Global Finance Journal 62 (2024) 101016

Contents lists available at ScienceDirect

Global Finance Journal

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

Deep reinforcement learning for portfolio selection

Yifu Jiang [a], Jose Olmo [a,b,*], Majed Atwi [a]

[a] Department of Economic Analysis, University of Zaragoza, Gran Via 2, 50005 Zaragoza, Spain
[b] Department of Economics, University of Southampton, Highfield Campus, SO17 1BJ Southampton, United Kingdom

ARTICLE INFO

JEL codes:
C10
C45
C61
G10
G11
G17

Keywords:
Portfolio trading
Portfolio risk awareness
Transaction cost
Deep reinforcement learning
Portfolio constraint

ABSTRACT

This study proposes an advanced model-free deep reinforcement learning (DRL) framework to construct optimal portfolio strategies in dynamic, complex, and large-dimensional financial markets. Investors' risk aversion and transaction cost constraints are embedded in an extended Markowitz's mean-variance reward function by employing a twin-delayed deep deterministic policy gradient (TD3) algorithm. This study designs a DRL-TD3-based risk and transaction cost-sensitive portfolio that combines advanced exploration strategies and dynamic policy updates. The proposed portfolio method effectively addresses the challenges posed by high-dimensional state and action spaces in complex financial markets. This methodology provides two optimal portfolios by flexibly controlling transaction and risk costs with (i) the constituents of the Dow Jones Industrial Average and (ii) the constituents of the S&P100 index. Results demonstrate a strong portfolio performance of the proposed DRL portfolio compared to those of several competitors from the traditional and DRL literatures.

1. Introduction

Portfolio allocation is a key area of interest in finance. In recent decades, research on this area has led to the development of sophisticated investment strategies based on optimizing the risk-return tradeoff. A pioneering study in this field was conducted by Markowitz (1952), who introduced the mean-variance framework to construct optimal portfolios. This construct has evolved over the years in various directions, including the construction of portfolios based on the maximization of investor preferences or the development of sophisticated tools to optimize investors' short- and long-term objective functions. In recent years, machine learning (ML) techniques, in particular, are widely applied to risk management and optimal portfolio management tasks in dynamic financial markets (Henriques & Sadorsky, 2023). Reinforcement learning (RL; Sutton & Barto, 2018) and deep learning (DL; Goodell et al., 2021; Mavruk, 2022) methods have been widely used to solve portfolio selection optimization problems. Furthermore, RL is a promising method that does not require a specific portfolio baseline to make investment decisions and solely based on financial market information obtained at a given time period. The RL-based learning process is optimized by maximizing portfolio return or minimizing portfolio risk (Chaouki et al., 2020; Sutton & Barto, 2018) and enhancing portfolio forecasting accuracy by learning from errors.

The recent success of RL in portfolio allocation problems is related to the model's ability to improve expected returns and reduce portfolio risk. However, several important issues remain that must be resolved prior to applying these methods for portfolio investment to capture the complex and volatile characteristics of financial markets within the optimal portfolio decision rubric (Aboussalah & Lee, 2020; Bühler et al., 2018; Li et al., 2018; Moody et al., 1998; Moody & Saffell, 2001; Yang et al., 2020; Zhao et al., 2023). In addition to

* Corresponding author at: Department of Economic Analysis, Universidad de Zaragoza, Gran Vía 2, 50005 Zaragoza, Spain.
E-mail addresses: 849201@posta.unizar.es (Y. Jiang), joseolmo@unizar.es (J. Olmo), matwi@unizar.es (M. Atwi).

https://doi.org/10.1016/j.gfj.2024.101016
Received 14 December 2023; Received in revised form 23 May 2024; Accepted 11 July 2024
Available online 14 July 2024

maximizing investment returns, other factors such as transaction costs and investor risk tolerance must be considered in portfolio decision-making (Bühler et al., 2018; Li et al., 2018). Moreover, the selection of historical data for RL model training may be problematic (Yang et al., 2020). The characterization of important model features and state variables depends on the training phase, which is based on time-series data. Thus, the selection of specific datasets affects these characterizations and, in turn, the effectiveness of the corresponding portfolio selection strategy (Zhao et al., 2023). Furthermore, RL performance depends on model stability, convergence speed, and the ability of the algorithm to learn the underlying dynamic relationships between variables. A key advantage of these methods is their ability to manage high-dimensional portfolios very efficiently (Li et al., 2018; Yang et al., 2020).

