital tools. RL applies a gradient-based or gradient-free approach to detect an optimal stochastic policy for continuous and discrete state-action settings(Sutton and Barto,2014). While considering an economic problem, despite traditional approaches, reinforcement learning methods prevent the suboptimal performance by imposing significant market constraints that lead to finding an optimal strategy in terms of market analysis and forecast. Despite RL successes in recent years, these results lack scalability and cannot manage high-dimensional problems. By combining both RL and DL methods, the DRL technique, where DL is equipped with the vigorous function approximation, representation learning properties of deep neural networks (DNN), and handling complex and nonlinear patterns of economic data, can efficiently overcome these problems.

This study contributes to the literature in the following ways. First, we systematically review the state-of-the-art applications of Reinforcement Learning in the field of Finance. Second, we summarize multiple RL models regarding specified Finance domains and identify the optimal RL model for application scenarios. The study uses the data processing method as the basis for analysis. Third, our review attempts to bridge the technological and application levels of RL and Finance, respectively. Fourth, we identify the features of various RL models and highlight their feasibility toward different Finance domains. This will help Researchers about the feasibilities of RL models toward a specified financial field. Due to the lack of connections between core economic domains and numerous RL models, they usually face difficulties. This study will fill this literature gap and guide financial analysts. we are guided by the following research questions(s):

RQ1. How and in which area of Finance can we use RL and DRL models to make better-informed decisions?
RQ2. How Can the existing DL, RL, and DRL approaches, based on their application, be classified for better understanding and application?
RQ3. What are the open issues and challenges in current deep reinforcement learning models for information-based decision-making in Finance?

The rest of this paper is organized as follows. Section 2 provides a background of DL, RL, and DRL techniques. Section 3 introduces our research framework and methodology. Section 4 analyzes the established RL models. Section 5 examines Deep Reinforcement learning methods. Section 6 captures a review of existing Financial applications using various RL, DL, and DRL methods. Section 7 discusses the influencing factors in the performance of financial RL models. Finally, Section 8concludes and outlines the scope for promising future studies.

2. Preliminary discussion on terms related to reinforcement learning

Since we aim to use RL for our study, we will first do a preliminary discussion on terms from RL and understand them in the context of finance. We will emphasize the idea of exploring and exploiting and the critical role it plays. We will conclude the section with a clear understanding of Reinforcement learning and deep reinforcement learning and how deep reinforcement learning can help complement traditional RL.

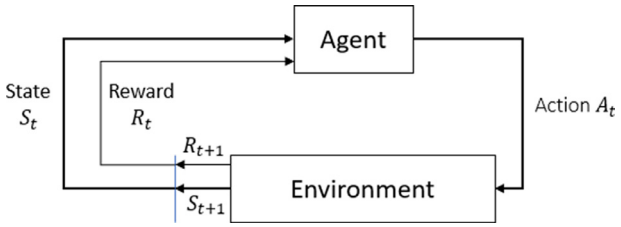

Fig. 1. Agent-environment interaction in an MDP (adopted from Sutton and Barto, 2016).

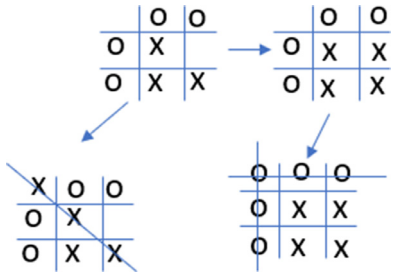

Fig. 2. Representation of state and possible action (one of the cases) in game of tic-tac-toe.

2.1. Markov decision process

Markov decision process (MDP) is the classical formalisation of sequential decision making, where actions influence not just immediate rewards, but also subsequential decision making, where actions influence not just the immediate rewards but also subsequent situations or states and through those future rewards (Puterman, 2014; *Van* Otterlo & Wiering, 2012). It involves delayed rewards and the need to tradeoff immediate and delayed rewards.

2.1.1. Introduction to markov decision processes

MDPs are mathematically idealized forms for decision-theoretic planning (DTP), reinforcement learning, and other learning problems in stochastic domains for which precise theoretical statements can be made (Sutton and Barto,2016).

MDPs are meant to be a precise framing of learning from interaction to achieve a goal. The learner and decision-maker are agents *(Van* Otterlo & Wiering, 2012*)*. It interacts with, comprising everything outside the agent, which is called the *environment (Van* Otterlo & Wiering, 2012*)*. These interact continually, the agent selecting actions and the environment responding to these actions and presenting new situations to the agent. The environment also gives rise to rewards, special numerical values that the agent seeks to maximize over time through its choice of actions*. MDPs consist of states, actions, transitions between states, and a reward function definition (Puterman,2014).Figure 1 shows an agent-environment interaction in an MDP.

2.1.2. States

A state is a unique characterization of all that is important in a state of the modeled problem (*Van* Otterlo & Wiering, 2012). It can be defined as the finite set {} where the size of the state space is N, i.e., $|S| = N$. For instance, in the tic-tac-toe game (see Figure below2 on the right), a state could be a tuple of 9 components corresponding to each cell of the game's board. It may contain either 0, 1, or 2 (or any other three different symbols) depending on if the cell is empty or occupied by one of the players. In our research, we confine ourselves to the discrete state set S in which each state is represented by a distinct symbol and, all states $s \in S$ are legal. Figure 2 shows a typical state in a tic-tac-toe games with possible actions

2.1.3. Action

Actions can be used to control the system state (Puterman,2014). The set of actions that can be applied in some state's ∈ S is denoted A(s), where A(s)⊆A. The set of actions A is defined as the finite set {, . . .,} where the size of the action space is K, i.e., |A| = K. Though not all actions can be applied in every state, in general, we will assume that A(s)=*A* for all s∈ S.

2.1.4. Transition function

By applying an action, $a_n \in$ A in a state ∈ S, the system transitions from s to a new state ∈ S, based on a probability distribution over the set of possible transitions (Puterman,2014). The transition function T is defined as:

$$T : S \times A \times S \rightarrow [0, 1] \quad (1)$$

i.e., the probability of ending up in state $s\prime$ after doing action a in state s is denoted T(s, a,$s\prime$). It is required that for all actions a, and all states s and $s\prime$:

T(s, a,$s\prime$) ≥ 0 and T(s, a,$s\prime$) ≤ 1.

Furthermore, for all states s and actions a,

$$\Sigma_{s\prime \in S} T(s, a, s\prime) = 1 \quad (2)$$

i.e., T defines a proper probability distribution over the possible next state. For talking about the order in which actions occur, one defines a discrete global clock, $t \in \{0, 1, 2, \ldots\}$, so that it denotes the state and action at time *t*, respectively. The system being controlled is Markovian* if the result of an action does not depend on the previous actions and visited states but only depends on the current state (Otterlo & Wiering, 2012)

$$P(s_{t+1}|s_t, a_t, s_{t+1}, a_{t-1}, \ldots) = P(s_{t+1}|s_t, a_t,) = T\left(s_t, a_t, s_{t+1},\right) \quad (3)$$

2.1.5. Reward function

Rewards are used to determine on how the MDP system should be controlled (*Van* Otterlo & Wiering, 2012). The agent should control the system by taking actions that lead to more (positive) rewards over time. For example, in classical optimization problems, we try to find cost functions and aim to maximize or minimize them under certain conditions (Sutton and Barto,2016). In RL, the agent's goal is formalized by a special signal (rewards) passing from the environment to the agent. A reward can be defined as:

$$R : S \times A \times S \rightarrow IR \quad (4)$$

2.1.6. Markova decision process

A Markov decision process (MDP) is a tuple (S, A, T, R) in which S is a finite set of states, A is a finite set of actions, T a transition probability function T: $S \times A \times S$! [0, 1] and R a reward function, R: $S \times A \times S$ à ℝ. Let's take an example of moving a pawn to a destination on a grid (as shown in below Fig. 3):

Markov Decision Process for the above figure can be stated as:

- States $S = \{s_0, s_1, s_2, \ldots..s_7\}$
- Actions $A = \{up, down, left, right\}$, depends on current state S.
- Transition probabilities = $\{T^{up}_{s_0,s_3} = 0.9; T^{right}_{s_0,s_1} = 0.1; \ldots..\}$
- Rewards = $\{R^{right}_{s_6,s_3} = +10; R^{up}_{s_2,s_4} = -10;$ otherwise, $R = 0\}$
- Start in s_0
- Game over when reaching s_7

With the above definition of an MDP, we can model several distinct systems like episodic and continuing tasks. First, there is the notion of episodes of some length _ where the goal is to take the agent from some starting state to a goal state. In these cases, one can distinguish between fixed horizon tasks in which each episode consists of a fixed number of steps or indefinite horizon tasks with an end. Still, episodes can have an arbitrary length (Otterlo & Wiering, 2012). An example for the former type would be tic-tac-toe, and for the latter, chess board game. Another

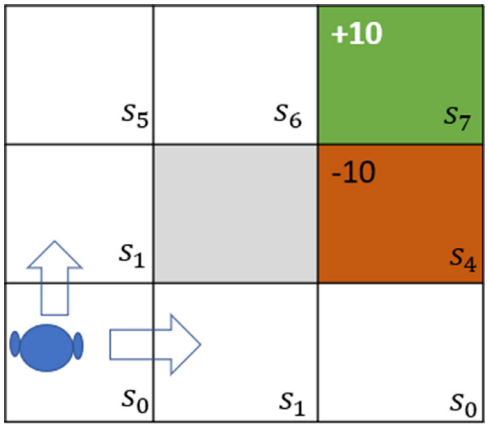

Fig. 3. Moving a pawn to a destination on a grid.

example of indefinite horizon tasks would be a robot trying to learn how to run or walk. In the beginning, it will fall early (so that episodes will be short), but if it is learning properly, at some point, it will be able to run or walk forever. The system's initial state is initialized with some initial state distribution in each episode.

$$I : S \rightarrow [0, 1] \quad (5)$$

In continuing or *infinite horizon* tasks, the system does not end unless done on purpose (theoretically, it never ends). A possible example would be software dedicated to sustaining the energy supply for a city by controlling the power plants (Sutton and Barto,2016).