This study proposes an advanced model-free deep reinforcement learning (DRL) framework that constructs optimal portfolios in high-dimensional settings. Specifically, this model combines DL and RL methods to create DRL framework and optimize portfolio strategies in dynamic, complex, and large-dimensional financial markets. Investor's risk aversion and transaction cost constraints are embedded in the model. Notably, DRL algorithms adopt neural networks as function approximators to automatically learn complex representations from high-dimensional data, thereby allowing them to capture key features and relationships among a variety of assets. The proposed model applies an adaptive portfolio trading method based on a twin-delayed deep deterministic policy gradient (TD3) algorithm. Thus, this study contributes to the DRL portfolio allocation literature by (i) building an investment strategy that naturally accommodates high-dimensional portfolios; (ii) embedding market risk, individual risk aversion, and transaction costs into the objective function; and (iii) solving Bellman type equations by combining twin networks, target networks, exploration strategies, and delayed policy updates that address the challenges posed by high-dimensional state and action spaces with nonlinear relationships.

The empirical performance of this strategy-building tool is assessed in an extensive out-of-sample exercise for high-dimensional portfolios with stocks from the Dow Jones Industrial Average and the S&P100 index using cumulative return, maximum drawdown (MDD), Sharpe ratio, and additional performance metrics. Our model outcomes are compared with those of minimum variance (MV), maximum Sharpe ratio, and other popular DRL methods designed for similar tasks. The empirical results confirm that our proposed model is superior in terms of profitability and risk trade-offs in the presence of transaction costs and different degrees of risk aversion.

The remainder of the study is organized as follows. Section 2 presents a brief literature review of DL, RL, and DRL models being applied to portfolio allocation problems. Section 3 introduces the proposed portfolio model's design, optimization formula, and the TD3-based algorithm. Section 4 discusses the empirical application and comparison results. Finally, Section 5 concludes the work.

2. Literature review

Rubesam (2022) investigated ML investment portfolio construction techniques and demonstrated the advantages of portfolio learning over traditional approaches. Mavruk (2022) considered ML methods to preselect stocks prior to the portfolio formation stage, and Fereydooni and Mahootchi (2023) implemented ML methods to improve investment decisions in financial markets. RL models are a type of ML that applies a different set of mathematical tools used for perception and representation learning. RL models are useful for sequential decision-making (Ngo et al., 2023). Goodell et al. (2021) provided a comprehensive review of ML finance tools, including how their hybridization can effectively address large and complex portfolio problems across various domains. They covered innovative techniques like the Markov decision process (MDP; Almahdi & Yang, 2017; Peck & Yang, 2011), Q-learning (QL), deep QL (DQL), proximal policy optimization (PPO), and deep deterministic strategy gradients (DDPG; Lillicrap et al., 2015). Additional techniques are commonly used for portfolio management, such as actor-critic (AC) networks (Aboussalah & Lee, 2020). Moody et al. (1998, 2001) designed a recurrent RL asset allocation method that used market data from the S&P500 index and US Treasury bill data.

QL has been applied to search for the optimal weights used to optimize portfolio asset allocations (Halperin, 2019; Zeng & Klabjan, 2018). The long short-term memory (LSTM) network improves the forecasting ability of DL and RL models, and Li et al (2021) used one to predict financial returns in global commodity markets. Similarly, Ta et al. (2020) investigated their application to quantitative trading and optimization-based stock prediction. Furthermore, Almahdi and Yang (2017, 2019) combined recurrent RL and a particle-swarm technique for portfolio weight allocation under market constraints, demonstrating that this method achieves better cumulative returns than those obtained from maximizing the contemporary Sharpe ratio and mean-variance approaches. Zhang and Maringer (2016) used a genetic algorithm to enhance portfolio trading performance with RL models.

Pigorsch and Schafer (2022) were among the first to develop a DQL portfolio trading strategy in a cross-sectional setting. Park et al. (2020) and Shavandia and Khedmati (2022) employed a multi-agent DQL algorithm to construct high-return portfolios. Huang and Tanaka (2022) designed a modularized and scalable multi-agent deep Q-network (DQN) to handle large-scale portfolios with heterogeneous data. Notably, DQL provides the algorithmic framework for DQN, which was initially tailored for discrete action spaces. Aboussalah and Lee (2020) addresses this limitation by suggesting stacked deep dynamic recurrent RL models. These models provide real-time portfolio management based on continuous actions in multidimensional state spaces. This approach maximizes expected returns while guaranteeing portfolio risk-tolerance constraints. Lin et al. (2020), and Wang and Ku (2022) applied DDPG portfolio strategies to maximize total returns while maintaining a risk diversification objective.

An important challenge for portfolio trading management is accommodating the underlying market risk while accounting for transaction costs (Kircher & Rsch, 2021; Moallemi & Saglam, 2015). Notably, transaction costs were ignored in earlier developments (Almgren & Chriss, 2001; Choi et al., 2019; Gaivoronski & Pflug, 2005; Li et al., 2018; Park et al., 2019). However, ignoring transaction costs can lead models to recommend over-aggressive portfolio trading. Some studies that include these costs as objective functions have partially addressed this issue (Ma et al., 2019; Qureshi et al., 2017; Roni & Jean-Luc, 1996; Zhang et al., 2011). Model-free RL algorithms were developed to adaptively control the impact of transaction costs. Building upon the work of Betancourt and Chen (2021), Zhao et al. (2023) applied DRL to develop portfolio policies and automatically execute transactions in such settings. A DRL-based portfolio management strategy on markets with transaction costs is proposed by Betancourt and Chen (2021). Similarly, Xu and

Dai (2022) used RL to derive hedging strategies that maximize profits while considering the effects of various transaction costs and underlying assets. Jang and Seong (2023) combined DRL with traditional portfolio theory, showing that DRL portfolios outperform other ML strategies. However, these authors did not incorporate risk aversion and transaction costs in the optimization of the investment portfolio. Recent work by Cui et al. (2023) provided a DRL algorithm that explicitly embeds the presence of risk alongside risk aversion objectives for portfolio decision-making. Garcia-Galicia et al. (2019), Wang and Zhou (2020) proposed models that consider transaction costs and mean-variance linear constraints into the portfolio objective functions. Furthermore, Moody and Saffell (2001) applied RL to optimize risk-adjusted returns while handling the impact of different transaction costs. Bühler et al. (2018) investigated how to use standard DRL algorithms to build nonlinear reward structures under transaction costs, liquidity, and risk-sensitive constraints, thus verifying the effectiveness of the portfolio trading method. Zhang et al. (2022) introduced transaction costs and risk-sensitive portfolio management methods based on DRL to maximize total returns. Li et al. (2018) and Yang et al. (2020) developed ensemble trading models using AC, PPO, and DDPG methods and embedded transaction costs and risk aversion factors into a novel cost-sensitive reward function. Moreover, Sebastian et al. (2021) proposed a DRL-based repeated portfolio method that accounts for asset variability and the existence of mutual asset correlations. The empirical results confirmed the ability of the method to achieve superior performance over previous RL and traditional portfolio selection methods.

High-dimensional portfolios benefit greatly from the advantages of diversification when containing a large number of assets. However, their management strategies require data-rich environments that present an obstacle to normal portfolio construction (Fernandez-Arjona & Filipovic, 2022). Pigorsch and Schafer (2022) is one of the first studies to propose a DQL method that constructs high-dimensional portfolios in a cross-sectional setting. Meanwhile, Bühler et al. (2018) developed an advanced DRL method for hedging a portfolio of derivatives under market constraints, including transaction costs, liquidity limits, and different risk-tolerance levels. The empirical results confirm the superiority of the proposed DRL methods against standard approaches to portfolio optimization in large dimensions. Notably, the performance advantage increases with the number of assets in the portfolio.

3. Model and methods

The first block of this section introduces the portfolio choice problem under the presence of transaction costs and investor risk aversion constraints. The second block introduces MDP for portfolio trading decisions, and the third one describes the proposed DRL-based portfolio and portfolio selection methodology.