2.1.7. Policies

Given an MDP(*S,A, T, R*), a policy is a computable function that outputs for each State's ∈ *S* an action $a \in A$ (or $a \in A(s)$). Formally, a *deterministic* policy π is a function defined as π: $S \rightarrow A$. It is also possible to define a *stochastic* policy as π: $S \times A \rightarrow$ [0,1] such that for each state's ∈ *S*, it holds that π (*s, a*) ≥ 0 and $\Sigma a \in A \pi$ (*s, a*) = 1. A policy π can be used to make evolve, i.e., to control, an MDP system in the following way:

- Starting from an initial state $s_0 \in$ S, the next action the agent will do is taken as $a_0 = \pi$ (s_0).
- After the action is performed by the agent, according to the transition probability function *T* and the reward function *R*, a transition is made from *s*0 to some state *s*1, with probability $T(s_0, a, s_1)$ and an obtained reward $r_0 = R(s_0, a_0, s_1)$.
- By iterating this process, one obtains a sequence $s_0, a_0, r_0, s_1, a_1, r_1, \ldots$ of state-action-reward triples over $S \times A \times \mathbb{R}$ which constitute a trajectory (or path) of the MDP.

The policy is part of the agent with the aim to control the environment modelled as an MDP.

MDPs provide a mathematical framework for modeling decision-making in situations where outcomes are partly random and partly under the control of a decision-maker, which could be very useful for studying optimization problems solved via dynamic programming in finance.

2.2. Exploration versus exploitation

To maximize the accumulated reward over time, the agent learns to select her actions based on her past experiences (exploitation) and/or by trying new choices (exploration). Exploration provides opportunities to improve performance from the current sub-optimal solution to the ultimate globally optimal one. Yet, it is time-consuming and computationally expensive as over-exploration may impair the convergence to the optimal solution. Meanwhile, pure exploitation., myopically picking the current solution based solely on experience, though easy to im-

plement, yields sub-optimal global solutions. Therefore, an appropriate trade-off between exploration and exploitation is crucial in designing RL algorithms to improve learning and optimization performance. Bergemann & Vlimki(1996) provided a friendly economic application of the exploration-exploitation dilemma. In this model, the actual value of each seller's product to the buyer is initially unknown, but additional information can be gained by experimentation. When assuming that prices are given exogenously, the buyer's problem is a common multi-armed bandit problem.

The exploration-exploitation dilemma is a very general problem that can be encountered in most data-driven decision-making processes when there is some feedback loop between data gathering and decision-making. An efficient approach addresses this dilemma.

2.3. Classification of reinforcement learning algorithms

Broadly there are two main branching points in an RL algorithm – *whether the agent has access to (or learns) a model of the environment* and the second being *what to learn*. Based on these two aspects, an RL algorithm will include one(or more) of the following components (Hambly et al., 2021):

- *A value function that provides a prediction of how good each state(state-action) pair is*,
- *Representation of the policy*,
- *A model of the environment.*

The first two components are related to model-free RL, whereas when the model of the environment is available, the algorithm is referred to as model-based RL. In the MDP setting, model-based algorithms maintain an approximate MDP model by estimating the transition probabilities and the reward function and deriving a value function from the approximate MDP. The policy is then derived from the value function. Examples include (Theate et al., 2021; Sutton and Barto,2016; Jiang & Liang,2017; Jiang et al., 2017). Another line of model-based algorithms makes structural assumptions on the model, using prior knowledge and utilizing the structural information for algorithm design.

Unlike the model-based method, model-free algorithms directly learn a value (or state-value) function or the optimal policy without inferring the model. Model-free algorithms can be further divided into two categories, value-based methods, and policy-based methods. Policy-based methods explicitly build a policy representation and keep it in memory during learning. Examples include policy gradient methods and trust-region policy optimization methods. As an alternative, value-based methods store only a value function without an explicit policy during the learning process (Singh and Sharma, 2022). Here, the policy is implicit and can be derived directly by picking the action with the best value.

Based on the problem, one can choose a model-based or model-free approach. Model-based methods rely on planning as their primary component, while model-free methods primarily rely on learning.

2.4. Optimal policies, value functions and bellman equations

To solve a reinforcement learning problem, we need to find a policy that gets a lot of rewards (over a period). So, defining a model of Optimality would be a necessary step in this regard. The approach can have two aspects - the goal of the agent's goal, what is being Optimized, and the second aspect is how optimal the way in which the goal is being Optimized. The first aspect focuses on gathering rewards, while the second aspect is related to the efficiency and Optimality of algorithms.

There are three models of Optimality in the MDP related to the types of tasks in hand – Finite Horizon, Discounted (Infinite Horizon), and Average rewards (*Van Otterlo and Wiering 2016*). IN our study, we will focus on discounted average reward criteria for Finite and Infinite Horizon (Koenig et al., 2019). The discounted sum of rewards by the agent can be defined as:

$$R_t^{\Upsilon} = \sum_{k=0}^{T-t} \Upsilon^k R_{t+k} \;=\; R_t + \Upsilon R_{t+1} + \Upsilon^2 R_{t+2} + \cdots + \Upsilon^{T-t} R_T \tag{6}$$

Where $T<\infty$ for episodic tasks and $T=\infty$ for continuing tasks, $\Upsilon \in [0,1]$ *is a discounted factor for future rewards ($\Upsilon<1$ if the task is continuing).*

One should make a note that Υ plays an important role in future rewards .The agent will be myopic if $\Upsilon = 0$ ($considering\ \Upsilon^0 = 1$) because the only reward (when we put $\Upsilon = 0$ in the above equation) would be the present reward. Whereas if $\Upsilon \approx 1$ then the agent will look for long-term rewards. For the episodic tasks with $\Upsilon = 1$, R_t^{Υ} is the sum of rewards at each step of an episode. The goal of the discounted average reward criteria in the context of MDP is to find a policy π^* that maximizes the expected return $\mathbb{E}_{\pi}[R_t^{\Upsilon}]$.The expectation $\mathbb{E}_{\pi}$ denotes the expected value when the MDP is controlled by policy π.To link the optimality criteria to policies we use values functions. It's an estimate of how good it is for the agent to be in a certain state (State value function, V^{π}) or to make a certain action when being some state (action-value function, Q^{π}). The **state value function $V^{\pi}(s)$**of an MDP is the expected reward starting from state s, and then following once policy π (otterlo-wiering,2012). While considering the discounted average reward,it can be represented as:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[R_t^{\Upsilon} | S_t = s\right] = \mathbb{E}_{\pi}\left[\sum_{k=0}^{T-t} \Upsilon^k R_{t+k} | S_t = s\right] \tag{7}$$

Where $T<\infty$ for episodic tasks and $T=\infty$ for continuing tasks.

Whereas the state-action value function $Q^{\pi}(s,a)$ is the expected reward starting from state s, taking action a, and then following policy π. Considering the discounted average reward,it can be represented as:

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[R_t^{\Upsilon} | S_t = s, A_t = a\right] = \mathbb{E}_{\pi}\left[\sum_{k=0}^{T-t} \Upsilon^k R_{t+k} | S_t = s, A_t = a\right] \tag{8}$$

Where $T<\infty$ for episodic tasks and $T=\infty$ for continuing tasks.

While π can be any policy, $\pi *$ denotes the optimal one with the highest expected cumulative reward i.e.$V^{\pi *}(s) > V^{\pi}(s)$ for all s∈S and all policies π. So,

$$V^*(s) \;=\; \max_{\pi} V^{\pi}(s) \tag{9}$$

Similarly, optimal Q-value Q^*:

$$Q^*(s,a) \;=\; \max_{\pi} Q^{\pi}(s,a) \tag{10}$$

One fundamental property of value function is that they satisfy certain recursive properties. For any policy π and any MDP (S,A,T,R),the value functions (V^{π} *and* Q^{π}) can be recursively defines in terms of so-called Bellman Equation (Bellman,2003):

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[R_t \; + \Upsilon V^{\pi}\left(S_{t+1}\right) \; | S_t = s\right] \tag{11}$$

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[R_t \; + \Upsilon Q^{\pi}\left(S_{t+1}, A_{t+1}\right) \; | S_t = s, A_t = a\right] \tag{12}$$

*for all s∈**S** and a∈**A**.*

Bellman equations are used for deriving algorithms such as Q-learning which we have also used in solving the optimization problem.

2.5. Generalized policy iteration

It is the idea of alternating between *any* method for Policy Evaluation and *any* method for Policy Improvement, including methods that are partial applications of Policy Evaluation or Policy Improvement. This generalized perspective unifies almost all algorithms that solve MDP Control problems(Sutton and Barto,2016). Generalized Policy Iteration is the idea that we can evaluate the Value Function for a policy with *any* Policy Evaluation method. We can improve a policy with *any* Policy Improvement method and not necessarily the classical Policy Iteration DP algorithm. We want to emphasize that neither Policy Evaluation nor

Policy Improvement needs to go entirely towards the notion of consistency they are striving for. As a simple example, think of modifying Policy Evaluation (say for a policy π) not to go all the way to V π, but instead perform, say, 3 Bellman Policy Evaluations. This means it would partially bridge the gap on the first notion of consistency (getting closer to $\boldsymbol{V}\ \pi$ but not going all the way to $\boldsymbol{V}\ \pi$), but it would also mean not slipping up too much on the second notion of consistency. As another example, think of updating just 5 of the states (say in large state space) with the Greedy Policy Improvement function (rather than the normal Greedy Policy Improvement function that operates on all the states). This means it would partially bridge the gap on the second notion of consistency (getting closer to $G(\boldsymbol{V}\ \pi)$ but not going all the way to $G(\boldsymbol{V}\ \pi)$), but it would also mean not slipping up too much on the first notion of consistency. A concrete example of Generalized Policy Iteration is Value Iteration. In Value Iteration, we apply the bellman policy Operator just once before moving on to policy improvement. Almost all control algorithms in Reinforcement Learning can be viewed as special cases of Generalized Policy Iteration. In some simple versions of Reinforcement Learning Control algorithms, the Policy Evaluation step is done for just a single state (versus all states in usual Policy Iteration, or even in Value Iteration). The Policy Improvement step is also done for just a single state.

So, these Reinforcement Learning Control algorithms are essentially an alternating sequence of single-state policy evaluation and single-state policy improvement (where the single-state is the state produced by sampling or the state encountered in a real-world environment interaction).

2.6. Reinforcement learning and deep reinforcement learning

In a reinforcement learning system, input and output pairs are not provided. Instead, the system's current state is given a specific goal and set of allowable actions and environmental constraints for their outcomes. The agent interacts with the environment through trial and error and learns to optimize the maximum reward. Reinforcement learning models have been applied successfully in closed-world environments such as games (Silver et al., 2018). Still, they are also relevant for multi-agent systems such as electronic markets. Popular RL algorithms use functions Q(s, a) or V(s) to estimate the sum of discounted rewards. As the function is defined by a tabular mapping of discrete inputs and outputs, this is limiting for continuous states or many states. However, a neural network can estimate the states (Chen et al., 2021). Deep reinforcement learning uses functional approximation instead of the tabular estimating state value(reference). Functional approximation eliminates the need to store all state and value pairs in a table and enables the agent to generalize the value of states it has never seen before or has partial information about by using the values of similar states(reference). So RL dynamically learns with trial-and-error methods to maximize the rewards, while deep reinforcement learns from existing knowledge and applies it to a new data set.