3.1. *Asset allocation problem under portfolio constraints*

Consider a financial market with N risky assets that has been trading for T periods, and let $\boldsymbol{p}_t = \left[p_{1,t}, \cdots, p_{N,t}\right]^{\mathbf{T}} \in \mathbb{R}_+^N$ denote the vector of asset prices such that $p_{n,t}$ indicates the closing price of the n-th asset at time t. Similarly, let $m_{n,t} \in \mathbb{Z}_+$ and $P_t \in \mathbb{R}$ denote the n-th asset shares and portfolio value at time t, respectively. Investors perform portfolio trading strategies over the N assets during each period, including buying, selling, and holding, which results in increasing, reducing, and no changes to asset shares, respectively. The portfolio selection decision variable is denominated in this literature as the trading action and is characterized by $k_{n,t}$ for asset n at time t. For each asset, this variable takes three possible values $\{-1,0,1\}$, denoting selling, holding, and buying, respectively. This strategy can be extended to consider multiple shares of each asset such that the domain of $k_{n,t}$is $\{-k, \cdots, -1, 0, 1, \cdots, k\}$, where -k and k denote the net number of shares that a trader can sell or buy, respectively, in a given period. Traders can set a maximum number of shares,$k \leq m_{\max}$, so that at each period t the number of shares on each asset n is given by $m_{n,t+1} = m_{n,t} + k_{n,t}$. The value of the portfolio at time t is given by the following equation:

$$P_t = P_{t-1} + \boldsymbol{p}_t^{\mathbf{T}} \boldsymbol{k}_t \tag{1}$$

where $\boldsymbol{p}_t^{\mathbf{T}}$ is the transpose of price vector $\boldsymbol{p}_t$, and $\boldsymbol{k}_t = \left[k_{1,t}, \cdots, k_{N,t}\right]^{\mathbf{T}}$.

Each trading process (buying, holding, or selling) generally involves transaction costs and it is necessary to consider this factor in the portfolio selection problem. Let ξ denote a positive constant that characterizes the transaction cost rate of issuing a new trade. Thus, the transaction cost is expressed as (Yang et al., 2020):

$$c_t^{\text{tran}} = \xi \times p_t^T |k_t| \tag{2}$$

For simplicity, we assume that the constant is fixed across assets, although this can be modified to accommodate the presence of different transaction costs and different liquidities.

Underlying risk is an important aspect to consider in an objective function. We differentiate two sources of risk that influence asset allocation, namely, true risk of the portfolio position, captured by portfolio variance σ_t^2, and investor risk aversion coefficient β. Here, the variance of the portfolio position is proxied by the sample variance of the portfolio return over the last t days, defined as $\sigma_t^2 = \frac{1}{t}\sum_{i=1}^{t}\left((P_i - P_{i-1})/P_{i-1} - \mu_t\right)^2$, where μ_t is the average return of assets over the last t periods. The literature takes different approaches to introduce risk aversion coefficient in portfolio optimization schemes. The most relevant tactic in financial economics is to leverage a utility function to model investor's preferences. Popular utility functions include constant absolute risk aversion and constant relative risk aversion (Chambers & Quiggin, 2007). Herein, we follow Markowitz's approach and introduce coefficientβthat reflects the investor's degree of risk aversion. To model the investor's attitude towards the underlying portfolio risk, we incorporate a cost function in the objective function as follows:

$$c_t^{\text{risk}} = \beta\sigma_t^2 \tag{3}$$

3.2. Markov decision process for portfolio trading

The portfolio trading decision is modeled as an MDP, characterized by a tuple $(\mathscr{S}, \mathscr{A}, \mathscr{P}, r)$ in which $\mathscr{S}$ is the state space, $\mathscr{A}$ is the action space, $\mathscr{P}(s_{t+1}|s_t, a_t)$ is the transition probability of state $s_{t+1} \in \mathscr{S}$, given the realization of action $a_t \in \mathscr{A}$ based on the observed state $s_t \in \mathscr{S}$ at the t-th learning time step. Similarly, $r_t(s_t, a_t, s_{t+1})$ is the reward function obtained from taking action a_t at state s_t under the realization of state s_{t+1} at time t + 1.

The state space is defined by a vector of state variables such that $s_t = [\boldsymbol{p}_t, \boldsymbol{m}_{t-1}] \in \mathscr{S}$. Similarly, action space $\mathscr{A}$ contains the set of possible actions, $a_t \in \mathscr{A}$. In the portfolio allocation problem, each action is defined as the quantity of an asset a trader wishes to buy, sell, or hold in a given period such that $a_t \in \{-k, \cdots, -1, 0, 1, \cdots, k\}$. Here, we use a_t and $k_{n,t}$ interchangeably. The action space can be normalized to [−1,1], which is a continuous action space that can be used for continuous-action RL algorithms.

For each state of nature and possible action, we define a portfolio policy$\pi(a_t, s_t)$that describes the trading strategy in a given scenario. Similarly, the reward function$r_t(s_t, a_t, s_{t+1})$is obtained by implementing the portfolio policy after state s_{t+1} is realized. We propose the following reward function that includes the variation in the value of the portfolio as well as the presence of transaction costs. The function accounts for risk by penalizing the volatility of the portfolio by the degree of the investor's risk aversion. The objective reward function is expressed as:

$$r_t = \boldsymbol{p}_t^{\text{T}}\boldsymbol{m}_t - \underbrace{\beta\sigma_t^2}_{\text{Risk cost}} - \underbrace{\xi\boldsymbol{p}_t^{\text{T}}|\boldsymbol{k}_t|}_{\text{Transaction cost}} \tag{4}$$

In DRL, the agent is the entity that executes decision-making, interacts with the environment, and learns how to choose actions to maximize cumulative returns and improve strategy to obtain better results. Specifically, the objective of the agent is to learn an optimal trading policy $\pi(s_t, a_t)$ that selects an action to maximize the reward function. Policies are evaluated using an action-value function, $Q_\pi(s_t, a_t)$ (i.e., the Q-function). Wiering and Otterlo (2012) expressed this function as a Bellman equation:

$$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}}\left[r_t(s_t, a_t, s_{t+1}) + \gamma\mathbb{E}_{s_{t+1}, a_{t+1}}[Q_\pi(s_{t+1}, a_{t+1})]\right] \tag{5}$$

where $0 < \gamma < 1$ is the discount factor, and $\mathbb{E}_{s_{t+1}}[\cdot]$ is the conditional expectation on the realization of state s_{t+1}.

QL is one of the most common algorithms in RL. During the learning process, the action-value $Q_\pi(s_t, a_t)$ function is continuously updated recursively to obtain the maximum cumulative reward, thereby obtaining the optimal trading strategy. The temporal difference error quantifies the difference between the expected and actual Q-value as the agent interacts with the environment and moves from one state to another by taking an action. The updated Q-value is determined by multiplying the temporal difference error and learning rate and adding it to the current Q-value (Wiering & Otterlo, 2012). Thus, we have:

$$Q_\pi^{New}(s_t, a_t) \leftarrow Q_\pi^{Old}(s_t, a_t) + \alpha\left(r_t(s_t, a_t, s_{t+1}) + \gamma \max_{a_{t+1}\in\mathscr{A}} Q_\pi\left(s_{t+1}, a_{t+1}\right) - Q_\pi^{Old}(s_t, a_t)\right) \tag{6}$$

where α is the learning rate, $0 < \alpha < 1$, γ is the discount factor, $Q_\pi^{New}(s_t, a_t)$ is the updated Q-value of $Q_\pi^{Old}(s_t, a_t)$, $r_t(s_t, a_t, s_{t+1})$ is the reward when the agent executes action a_t at state s_t and receives the new state, s_{t+1}, from the market environment. $\max_{a_{t+1}\in\mathscr{A}} Q_\pi(s_{t+1}, a_{t+1})$indicates the maximum Q-value after selecting action a_{t+1} at state s_{t+1}, $Q_\pi^{Old}(s_t, a_t)$ on the right side of (6) denotes the estimated Q-value under action a_t at state s_t, and $r_t(s_t, a_t, s_{t+1}) + \gamma \max_{a_{t+1}\in\mathscr{A}} Q_\pi(s_{t+1}, a_{t+1}) - Q_\pi^{Old}(s_t, a_t)$ is the temporal difference error. This function represents the signal used to adjust estimates over time (Wiering & Otterlo, 2012).