3. Methodology

3.1. Research methodology

The review adopts the Prisma standard to identify and review the RL and DRL methods used in Finance. The Prisma method includes four steps (Moher et al., 2015): Identification, screening, eligibility, and inclusion. Web-of Science and Elsevier Scoops, 460 of the most relevant articles are identified between 2014 and November 2021. The screening step includes two sub-steps. First, duplicate articles are eliminated, resulting in260 unique articles only to move for examination of relevance based on title and abstract. As a result, we got 86 articles for further consideration. Then as the next step in Prisma, we looked for eligibility where we had read the full text of these articles, and finally, we had 64 of them considered for final review in this study. As the final step of the Prisma model, we created the study database, which we have used for the quantitative and qualitative analyses. Our research database comprises 64 articles, and all our analyses and finding are based on these articles. Fig. 4 below shows the steps in creating our research database based on the Prisma model.

3.2. Taxonomy of widely applied RL and DRL in finance

Broadly there are two main branching points in an RL algorithm – *whether the agent has access to (or learns) a model of the environment* and the second being *what to learn*. Based on these two questions, the below Figure 5 shows a taxonomy of algorithms in modern RL (Lillicrap et al., 2015; Mnih et al., 2016; (Schulman et al., 2015) ; (Silver et al., 2018) ; Fujimoto et al., 2018; Haarnoja et al., 2018;Ha & Schmidhuber, 2018)

4. Discussion based on reinforcement learning approaches and algorithms

4.1. RL approaches

4.1.1. Critic-only approach

The critic-only approach is the most frequent application of RL in finance(reference). This approach aims to learn a value function based on which the agent can compare ("criticize") the expected outcomes of different actions. Then, during the decision-making process, the agent senses the current state of the environment and selects the action with the best outcome according to the value function. The reward function in the critic-only approach does not need to be differentiable and is highly flexible, making it applicable to a comprehensive set of problems. Furthermore, this property allows modeling complex reward schemes with several if-else branches.

Moreover, the preference between immediate and future rewards can be carefully controlled due to the explicit use of a discount factor. However, the most noticeable limitation is the agent's discrete action space closely related to Bellman's "curse of dimensionality" (Bellman, 1957). Some researchers like Brown (2000) find the approach to be "brittle" in the presence of noise or not to converge under certain conditions. In the future, an effort to enrich the state with other data sources and whether immediate or terminal rewards (one reward at the end) perform better and under what circumstances should be explored.

4.1.2. Actor-only approach

In this approach, the agent senses the state of the environment and acts directly, i.e., without computing and comparing the expected outcomes of different actions. Hence, the agent learns a direct mapping (a policy) from states to actions. The main advantages are continuous action space, faster convergence, and higher transparency. Having continuous actions, the agent can carefully interact with the environment, for example, to gradually increase an investment. Moreover, using multiple output neurons combined with a SoftMax activation function, a portfolio consisting of several assets can be managed simultaneously.

On the other hand, the most noticeable disadvantage of the actor-only approach is the need for a differentiable reward function, limiting the reward schemes that can be modeled. However, the impact of different network architectures (e.g., type and number of hidden layers) for deep RRL agents and the effect of varying reward functions for multi-security portfolios need to be further explored. Furthermore, the value of non-price-based information, e.g., sentiment data, could be analyzed.

4.1.3. Actor-critic approach

Actor-critic RL combines actor-only and critic-only RL (Yang and Xie, 2019). As its name suggests, actor-critic RL comprises two agents, the actor and the critic. The actor determines the actions and forms the policy of the system. The actor receives the current state as input at every step and computes the agent's action as output. The critic evaluates

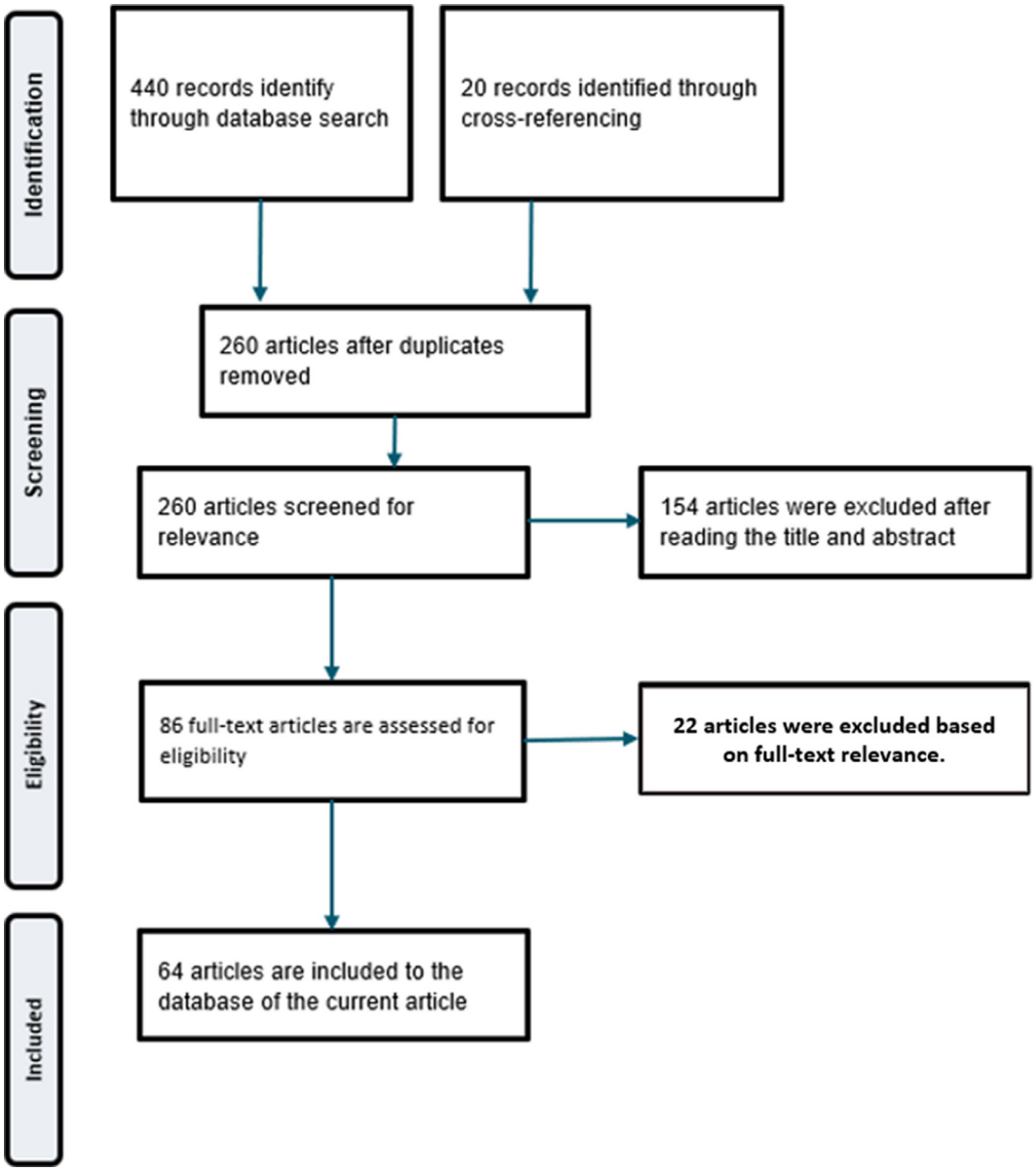

Fig. 4. Methodology for Systematic selection of articles for the review.

these actions. As a result, it gets the current state and the actor's action as input and computes the discounted future reward as output. The key idea is to gradually adjust the actor's policy parameters to maximize the reward predicted by the critic. Despite the ambition to combine the advantages of both agents, only a few studies are employing actor-critic RL in financial markets. These works include an RL agent to improve the forecast of stock returns obtained by an Elman network and combining actor-critic RL with fuzzy logic. Unfortunately, neither of these agents is comparable to the critic-only and actor-only agents discussed above. Future research could develop an actor-critic agent whose action resembles the trading decisions. Based on that, it should be analyzed whether the ambition of combining the advantages of actor-only and critic-only RL can be realized. However, due to a lack of large-scale comparative study, the actor-only approach has been widely used RL approach in finance (Levy, Platt, and Saenko 2019; Roder et al. 2020). First, the main reasons are its continuous actions. Secondly, it is usually a small number of parameters (which makes it less prone to overfitting). Third, its good convergence behavior (which results in faster training) and its recent improvements with deep learning techniques.

4.2. RL algorithms

Dynamic programming and Approximate dynamic programming algorithms are widely used to solve the problem of Prediction and control (Puterman, 2014) with the assumption that the agent has access to a model of the MDP environment.; however, in the real-world scenarios, this does hold. We often need to access the actual MDP environment directly, if not always. One can note that the real MDP environment doesn't give agents transition probabilities, i.e., it simply serves up a new state and reward when the agent act in a specific state. In other words, it gives the agent individual experiences of the next state and reward, rather than the actual probabilities of occurrence of the next states and rewards. So, what remains to be answered is whether we can infer the Optimal Value Function/Optimal Policy without access to a model. And this can be achieved with Reinforcement Learning algorithms. RL overcomes complexity, specifically, the Curse of Dimensionality and Curse of Modeling (Avella et al., 2014).

Most RL algorithms are founded on the Bellman Equations, and all RL Control algorithms are based on the fundamental idea of *Generalized*

*Policy Iteration (*Liu et al., 2015*).* But the exact ways in which the Bellman Equations and Generalized Policy Iteration idea are utilized in RL algorithms differ from one algorithm to another. They vary significantly from the Bellman Equations/ Generalized Policy Iteration idea used in DP algorithms.

4.2.1. Monte -Carlo

In the context of RL, Monte-Carlo refers to developing estimates of values without invoking Bellman equations for policy evaluation or optimization where both sides of the equations contain unknowns. Rather it uses direct estimates (Gobillon & Magnac,2016). As a result, Monte-Carlos's method accuracy is entirely independent of the size of the state space. Hence, it forms the basis for policy validation and hyperparameter tuning in state-of-the-art empirical reinforcement learning research.

However, Monte-Carlo policy evaluation requires the data collection protocol, and essentially the policy used to collect data and evaluation should be the same, implying the algorithm is on-policy (Mavrotas & Makryvelios,2021). When such assumptions fail, especially when the data collection policy is different from the evaluated policy, we face an off- policy evaluation problem. The extension of Monte-Carlo methods still benefits from the independence of the state space size; the dependence on the horizon is exponential (Li et al., 2020).