Iterating (6) allows the Q-value function to gradually converge to the optimal Q-function as the agent iteratively interacts with the environment and learns how to make better decisions that maximize the cumulative reward. Moreover, learning rate α plays a key role as it controls the frequency of updates for each Q-value. Specifically, it determines the degree of balance between the new and old values. The investor observes the arrival of new information in each state (e.g., number of assets and asset prices), followed by selecting an available action with optimized policy π that maximizes the Q-value. Subsequently, the investor obtains an instantaneous reward from the market environment, which is used to evaluate the quality of the selected action, thereby resulting in an adjusted portfolio strategy based on the evaluation. The detailed implementation of the learning processes by the investor (agent) is described at the end of the following section.

3.3. TD3-based portfolio trading algorithm

We adopted the TD3 algorithm to search the optimal portfolio trading strategy in dynamic financial markets. TD3 extends and improves the DDPG algorithm by introducing new features to make it increasingly stable during training and to improve convergence speeds. Furthermore, DDPG is an AC algorithm that extends the deterministic policy gradient algorithm to continuous action spaces. It employs a neural network as the actor to approximate the optimal policy and another neural network as the critic to approximate the state-action value function. The actor network directly outputs the deterministic action, given the current state. The critic network learns the Q-value function by taking the state and action as inputs. TD3 algorithm includes a convolutional neural network (CNN) or LSTM actor network with weight $\boldsymbol{\psi}$ and two critic networks with weights $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. The critic networks more effectively avoid overestimating the Q-value function and are, therefore, able to achieve better learning performance. After receiving the portfolio trading policy, $\pi(s_t, a_t)$, from the actor network, the two critic networks update their loss functions as follows:

$$L(\theta_i) = \frac{1}{I}\sum_{t}^{t+i}(y_t - Q_\pi(s_t, a_t|\theta_l))^2, l = 1, 2 \tag{7}$$

where I is the size of the replay buffer, which is the amount of information used for model training, i is the i-th sample of the replay buffer, and the information set in the replay buffer is denoted as $\mathscr{D}$. As the TD3 has two critic networks, $l$denotes the l-th critic network, and $l = 1, 2$. The difference between the Q-value function$Q_\pi(s_t, a_t|\theta_l)$in (7) and $Q_\pi(s_t, a_t)$in (6) is that $Q_\pi(s_t, a_t|\boldsymbol{\theta}_l)$ adopts the CNN backbone and optimizes neural network weights, $\boldsymbol{\theta}_l$, using information from the available dataset. y_t is the target Q-value function, expressed as:

$$y_t = r_t(s_t, a_t, s_{t+1}) + \gamma Q_\pi\left(s_{t+1}, a_{t+1}|\boldsymbol{\theta}_l'\right) \tag{8}$$

where $\boldsymbol{\theta}_l'$ are the updated neural network weights of the l-th critic network.

In TD3, the target Q-value function, y_t, plays a key role in training the portfolio learning model. A target Q-value is adopted in various DRL algorithms to update the Q-values during the learning process, which represents the estimated maximum one-period ahead reward that the agent can achieve by selecting the best action in the next state and discounting it by γ. This target Q-value y_t is used to update the current Q-value function $Q_\pi(s_t, a_t|\boldsymbol{\theta}_l)$. Then, the actor network uses the policy gradient method to update

$$\nabla_\psi J(\psi) \approx \mathbb{E}_{s_t}\left[\nabla_a Q\left(s_t, a_t|\theta_l\right)|_{a_t=\pi(s_t;\psi)} \nabla_\psi \pi(s_t|\psi)\right] \tag{9}$$

Because a traditional DDPG can sometimes converge to the local optimal portfolio policy, TD3 introduces the following three techniques to avoid this. The first involves target policy smoothing, a technique that reduces the estimation bias of the Q-value and improves model generalization by adding noise on target policy output. The equation for target strategy smoothing is expressed as follows:

$$\widetilde{a} = \pi(s_{t+1}|\boldsymbol{\psi}) + \varsigma, \varsigma \sim clip(\mathscr{N}(0, \widetilde{\delta}), -\kappa, \kappa) \tag{10}$$

where $\pi(s_{t+1}|\boldsymbol{\psi})$ is the action output from the main network, $\widetilde{a}$ is the target action after adding noise, ς is Gaussian noise after clipping with variance $\widetilde{\delta}^2$ for policy smoothing, and κ is the clipping amplitude.

The second technique is double-QL, which reduces the overestimation of Q-values by maintaining two critic network Q-value functions. The formula for double-QL is expressed as:

$$y = r_t + \gamma \cdot min\left(Q_1'\left(s_{t+1}, a'|\boldsymbol{\theta}_1'\right), Q_2'\left(s_{t+1}, a'|\boldsymbol{\theta}_2'\right)\right) \tag{11}$$

where y is the target Q-value, r_t is the actual reward, γ is the discount factor, Q_1' and Q_2' are the Q-value functions of the two target networks used to calculate the minimum Q-value.

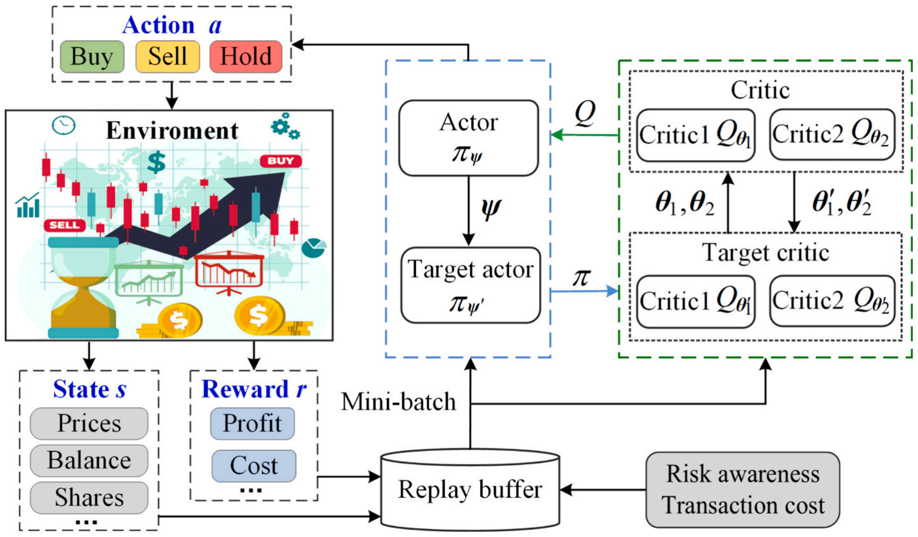

Fig. 1. TD3-based portfolio trading framework.

Next, according to (8) and (9), the portfolio trading agent updates the weights of the three online networks using the following equations:

$$\begin{aligned} \boldsymbol{\theta}_{l,t} &= \boldsymbol{\theta}_{l,t-1} + \alpha_c L\left(\boldsymbol{\theta}_{l,t-1}\right)\boldsymbol{\theta}_{l,t-1} \\ \boldsymbol{\psi}_t &= \boldsymbol{\psi}_{t-1} + \alpha_a \nabla_{\boldsymbol{\psi}_{t-1}} J(\boldsymbol{\psi}_{t-1})\boldsymbol{\psi}_{t-1} \end{aligned} \tag{12}$$

where α_a and α_c are the learning rates of the actor and critic networks, respectively.

The target networks are updated periodically by tracking the main networks using the following update rule:

$$\begin{aligned} \boldsymbol{\theta}'_{l,t} &= \tau\boldsymbol{\theta}_{l,t-1} + (1-\tau)\boldsymbol{\theta}'_{l,t-1} \\ \boldsymbol{\psi}'_t &= \tau\boldsymbol{\psi}_{t-1} + (1-\tau)\boldsymbol{\psi}'_{t-1} \end{aligned} \tag{13}$$

where τ is the hyperparameter that controls the rate of target network updates.