4.2.2. Temporal-Difference

The temporal difference (TD) algorithm estimates and updates the value of the current state using the values of its adjacent states and a reward function (Ng et al., 2018). Therefore, it is a model-free algorithm that effectively combines the strengths of the Monte Carlo Tree Search and Dynamic Programing reinforcement learning algorithms. In TD reinforcement learning, an agent is placed in an interactive environment where each action generates a new state. The environment responds by returning a reward value based on reward mechanisms. Like all other RL algorithms, the TD algorithm's goal is to maximize the cumulative reward. It achieves this by making continuous adjustments to the strategy; these adjustments are based on the returned reward values (Li et al., 2019). However, unlike other reinforcement learning approaches, the TD algorithm uses a sampling-learning process, i.e., it uses the value of the current action and its immediately adjacent states to estimate and update the value of the current state. Hence, the current model is updated immediately after a sample is obtained. This iterative process continues until the model converges.

4.2.3. Policy gradient algorithms

PG algorithms are practical in large action spaces, high-dimensional or continuous action spaces because in such spaces selecting an action by deriving an improved policy from an updating Q-Value function is intractable. Furthermore, a key advantage of PG is that it naturally explores because the policy function approximation is configured as a stochastic policy. Moreover, PG finds the best Stochastic Policy. This is not a factor for MDPs since we know an optimal Deterministic Policy for any MDP. Still, we often deal with Partially Observable MDPs in the real world, for which the set of optimal policies might all be stochastic policies. We have an advantage in the case of MDPs since PG algorithms naturally converge to the deterministic policy (the variance in the policy distribution will automatically converge to 0).

In contrast, we must reduce the ϵ of the ϵ-greedy policy by hand in Value Function-based algorithms. The appropriate declining trajectory of ϵ is typically hard to figure out by manual tuning. In situations where the policy function is simpler than the Value Function, we naturally benefit from pursuing Policy-based algorithms than Value Function-based algorithms. Perhaps the most significant advantage of PG algorithms is that prior knowledge of the functional form of the Optimal Policy enables us to structure the known functional form in the function approximation for the policy. Lastly, PG offers numerical benefits as small changes in θ yield small changes in π and consequently small changes in the distribution of occurrences of states. This results in stronger convergence guarantees for PG algorithms relative to Value Function-based algorithms.

The main disadvantage of PG Algorithms is that they typically converge to a local optimum because they are based on gradient ascent. In contrast, Value Function-based algorithms converge to a global optimum. Furthermore, the Policy Evaluation of PG is typically inefficient and can have high variance. Lastly, the Policy Improvements of PG happen in small steps, so PG algorithms are slow to converge.

5. Deep reinforcement learning

5.1. Basics of neural network

A typical neural network is a nonlinear function represented by a collection of neurons. These are typically arranged as several layers connected by operators such as filters, poolings and gates, that map an input variable in $\mathbb{R}^n$ to an output variable in $\mathbb{R}^m$ for some $n,m\epsilon\mathbb{Z}^+$ (Lim et al., 2021)

The basic architecture of the perceptron is shown in FigFigure 6. Consider a situation where each training instance is of the form (X,y), where each $X = [x_1 ,\ldots]$ contains d feature variables, and y $\in \{-1, +1\}$ contains the observed value of the binary class variable. By "observed value" we refer to the fact that it is given to us as a part of the training data, and our goal is to predict the class variable for cases in which it is not observed (Li et al., 2017).

The input layer contains d nodes that transmit the d features $X = [x_1 \ldots.x_d]$ with edges of weight $W = [w_1 \ \ldots]$ to an output node. The input layer does not perform any computation. The linear function $W \cdot X = d\ i = 1\ w_i\ x_i$ is computed at the output node. Subsequently, the sign of this real value is used to predict the dependent variable of X. Therefore, the prediction ˆy is computed as follows:

$$\hat{y} = \text{sign}\{W \cdot X\} = \text{sign}\{d\ j = 1\ w_{\ j} x_j\}(1.1) \tag{13}$$

The three most popular neural network architectures are – Fully connected neural networks(FNN), Convolutional neural networks (CNN) and Recurrent neural network (RNN).

A fully connected Neural Network (FNN) is the simplest neural network architecture where any given neuron is connected to all neurons in the previous layer (Chen et al., 2021). To describe the setup, we fix the number of layers L $\epsilon\mathbb{Z}^+$ in the neural network and the width of the i th layer $n_i\ \epsilon\ \mathbb{Z}^+$ for i = 1,2….,I. Then for an input variable $z\ \epsilon\ \mathbb{R}^n$, the functional form of the FNN is

$$F(z; W, b) = W_I \cdot \sigma\left(W_{i-1} \cdots \sigma\left(W_1 z + b_1\right) \cdots + b_{I-1}\right) + b_I \tag{14}$$

Where (W,b) represents all the parameters in the neural network, with $W = (W_1 , W_2, W_3 \ldots.\ W_I)$ and $b = (b_1 , b_2, b_3 \ldots..\ b_I)$. In the neural network's literature, the W_I's are often called the weight matrices, the b_I's are called biasvectors, and $\sigma$is referred to as the *activation function.* Several popular choices for the activation function include ReLU with $\sigma(u) = \max(u; 0)$, Leaky ReLU with $\sigma(u) = a_1 \max(u; 0)$- a2 max(-u, 0) where a1; a2 > 0, and smooth functions such as $\sigma(.) = \tanh(.)$.

Convolutional neural networks (CNNs) are another type of feedforward neural network that are especially popular for image processing. In the finance setting CNNs have been successfully applied to price prediction based on inputs which are images containing visualizations of price dynamics and trading volumes (Jiang et al., 2020). The CNNs have two main building blocks – convolutional layers and pooling layers. Convolutional layers are used to capture local patterns in the images and pooling layers are used to reduce the dimension of the problem and improve the computational efficiency. A simple CNN can be represented as:

$$F(z; \boldsymbol{H}, \boldsymbol{W}, \boldsymbol{b}) = \ \boldsymbol{W} \cdot \sigma(z\ *\ \boldsymbol{H}) + \boldsymbol{b} \tag{15}$$

Recurrent Neural Networks (RNNs) are a family of neural networks that are widely used in processing sequential data, including speech,

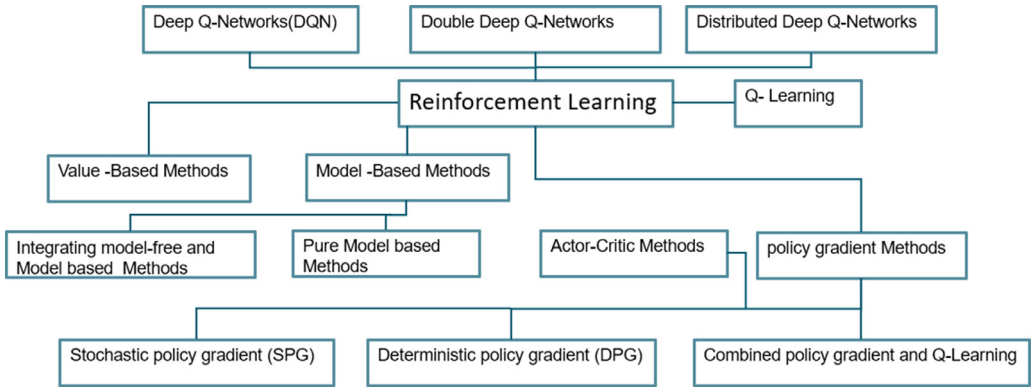

Fig. 5. Select taxonomy of dominant algorithms in Modern RL widely used in Finance applications (combining Lillicrap et al., 2015; Mnih et al., 2016; Schulman et al., 2017;Silver et al., 2017; Fujimoto et al., 2018; Ha & Schmidhuber, 2018; Haarnoja et al., 2018).

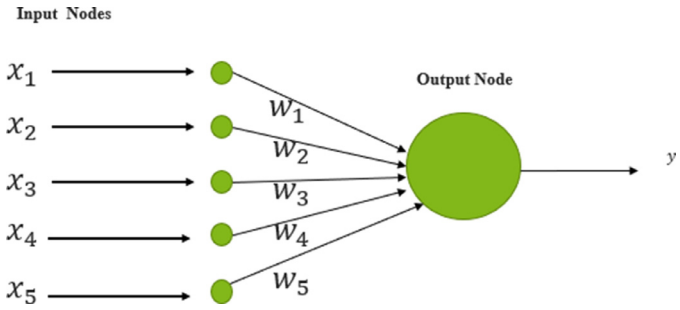

Fig. 6. Basic architecture of perceptron.

text, and financial time series data. Unlike feed-forward neural networks, RNNs are a class of artificial neural networks where connections between units form a directed cycle. RNNs can use their internal memory to process arbitrary sequences of inputs and hence are applicable to tasks such as sequential data processing.

For RNNs, our input is a sequence of data $Z_1, Z_2 \ldots\ldots Z_T$. An RNN models the internal stateh_t bya recursive relation

$$h_t = F\left(h_{t-1}, Z_t; \theta\right) \tag{16}$$

where F is a neural network with parameter θ(for example θ = (W, b)). Then the output is given by

$$\hat{s}_t = G\left(h_{t-1}, Z_t; \theta\right) \tag{17}$$

where G is another neural network with parameter θ. There are two important variants of the vanilla RNN introduced above – the long short-term memory (LSTM) and the gated recurrent units (GRUs) (Zhang et al., 2022) Compared to vanilla RNNs, LSTM and GRUs are shown to have better performance in handling sequential data with long-term dependence due to their flexibility in propagating information flows (Heryadi et al., 2017).

5.2. Deep value-based methods

In Reinforcement learning the distribution of reward function r and the transition function P are unknown to the agent However the agents can observe samples (s,a,r,s') by interacting with it, without having any prior knowledge of r or P. The agent can learn the value function using the samples(Sutton and Barto,2014). This is the basic concept used in classical temporal-difference learning,Q-learning and SARSA (State-Action-Reward-State-Action).

5.2.1. Temporal-Difference in prediction learning

We already have introduced TD in the previous section. Here we discuss the learning aspect of TD. TD learning is an unsupervised technique in which the learning agent learns to predict the expected value of a variable occurring at the end of a sequence of states (Heryadi et al., 2017). Reinforcement learning (RL) extends this technique by allowing the learned state-values to guide actions that change the environment state. Thus, RL is concerned with the more holistic problem of an agent learning effective interaction with its environment. TD algorithms are often used in reinforcement learning to predict the total amount of reward expected over the future. Still, they can also be used to predict other quantities. Continuous-time TD algorithms have also been developed(Sutton and Barto,2014). Given some samples (s; a; r; s') obtained by following a policy $\boldsymbol{\pi}$, the agent can update her estimate of the value function V^{π}at the $(n+1)$-th iteration by:

$$V^{\Pi,(n+1)}(S) \leftarrow \left(1-\beta_n(s,a)\right)V^{\Pi,(n)}(s) + \beta_n(s,a)\left[r + \gamma V^{\Pi,(n)}(s\prime)\right] \tag{18}$$

5.2.2. Least-Squares temporal difference learning

The Least-Squares TD(λ) algorithm, or LSTD(λ), converges to the same coefficients $\beta\lambda$ that TD(λ) does. However, instead of performing gradient descent, LSTD(λ) builds explicit estimates. Estimates of the C matrix and d vector (actually, estimates of a constant multiple of C and d) solve d + C$\beta\lambda$ = 0 directly (Bradtke & Barote,1996; Boyan,2002; Xu et al., 2002). LSTD explicitly solves the value function parameters that result in zero-mean TD update overall observed state transitions. The resulting algorithm is considerably more data-efficient than TD(0) but less computationally efficient. LSTD requires O(n2) computation per time step, even incremental inverse computations. Hence, the improved data efficiency is obtained at a substantial computational price. LSTD can be seen as immediately solving for value function parameters for which the sum TD update overall the observed data is zero.

5.3. Deep Policy- based Methods

Deep policy-based methods are extensions of policy-based methods using neural network approximations. Using neural networks to parametrize the policy and/or value functions in the vanilla version of policy-based methods leads to neural Actor-Critic algorithms (Zhang et al., 2020) and deep DPG (DDPG). In addition, since introducing an entropy term in the objective function encourages policy exploration and speeds the learning process; there have been some recent developments in (off-policy) soft Actor-Critic algorithms (Dai et al., 2022) using neural networks, which solve the RL problem with entropy regularization.

Below we introduce the DDPG algorithm, one of the most popular deep policy-based methods, which has been applied in many financial problems' is a model-free off-policy Actor-Critic algorithm, first introduced in, which combines the DQN, and DPG algorithms extend the DQN to continuous action spaces by incorporating DPG to learn a deterministic strategy. To encourage exploration, DDPG uses the following

Table 1: overview of studies that uses Reinforcement Learning in financial prediction problems

Category	Subcategory	Decision making objective
Credit Risk	Consumer Credit risk	- general consumer default - credit card delinquency and default - Bill payment in developing countries - credit card repayment patterns.
	Corporate credit risk	- Firms' credit rating changes - Corporate bankruptcy - Fintech loan default - Recovery rates of corporate bonds
	Real estate credit risk	- Mortgage loan risk - Commercial real estate default
Financial policy and Outcomes	Financial outcomes	- Capital structure - Earnings
	Fraud (accounting)	- Accounting fraud from financial statements variables - Accounting fraud from annual report topics
	Startup investment outcome	- Startup acquisitions - Startup valuations and success probabilities
Trading and assets	Equities	- Stock returns - Stock volatility - Stock covariance - Equity risk premium
	Bonds	- Future excess returns
	Foreign Exchange	- Direction of changes in exchange rates
	Derivatives	- Prices of options on index futures - Prices of general derivatives
	Financial claims	- Stochastic discount factor
	Market	- Lifespan of trading orders
	Microstructure	- General microstructure variables
	Investors	- Mutual fund performance Retail investors' portfolio allocations and performance
	Algorithmic Trading	- buy and sell decision
	Financial Text Mining	- sentiment analysis - real time streaming news/tweets - instant text- based information

action:

$$a_t \sim \pi^D\left(s_t;\theta_t\right)+\epsilon \tag{19}$$

Where π^D is a deterministic policy and a random variable sampled from some distribution N can be chosen according to the environment. One must make a note that the algorithm requires a small learning rate.

6. Financial applications

In this section we discuss an overview of various interesting utilization of DL, RL and DRL approaches in finance. We have categorized our review of financial applications in the below category (as shown in table 1):

We have taken our study to category level bringing some research and finding from each category level as below:

6.1. Information management for credit risk

Risk Assessment measures the risk attached to any given asset, be it firm, person, product, or the bank. It has been one of RL researchers' most studied and researched areas. The problem has been studied under different categories, for example, Credit scoring and evaluation, Credit rating, bankruptcy prediction (Hosaka, 2019), loan/insurance underwriting, bond rating (Chi et al., 2021), loan application, consumer credit determination, corporate credit rating, mortgage choice decision, financial distress prediction, and business failure prediction (Chen et al., 2020). Asset pricing is highly dependent on risk assessment measures; therefore, identifying the risk status is critical. The majority focus of the risk assessment studies is on credit scoring and bank distress classification. However, a few papers also cover Crisis forecasting and detection of risky transactions (Serrano, 2018).

For credit score classification, Luo et al., and Wu (2017), in first of its kind implementation, used a deep belief network learning approach for Corporate Credit rating and corresponding credit classification (A, B, or C). Out of all the tested models, Deep Belief Network with Restricted Boltzmann Machine performed the best. Similarly, Yu et al., and Chen (2018), a cascaded hybrid model of DBN, Backpropagation, and SVM for credit classification, was implemented, and good performance results (the accuracy was above 80–90%) were achieved. Finally, Li et al. (2017) performed credit risk classification using an ensemble of deep MLP networks, each using subspaces of the whole space by k-means (using minority class in each, but only a partial subspace of the majority class).

Multiple subspaces were used for each classifier to handle the data imbalance problem, where each had all the positive (minor) instances. Still, a subsample of negative (majority) instances finally used an ensemble of deep MLPs combining each subspace model. Credit scoring was performed using an SAE network and GP model to create credit assessment rules to generate good or bad credit cases. In another study, Neagoe et al.(2018) classified credit scores using various DMLP and deep CNN networks. Zhu et al., 2018approached the consumer credit scoring classification with hybrid deep learning models. They transformed the 2-D consumer input data into a 2-D pixel matrix and used the resulting image to train and test data for DRL using CNN.

Multiple subspaces were used for each classifier to handle the data imbalance problem, where each had all the positive (minor) instances. Still, a subsample of negative (majority) instances finally used an ensemble of deep MLPs combining each subspace model. Credit scoring was performed using an SAE network and GP model to create credit assessment rules to generate good or bad credit cases. In another study, Neagoe et al. (2018) classified credit scores using various DMLP and

Table 2: some of the major articles on Credit risk (2014 - 2020)

Authors	Study Period	Method	Feature set
Li et al.	2017	DRL, DNN	Personal finance variables
Tran et al.	2016	RL,DCNN	Personal Finance variables
Rawte et al.	2006-2017	RL,LSTM	Weighted sentiment polarity and ROA
Malik et al.	1976-2017	DRL,LSTM,RL	Macro-economic variables and Bank transactions
Hosaka	2014-2019	CNN, RL	Financial ratios
Ozbayoglu et al.	2016-2019	RL-LSTM, CNN	Order details and transaction details
Chatzis	1996-2017	DNN, Logit, RL,RL-LSTM	Index data, exchange rates

Table 3: Some of the major articles on Fraud Management works (2014 - 2020)

Authors	Study Period	Method	Feature set
Gomes	2009 -2017	RL,Autoencoders	Party name, type of Expends, state expenses
Jurgovsky et al	2015	LSTM	Transaction and bank details
Heryadi and Warnars	2016-2017	CNN,RL-LSTM	Financial transactions over period
Han et al	2018	RL-LSTM	Pricing,Interest rates

deep CNN networks. Zhu et al. (2018)approached the consumer credit scoring classification with hybrid deep learning models. They transformed the 2-D consumer input data into a 2-D pixel matrix and used the resulting image to train and test data for DRL using CNN.

Financial distress prediction for banks and corporates is studied extensively. Lanbouri and Achchaba (2015) used a Deep belief network and SVM to predict the financial distress, thereby identifying if it's in trouble or not. At the same time, Rawte et al., and Zaki (2018) used RL-LSTM for bank risk classification. Ronnqvist and Sarlin (2015) explored the word sequence learning to extract new semantics, and associated events were labeled with the bank stress. Determined bank stress was then classified against the formed semantic vector representation threshold. Prediction and semantic meaning extraction were integrated neatly.

Ozbayoglu et al., and Sezer (2020) were able to classify with high accuracy using CNN and LSTM networks if a stock market transaction was risky or not. Researchers have been also developed several RL and DRL models for detecting events that caused the stock market to crash (Zhang et al., 2018)

6.2. Financial policies and outcomes

6.2.1. Information management for fraud detection

Several types of financial fraud cases exist, including credit card fraud (Jurgovsky et al., 2018), money laundering, consumer credit fraud, tax evasion, bank fraud, and insurance claim fraud. This is one of the most extensively studied areas of finance for AI and ML researched as anomaly detection and is generally a classification problem. Several survey papers were published accordingly as the topic has attracted much attention from researchers. At different times, Kirkos et al.(2017), Ngai et al., and Sun (2011), and West et al. (2016) all reviewed the accounting and financial fraud detection studies based on soft computing and data mining techniques. as shown in table 2

Heryadi and co-researchers (2017) have focused on their research identifying credit card fraud and exploring the effect of data imbalance on fraud and non-fraud data. Roy et al. (2018) used the LSTM model for credit card fraud detection, whereas Gomez et al.(2018) used MLP networks to classify if a credit card transaction was fraudulent or not. Furthermore, many researchers used an ensemble of Feed Forward Neural Network for card fraud detection. In a similar study, Gomes et al., and Carvalho (2017) worked on parliamentary expenditure spending proposing an anomaly detection model to identify anomalies in spending in Brazilian elections using deep AE. Wang et al. (2018) used text mining and DNN models to detect automobile insurance fraud. Other similar studies have applied LSTM and used character sequences in financial transactions and the responses from the other side to detect if the transaction was fraud or not. Finally, Goumagias et al. (2018) used deep Q-learning (RL) to predict risk-averse firms. Table 3 summarizes some of the major articles on Fraud management.

6.3. Trading and assets

6.3.1. Big data driven portfolio optimization

In portfolio optimization problems, a trader needs to select and trade the best portfolio of assets to maximize some objective function, which typically includes the expected return and some measure of the risk. By investing in such portfolios (Lee and Soo, 2017), the trader diversifies his investments, which achieves a higher return per unit of risk than only investing in a single asset (Lee and Yoo, 2020). Both value-based methods such as Q-learning (Han et al., 2018; Tsantekidis et al., 2017), SARSA (Han et al., 2018), and DQN (Kumar and Vadlamani, 2016), and policy-based algorithms such as DPG and DDPG (Iwasaki & Chen,2018) have been applied to solve portfolio optimization problems. The state variables are often time, asset prices, past asset returns, current assets holdings, and remaining balance. The control variables are typically the amount/proportion of wealth invested in each portfolio component. Examples of reward signals include portfolio return (Han et al., 2018), Sharpe ratio (Han et al., 2018; Tsantekidis et al., 2017), and profit (Tsantekidis et al., 2017). constantly rebalanced portfolio (CRP) is used as a benchmark strategy, where the portfolio is rebalanced to the initial wealth distribution among the assets at each period. The buy-and-hold or do-nothing strategy does not take any action but instead holds the initial portfolio until the end. The performance measures studied in these papers include the Sharpe ratio (Iwasaki & Chen,2018), the Sortino ratio (Lim et al., 2021), portfolio returns (Wang,2019; Xiong,2018), portfolio values (Jiang & Liang,2017; Xiong et al., 2018), and cumulative profits (Du et al., 2016). Some models incorporate the transaction costs (Jiang & Liang,2017; Liang et al., 2018) and investments in the risk-free asset [Du et al., 2016; Wang,2019, 2016; Jiang & Liang,2017]. Du et al. (2016) considered the portfolio optimization problems of a risky asset and a risk-free asset for value-based algorithms. They compared the performance of the Q-learning algorithm and a Recurrent RL (RRL) algorithm under three different value functions, including the Sharpe ratio, differential Sharpe ratio, and profit. The RRL algorithm is a policy-based method that uses the last action as an input. They concluded that the Q-learning algorithm is more sensitive to the choice of the value function and has less stable performance than the RRL algorithm. They also suggested that the (differential) Sharpe ratio is preferred rather than the profit as the reward function. For policy-based algorithms, Jiang and Liang (2017) proposed a framework combining neural networks with DPG. They used a so-called ensemble of identical independent evaluators (EIIE) topology to predict the potential growth of the assets in the immediate future using historical data, which includes the highest, lowest, and closing prices of portfolio components.

Table 4: Some of the major articles on Portfolio management (2014- 2020)

Authors	Study Period	Method	Feature set
Zhou Bo	1993-2017	LSTM,MLP,RL	Firm's characteristics, Price
Lee and Yoo	1997-2016	RNN,LSTM,GRU	Price data, OCHLV
Heaton	2014-2017	CNN, RNN,LSTM	Price data
Jiang and Liang	2015-2016	CNN,RL	Price data
Almahdi and yang	2017	RL	Transaction costs
Liang et al.	2015-2018	DRL, PPO,RL	Fundamental data, OCHLV
Iwasaki and Chen	2016-2018	LSTM,CNN	Text

The actual crypto currency market data experiments showed that their framework achieves a higher Sharpe ratio and cumulative portfolio values than three benchmarks, including CRP and several published RL models. Singh et al., 2022 explored the DDPG algorithm for the portfolio selection of 30 stocks, where at each time point, the agent can choose to buy, sell, or hold each stock. The DDPG algorithm was shown to outperform two classical strategies, including the min-variance portfolio allocation method, in terms of several performance measures, including final portfolio values, annualized return, and Sharpe ratio using historical daily prices of the 30 stocks. One can note that the classical stochastic control approach for portfolio optimization problems across multiple assets requires both a realistic representation of the temporal dynamics of individual assets and an adequate representation of their co-movements. It is challenging when the assets belong to different classes (stocks, options, futures, interest rates, and derivatives). On the other hand, the model-free RL approach does not rely on the specification of the joint dynamics across assets . Table 4 summarizes these studies with their intended purposes.

6.3.2. Information management for asset and derivatives market

Accurate pricing and valuation of an asset is a fundamental research and study area in finance. Many AI/ML models have been developed for banks, corporates, real estate, derivative products, etc. However, RL and DRL have not been widely applied to this field yet. There are some possible implementation areas where RL models can assist the asset pricing researchers or valuation experts(Hsu et al., 2019). Unfortunately, there were only a handful of studies that we could find during our research within the RL and finance community. In their sentiment analysis to predict stock price prediction, Iwasaki et al. (2018) used a DFNN model and the analyst reports. Based on the forecast, different portfolio selection approaches were implemented. Culkin et al.(2017) proposed a novel method that used the feedforward DNN model to predict option prices by comparing their results with the Black & Scholes option pricing formula. Similarly, Hsu et al., 2018proposed a novel method using bid-ask spreads and Black & Scholes option price model parameters with 3-layer DMLP that predicted TAIEX option prices. Finally, Feng et al.(2019)explored characteristic features like Asset growth, Industry momentum, Market equity, Market Beta to predict US equity returns. Meanwhile, derivative products based financial models are widely popular and common.RL models development could benefit Options pricing, hedging strategy development, financial engineering with options, futures, forward contracts. Some recent studies show that the topic has gotten the attention of researchers, and they have started showing interest in RL models that can provide solutions to this complex and challenging field. Table 5 summarizes these studies with their intended purposes.

6.3.3. Algorithmic trading

It is defined as a buy-sell decision made solely by algorithmic models based on simple rules, mathematical models, or complex functions. With the advent of online trading platforms and guidelines, Algorithmic trading has been a hugely successful application in the last couple of years. Most of the Algo-trading applications are coupled with price prediction models for market timing purposes (Du et al., 2016). Therefore, most price or trend forecasting models that trigger buy-sell signals based on their prediction are also considered Algo-trading systems (Mahmoudi et al., 2018). However, some studies propose stand-alone Algo-trading models focused on the dynamics of the transaction itself. Optimizing trading parameters such as bid-ask spread, analysis of limit order book, position-sizing was the most preferred DL model in predicting stock or index prices implementations (Lei et al., 2020). Arpaci et al. (2017) studied market microstructures-based trade indicators used to input RNN with Graves LSTM to perform the price prediction for algorithmic stock trading. Bao et al., 2017 used technical indicators as the input into Wavelet Transforms (WT), Stacked Autoencoders (SAEs), and LSTM for the forecasting of stock prices. Zhang et al. (2022) implemented. CNN and the LSTM model were implemented together where CNN was used for stock selection, LSTM was used for price prediction. Deng et al. (2017) used Fuzzy Deep Direct Reinforcement Learning (FDDR) for stock price prediction and trading signal generation. For index prediction, the following studies are noteworthy. Saleh(2020) used LSTM to make the price prediction of the S&P500 index. Si et al.(2017) implemented Chinese intraday futures (Boukas et al., 2021) market trading model with DRL and LSTM. Brzeszczynski et al.(2019) used the feedforward DNN method and Open, Close, High, Low (OCHL) of the time series index data to predict Singapore Stock Market index data. In addition, forex or cryptocurrency trading was implemented in some studies. Finally, Jeong et al.(2019) combined deep Q-learning and DNN to implement price forecasting. They intended to solve three different problems: increasing profit in a market (Bari and Agah, 2020); Prediction of the number of shares to trade and Preventing overfitting with insufficient financial data. A summary of major work on Algo-Trading Text Mining is presented in below table (Table 6).

6.3.4. Financial big data analytics using text mining

The evolution of the financial models has been hugely influenced by the broad reach of social media, instant text-based information, and real-time streaming. It results again in the popularity of financial text mining studies. Though many of these studies are directly linked to sentiment analysis through crowdsourcing, there are many implementations in financial statements, content retrieval of news, disclosures, etc. There are a few surveys focused on text mining and financial news analytics (Kushwaha et al., 2021). For example, Mitra et al. (2012) edited a book on news analytics in finance. In contrast, Kumar et al., 2021,Loughran et al.(2016) Kumar et al.(2016) have published surveys on textual analysis of financial documents, corporate disclosures, and news.

Ying et al., and Yang (2017) used individual social security payment types (paid, unpaid, repaid, transferred) to classify and predict using LSTM, HMM, and SVM. Sohangir et al.(2018), in their study, classify the authors of Stock Twits as export or non-expert using neural network models. Costa and Silva (2016) applied RL-LSTM and used character sequences in financial transactions and the responses from the other side to detect if the transaction was fraud or not. Wang et al.(2018)used text mining and DNN models to detect automobile insurance fraud. At DeepAI, the word sequence learning extracted the news semantics, and bank stress was determined and classified with the associated events. Day et al. (2016) used financial sentiment analysis using text mining and DNN for algorithmic stock trading. Cerchiello et al., 2018 used the fundamental data and text mining from the financial news (Reuters) to classify the bank distress. Rönnqvist and Sarlin (2017) identified the

Table 5: Some of the major articles on asset pricing and derivatives (2014 - 2020)

Authors	Study Period	Method	Feature set
Feng et al.	2019	RDL,DL	Firm characteristics
Iwasaki and chen	2016-2018	RL-LSTM, Bi-LSTM	Text
Hsu et al.	2018	Black Scholes, RL	Fundamental analysis, option price
Deng et al.	2017	RL	Stock market data, Firm data
Martinez-Miranda et al.	2016	RL	Stock market data
Feuerriegel and Prendiger	2016	RL	Transaction fees, Stock Prices.
Zhang and Maringer	2015	RL	Stock Prices and Trade data
Chen et al.	2018	RL	Trading data, Market data.
Chakraborty	2019	RL	Trading market data
Mosavi et al	2020	DRL	-

Table 6: Some of the major articles on Algo-Trading Text Mining (2014-2020)

Authors	Study Period	Method	Feature set
Dixon et al.	2014	DNN	Price Data
Deng et al.	2014-2017	RL,DNN,FDDR	Profit, return,Profit-loss
Bari and Agah	2016-2018	RL,DRL	Text and Price data
Jeong and Kim	2017	DNN,DRL	Total profit, correlation
Zhang et al.	2017	CNN,LSTM	Annual return, Maximum Retracement
Sirignano and Rama	2017-2019	RL,DRL	Transactional data, Price data
Serrano,will	2016-2019	RL,DNN	Asset Price data
Boukas et al	-	DRL	Market data
Lei et al	-	DRL	Price data, Asset price

Table 7: Some of the major articles on Financial Text Mining (2014 - 2020)

Authors	Study Period	Method	Feature set
Verma et al.	2013-2017	LSTM, RL	Index data, news
Ying et al.	2008-2014	RNN	Insurance details like ID, area-code.
Zhang et al	2015	RL,CNN	Price data, News and social media
Mahmoudi et al	2008-2016	CNN,LSTM,DL	Sentences, Stock-Twits messages
Sohangir and Wang	2015-2016	CNN, Doc2vec	Sentences, Stock-Twits messages.
Lee and Soo	2017	LSTM, RL,DL	Technical indicators, Price data, News
Chi et al.	2014-2020	CNN,LSTM,RL	Input news, OCHLV, Technical indicators.

bank distress by extracting the data from the financial news through text mining—the proposed method used DFNN on semantic sentence vectors to classify if there was an event or event not.

For financial sentiment analysis, Almahdi and Yang (2019)compared CNN, LSTM, and GRU-based DL models against MLP. Rawte et al. (2018) tried to solve three problems using CNN, LSTM, SVM, RF: Bank risk classification (Dixon et al., 2017), sentiment analysis, and Return on Assets (ROA) regression. Finally, Chang et al., and Ke (2016) implemented the estimation of information content polarity (negative/positive effect) with text mining, word vector, lexical, contextual input, and various LSTM models. They used the financial news from Reuters. Finally, it is worth mentioning that there are also some studies (Khadjeh et al., 2014; Kumar and Vadlamani,2016) of text mining for financial prediction models. In the below table (table 7) we highlight some of the major articles on Financial Text mining .

7. Discussion

Our review establishes that both DL and DRL algorithms have almost the same prediction quality in terms of statistical error when used for prediction purposes. In addition, reinforcement learning can simulate more efficient models with more realistic market constraints. While deep RL goes further, that solves the scalability problem of RL algorithms, which is crucial with fast users and financial growth. Moreover, it efficiently works with high-dimensional settings as highly desired in the financial market (Chakraborty, 2019). Where DRL can give notable helps to design more efficient forecasting algorithms and analyze the market with real-world parameters (as shown in table 8). For example, the deep deterministic policy gradient (DDPG) method (Xiong et al., 2018) in stock trading demonstrated how the model could handle large settings concerning stability, improve data use, and equilibrate risk while optimizing the return with a high-performance guarantee (Chatzis et al., 2018). Another example in the deep RL framework used the DQN scheme to amend the news recommendation accuracy dealing with huge users simultaneously with a considerably high-performance method guarantee (Mosavi et al., 2020). Our review demonstrated outstanding potential for improving deep learning methods applied to the RL problems to better analyze the relevant problem for finding the best strategy in a wide range of finance domains.

7.1. Practical implication

While using deep reinforcement learning architecture, the combination of both DL and RL approaches, for RL to resolve the scalability problem and be applied to the high-dimensional issues desired in a real-world market setting. We reviewed approaches such as CNN, RNN, LSTM, and a few others from both DL and Deep RL in various applications of Finance. The advanced models engaged in providing improved prediction, extracting better information, and finding optimal strategy mostly in complicated and dynamic market conditions. Our work represents that the fundamental issue of all proposed approaches is mainly dealing with the model complexity, robustness, accuracy, performance, computational tasks, risk constraints, and profitability. Practitioners can employ a variety of DL and deep RL techniques, with the relevant strengths and weaknesses, that serve the Finance problems to enable the machine to detect the optimal strategy associated with the market. Recent works showed that the novel techniques in DNN and recently interacting with reinforcement learning, so-called deep RL, can considerably enhance the model performance and accuracy while handling real-world finance problems. Finally, our review indicates that the recent DL and deep RL approaches perform better than the classical ML approaches. Therefore, more efficient novel algorithms can be designed

Table 8: Comparative study of DL,RL and DRL models for decision making in Finance

Methods	Application	Dataset type	Complexity	Accuracy /Processing rate
AE	Risk management	Historical	High	High /Reasonable
AE & RBM	Credit Card Fraud	Historical	Reasonably high	High/Reasonable
CNNR	Profit Market	Transactional	Reasonable	Reasonably high/High
DNN	Stock pricing	Historical	Reasonable	High/Reasonable
DDPG	Stock trading	Historical	Reasonable	Reasonably high/High
LSTM-SVR	Investment	Historical	Reasonable	Reasonably high/High
IMF&MB	Portfolio management	Historical	High	Reasonably high/High

by combining Deep Neural network architecture in reinforcement learning concepts to detect the optimal strategy, namely, optimize the profit and minimize the loss while concerning the risk parameters in a highly competitive market.

7.2. Future research agenda

RL is highly growing research in DL. However, most of the well-known results in RL have been in single-agent environments, while the cumulative reward involves the single state-action spaces. It would be interesting that consider multi-agent scenarios in RL that comprise more than one agent where the cumulative reward can be affected by the actions of other agents. However, some recent research applied multi-agent reinforcement learning (MARL) scenarios to a small set of agents but not to a large set of agents (Jaderberg et al., 2019. For instance, in the financial markets where there are a huge number of agents simultaneously, the acts of agents concerning the optimization problem might be affected by the decisions of the other agents. As a result, the MARL can provide an imprecise model and inaccurate prediction while considering infinite agents. Though the performance of the deep reinforcement learning algorithm for information-based decision-making in finance has been good in the controlled conditions. However, its application to the actual trading environment, such as investment banks, commercial banks, and exchanges, has been limited (Arjun et al., 2021). In the future, with the support of the big data platform and the AI platform, the deep reinforcement learning model will finally realize the landing and application of intelligent scenario products, such as investment research, internal risk control, and wealth management in the financial field.

8. Conclusion

This paper reviews how and in which area of Finance we can use RL and DRL models to make better-informed decisions. Secondly, we establish how the existing DL, RL, and DRL approaches can be classified for better understanding and application based on their application. And finally, it addresses the open issues and challenges in current deep reinforcement learning models for information-based decision-making in Finance. In our review, we reviewed how in Comparison with the traditional statistical analysis method of Finance, applying deep reinforcement learning in Finance can efficiently obtain valuable information from many nonlinear and complex financial data. Thus, RL and DRL can effectively forecast and detect difficult market trends compared to the traditional ML algorithms, with the significant advantage of high-level feature extraction properties and proficiency of the problem solver methods. Furthermore, reinforcement learning enables us to construct a more efficient framework considering crucial market constraints by integrating the prediction problem with the portfolio structure task.

With the support of the big data platform and the AI platform, the deep reinforcement learning model will finally realize the landing and application of intelligent scenario products.

Funding

"Not Applicable"

Ethics approval

"Not Applicable"

Consent to participate

"Not Applicable"

Consent for publication

"Not Applicable"

Availability of data and material

"Not Applicable"

Code availability

"Not Applicable"

Declarations

We hereby declare that the work described has not been published before in any form and it is not under consideration for publication elsewhere, its submission to the Journal of Information management Data Insights has been approved by all authors as well as the responsible authorities – tacitly or explicitly – at the institute where the work has been carried out. If accepted, it will not be published elsewhere in the same form, in English or any other language, including electronically, without the written consent of the copyright holder and Journal of Information management Data Insights will not be held legally responsible should there be any claims for compensation or dispute on authorship.

Declaration of Competing Interest

The authors report no conflict of interest in the study undertaken. All contributors have been listed as authors and all authors have contributed.

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Almahdi, S., & Yang, S. Y. (2019). A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning. *Expert Systems with Applications, 130*, 145–156.

Arjun, R., Kuanr, A., & Suprabha, K. R. (2021). Developing banking intelligence in emerging markets: Systematic review and agenda. *International Journal of Information Management Data Insights, 1*(2), Article 100026.

Bao, W., Yue, J., & Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one, 12*(7), Article e0180944.

Bari, O. A., & Agah, A. (2020). Ensembles of text and time-series models for automatic generation of financial trading signals from social media content. *Journal of Intelligent Systems, 29*(1), 753–772.

Bergemann, D., & Välimäki, J. (1996). Learning and strategic pricing. *Econometrica, 64*(5), 1125–1149 1125-1149.

Boukas, I., Ernst, D., Théate, T., Bolland, A., Huynen, A., Buchwald, M., et al., (2021). A deep reinforcement learning framework for continuous intraday market bidding. *Machine Learning, 110*(9), 2335–2387.

Bradtke, S. J., & Barto, A. G. (1996). Linear least-squares algorithms for temporal difference learning. *Machine learning, 22*(1), 33–57.

Brzeszczyński, J., & Ibrahim, B. M. (2019). A stock market trading system based on foreign and domestic information. *Expert Systems with Applications, 118*, 381–399.

Cerchiello, P., Nicola, G., Rönnqvist, S., & Sarlin, P. (2018). Deep learning for assessing banks' distress from news and numerical financial data. *Michael J. Brennan Irish Finance Working Paper Series Research Paper*, 18–115.

Chakraborty, S. (2019). Capturing financial markets to apply deep reinforcement learning. *arXiv preprint* arXiv:1907.04373.

Chang, C. Y., Zhang, Y., Teng, Z., Bozanic, Z., & Ke, B. (2016). Measuring the information content of financial news. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (pp. 3216–3225).

Chatzis, S. P., Siakoulis, V., Petropoulos, A., Stavroulakis, E., & Vlachogiannakis, N. (2018). Forecasting stock market crisis events using deep and statistical machine learning techniques. *Expert systems with applications, 112*, 353–371.

Chen, S. S., Choubey, B., & Singh, V. (2021). A neural network-based price sensitive recommender model to predict customer choices based on price effect. *Journal of Retailing and Consumer Services, 61*, Article 102573.

Chen, Shiuann-Shuoh, & Singh, V. (2020). Developing a Cloud EBC System with 2P-Cloud Architecture. *Journal of Applied Science and Engineering, 23*(2), 185–195.

Chi, M., Hongyan, S., Shaofan, W., Shengliang, L., & Jingyan, L. (2021). Bond default prediction based on deep learning and knowledge graph technology. *IEEE access : Practical innovations, open solutions, 9*, 12750–12761.

Dai, P., Yu, W., Wang, H., & Baldi, S. (2022). Distributed Actor-Critic Algorithms for Multiagent Reinforcement Learning Over Directed Graphs. *IEEE Transactions on Neural Networks and Learning Systems*. 10.1109/TNNLS.2021.3139138.

Day, M. Y., & Lee, C. C. (2016). Deep learning for financial sentiment analysis on finance news providers. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (pp. 1127–1134). IEEE.

Dixon, M., Klabjan, D., & Bang, J. H. (2017). Classification-based financial markets prediction using deep neural networks. *Algorithmic Finance, 6*(3–4), 67–77.

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

Fujimoto, S., Hoof, H., & Meger, D. (2018). Addressing function approximation error in actor-critic methods. International conference on machine learning, *PMLR 1587-1596*.

Gobillon, L., & Magnac, T. (2016). Regional policy evaluation: Interactive fixed effects and synthetic controls. *Review of Economics and Statistics, 98*(3), 535–551.

Gomes, T. A., Carvalho, R. N., & Carvalho, R. S. (2017). Identifying anomalies in parliamentary expenditures of brazilian chamber of deputies with deep autoencoders. 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), *940-943)*.

Ha, D., & Schmidhuber, J. (2018). World models. *arXiv preprint* arXiv:1803.10122.

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

Hambly, B., Xu, R., & Yang, H. (2021). Recent Advances in Reinforcement Learning in Finance. *arXiv preprint* arXiv:2112.04553.

Han, J., Jentzen, A., & Weinan, E. (2018a). Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences, 115*(34), 8505–8510.

Heryadi, Y., & Warnars, H. L. H. S. (2017). Learning temporal representation of transaction amount for fraudulent transaction recognition using CNN, Stacked LSTM, and CNN-LSTM. In *2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)* (pp. 84–89).

Hosaka, T. (2019). Bankruptcy prediction using imaged financial ratios and convolutional neural networks. *Expert systems with applications, 117*, 287–299.

Hsu, P. Y., Chou, C., Huang, S. H., & Chen, A. P. (2018). A market making quotation strategy based on dual deep learning agents for option pricing and bid-ask spread estimation. In *2018 IEEE international conference on agents (ICA)* (pp. 99–104).

Iwasaki, H., & Chen, Y. (2018). Topic sentiment asset pricing with dnn supervised learning. *SSRN Electronic Journal*. 10.2139/ssrn.3228485.

Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., & Castaneda, A. G. (2019). Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science (New York, N.Y.), 364*(6443), 859–865.

Jeong, G., & 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*, 125–138.

Jiang, J., Kelly, B. T., & Xiu, D. (2020). (*Re*-)Imag(in)ing Price Trends. *SSRN Electronic Journal*. 10.2139/ssrn.3756587.

Jiang, Z., & Liang, J. (2017). Cryptocurrency portfolio management with deep reinforcement learning. In *2017 Intelligent Systems Conference (IntelliSys)* (pp. 905–913). IEEE.

Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P. E., He-Guelton, L., et al., (2018). Sequence classification for credit-card fraud detection. *Expert Systems with Applications, 100*, 234–245.

Kumar, B. S., & Ravi, V. (2016). A survey of the applications of text mining in financial domain. *Knowledge-Based Systems, 114*, 128–147.

Kumar, S., Kar, A. K., & Ilavarasan, P. V. (2021). Applications of text mining in services management: A systematic literature review. *International Journal of Information Management Data Insights, 1*(1), Article 100008.

Kushwaha, A. K., Kar, A. K., & Dwivedi, Y. K. (2021). Applications of big data in emerging management disciplines: A literature review using text mining. *International Journal of Information Management Data Insights, 1*(2), Article 100017.

Lanbouri, Z., & Achchab, S. (2015). A hybrid Deep belief network approach for Financial distress prediction. In *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)* (pp. 1–6).

Lee, C. Y., & Soo, V. W. (2017). Predict stock price with financial news based on recurrent convolutional neural networks. In *2017 conference on technologies and applications of artificial intelligence (TAAI)* (pp. 160–165).

Lee, S. I., & Yoo, S. J. (2020). Threshold-based portfolio: The role of the threshold and its applications. *The Journal of Supercomputing, 76*(10), 8040–8057.

Lei, K., Zhang, B., Li, Y., Yang, M., & Shen, Y. (2020). Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. *Expert Systems with Applications, 140*, Article 112872.

Li, X., Huang, Y. H., Fang, S. C., & Zhang, Y. (2020). An alternative efficient representation for the project portfolio selection problem. *European Journal of Operational Research, 281*(1), 100–113.

Li, X., Lv, Z., Wang, S., Wei, Z., & Wu, L. (2019). A reinforcement learning model based on temporal difference algorithm. *IEEE access : Practical innovations, open solutions, 7*, 121922–121930.

Li, Y., Lin, X., Wang, X., Shen, F., & Gong, Z. (2017). Credit risk assessment algorithm using deep neural networks with clustering and merging. In *2017 13th International Conference on Computational Intelligence and Security (CIS)* (pp. 73–176).

Liang, Z., Chen, H., Zhu, J., Jiang, K., & Li, Y. (2018). Adversarial deep reinforcement learning in portfolio management. arXiv preprint arXiv:1808.09940.

Lillicrap, T. P., .Hunt, J. J., .Pritzel, A., Heess, N., Erez, T., & Tassa, Y. et al. (2015).Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Lim, Q. Y. E., Cao, Q., & Quek, C. (2021). Dynamic portfolio rebalancing through reinforcement learning. *Neural Comput & Applic, 34*, 7125–7139.

Liu, D., Wei, Q., & Yan, P. (2015). Generalized policy iteration adaptive dynamic programming for discrete-time nonlinear systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 45*(12), 1577–1591.

Loughran, T., & McDonald, B. (2016). Textual analysis in accounting and finance: A survey. *Journal of Accounting Research, 54*(4), 1187–1230.

Luo, C., Wu, D., & Wu, D. (2017). A deep learning approach for credit scoring using credit default swaps. *Engineering Applications of Artificial Intelligence, 65*, 465–470.

Mahmoudi, N., Docherty, P., & Moscato, P. (2018). Deep neural networks understand investors better. *Decision Support Systems, 112*, 23–34.

Mavrotas, G., & Makryvelios, E. (2021). Combining multiple criteria analysis, mathematical programming and Monte Carlo simulation to tackle uncertainty in Research and Development project portfolio selection: A case study from Greece. *European Journal of Operational Research, 291*(2), 794–806.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., et al., (2016). Asynchronous methods for deep reinforcement learning. In *International conference on machine learning* (pp. 1928–1937).

Moher, D., Shamseer, L., Clarke, M., Ghersi, D., Liberati, A., Petticrew, M., & Stewart, L. A. (2015). Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement. *Systematic reviews, 4*(1), 1–9.

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

Neagoe, V. E., Ciotec, A. D., & Cucu, G. S. (2018). Deep convolutional neural networks versus multilayer perceptron for financial prediction. In *2018 International Conference on Communications (COMM)* (pp. 201–206).

Ng, J. Y. H., & Davis, L. S. (2018). Temporal difference networks for video action recognition. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (pp. 1587–1596).

Ngai, E. W., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision support systems, 50*(3), 559–569.

Otterlo, M. V., & Wiering, M. (2012). Reinforcement learning and markov decision processes. In *Reinforcement learning* (pp. 3–42). Berlin, Heidelberg: Springer.

Ozbayoglu, A. M., Gudelek, M. U., & Sezer, O. B. (2020). Deep learning for financial applications: A survey. *Applied Soft Computing, 93*, Article 106384.

Puterman, M. L. (2014). *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons.

Rawte, V., Gupta, A., & Zaki, M. J. (2018). Analysis of year-over-year changes in risk factors disclosure in 10-k filings. In *Proceedings of the Fourth International Workshop on Data Science for Macro-Modeling with Financial and Economic Datasets* (pp. 1–4).

Rönnqvist, S., & Sarlin, P. (2015). Detect & describe: Deep learning of bank stress in the news. In *2015 IEEE Symposium Series on Computational Intelligence* (pp. 890–897). IEEE.

Rönnqvist, S., & Sarlin, P. (2017). Bank distress in the news: Describing events through deep learning. *Neurocomputing, 264*, 57–70.

Saleh, H. (2020). *The the deep learning with pytorch workshop: Build deep neural networks and artificial intelligence applications with pytorch*. Packt Publishing Ltd.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning* (pp. 1889–1897).

Serrano, W. (2018). Fintech model: The random neural network with genetic algorithm. *Procedia Computer Science, 126*, 537–546.

Sharma, S., Rana, V., & Kumar, V. (2021). Deep learning based semantic personalized recommendation system. *International Journal of Information Management Data Insights, 1*(2), Article 100028.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., & Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science (New York, N.Y.), 362*(6419), 1140–1144.

Singh, V., Chen, S.-. S., & Schneider, M. (2022). *Anomaly detection in procure to pay business processes: A clustering and time series analysis-based approach* (SSRN Scholarly Paper ID 4012815). *Social Science Research Network*.

Singh, V., & Sharma, S. K. (2022). Application of blockchain technology in shaping the future of food industry based on transparency and consumer trust. *Journal of Food Science and Technology*, 1–18.

Sohangir, S., & Wang, D. (2018). Finding expert authors in financial forum using deep learning methods. In *2018 second IEEE international conference on robotic computing (IRC)* (pp. 399–402).

Théate, T., & Ernst, D. (2021). An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications, 173*, Article 114632.

Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M., & Iosifidis, A. (2017). Forecasting stock prices from the limit order book using convolutional neural networks. In *2017 IEEE 19th conference on business informatics (CBI)* (pp. 7–12). *1*.

Verma, S., Sharma, R., Deb, S., & Maitra, D. (2021). Artificial intelligence in marketing: Systematic review and future research direction. *International Journal of Information Management Data Insights, 1*(1), Article 100002.

Wang, H. (2019). Large scale continuous-time mean-variance portfolio allocation via reinforcement learning. Available at SSRN 3428125.

Wang, Y., & Xu, W. (2018). Leveraging deep learning with LDA-based text analytics to detect automobile insurance fraud. *Decision Support Systems, 105*, 87–95.

West, J., & Bhattacharya, M. (2016). Intelligent financial fraud detection: A comprehensive review. *Computers & security, 57*, 47–66.

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

Yang, H., & Xie, X. (2019). An actor-critic deep reinforcement learning approach for transmission scheduling in cognitive internet of things systems. *IEEE Systems Journal, 14*(1), 51–60.

Ying, J. J. C., Huang, P. Y., Chang, C. K., & Yang, D. L. (2017). A preliminary study on deep learning for predicting social insurance payment behavior. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 1866–1875).

Yu, L., Zhou, R., Tang, L., & Chen, R. (2018). A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data. *Applied Soft Computing, 69*, 192–202.

Zhang, K., Koppel, A., Zhu, H., & Basar, T. (2020). Global convergence of policy gradient methods to (almost) locally optimal policies. *SIAM Journal on Control and Optimization, 58*(6), 3586–3612.

Zhang, W., Li, L., Zhu, Y., Yu, P., & Wen, J. (2022). CNN-LSTM neural network model for fine-grained negative emotion computing in emergencies. *Alexandria Engineering Journal, 61*(9), 6755–6767.

Zhang, X., Zhang, Y., Wang, S., Yao, Y., Fang, B., & Philip, S. Y. (2018). Improving stock market prediction via heterogeneous information fusion. *Knowledge-Based Systems, 143*, 236–247.

Zhu, B., Yang, W., Wang, H., & Yuan, Y. (2018). A hybrid deep learning model for consumer credit scoring. In *2018 international conference on artificial intelligence and big data (ICAIBD)* (pp. 205–208). IEEE.