The third technique is the delayed policy update, which reduces the update frequency of the actor network and forces the critic network to update more frequently than the actor, thus improving stability and convergence speed.

Fig. 1 presents the TD3-based portfolio trading framework, which aims to devise a portfolio trading strategy that maximizes the portfolio reward$r_t(s_t, a_t, s_{t+1})$in a dynamic environment. The learning agent (i.e., investor) collects the market state$s_t$to train the learning model, where the risk awareness behavior of the investor and transaction cost are input to the model through the reward function and included in the training step. Subsequently, a portfolio actiona_t (i.e., buy, sell, or hold), is selected as a combination to maximize the Q-value function. After executing the selected action, the investor receives reward r_t from the financial market and moves one period ahead. Thus, a new set of states, s_{t+1}, and possible actions are encountered at time t + 1. The replay buffer stores all training samples from which the agent randomly selects a mini-batch sample set at each training epoch to train the portfolio learning model.

The TD3 method for optimal portfolio allocation is listed in Algorithm 1. To ease the interpretation of the method, the main steps of the algorithm are summarized as follows:

Step 1: Initialize the TD3-based learning framework, including actor $\pi(s|\boldsymbol{\psi})$ and critics $Q_1(s, a|\boldsymbol{\theta}_1)$ and $Q_2(s, a|\boldsymbol{\theta}_2)$with random weights $\boldsymbol{\psi}$, $\boldsymbol{\theta}_1$, and $\boldsymbol{\theta}_2$, respectively, and initialize the replay buffer set, $\mathscr{D}$.

Step 2: Observe state s from the trading market, which contains asset prices and market shares.

Step 3: Input the financial market information into the learning framework, including the asset prices, number of assets, etc.

Step 4: Select one available action, a (buying, holding, or selling), to maximize the reward function, r_t, and execute the action before observing state s_{t+1}, which is realized in the next period.

Step 5: Store transition (s_t, a_t, r_t, s_{t+1}) into the replay buffer, $\mathscr{D}$, where the learning agent will sample a mini-batch of transitions (s_t, a_t, r_t, s_{t+1}).

Step 6: Update the actor and critic networks by applying the gradient based method of (7), (8), (12), and (13).

Step 7: The TD3 model is fully trained after a certain number of learning episodes, when the training loss becomes less than a given tolerance level or all sampled data are completely trained.

Algorithm 1. Portfolio risk and transaction cost-aware TD3-based portfolio trading.

```
Input the maximum number of episodes, $I$, the maximum number of steps per episode, $T$, exploration noise standard deviation, $\delta$, policy noise standard deviation, $\tilde{\delta}$, clip factor, $\kappa$, mini-batch size, $M$, discount factor, $\gamma$, target policy smoothing coefficient, $\tau$, and delay parameter, $d$. The financial market information is also inputted in the learning system.
Initialize actor network $\pi(s|\boldsymbol{\psi})$ and critic networks $Q_1(s,a|\boldsymbol{\theta}_1)$ and $Q_2(s,a|\boldsymbol{\theta}_2)$ with random weights $\boldsymbol{\psi}$, $\boldsymbol{\theta}_1$, and $\boldsymbol{\theta}_2$, respectively.
Initialize target actor network $\pi(s|\boldsymbol{\psi}')$ and target critic networks $Q_1(s,a|\boldsymbol{\theta}_1')$ and $Q_2(s,a|\boldsymbol{\theta}_2')$ with weights $\boldsymbol{\psi}' \leftarrow \boldsymbol{\psi}$, $\boldsymbol{\theta}_1' \leftarrow \boldsymbol{\theta}_1$, and $\boldsymbol{\theta}_2' \leftarrow \boldsymbol{\theta}_2$, respectively.
Initialize replay buffer $\mathcal{D}$.
For episode i in $1,2,...,I$ do the following:
  For step t in $1,2,...,T$ do the following:
    Observe state $s_t$ in the financial market.
    Select the action with exploration noise $a = \pi(s_t|\boldsymbol{\psi}) + \varsigma, \varsigma : clip(\mathcal{N}(0,\delta), -\kappa, \kappa)$.
    Execute action $a_t$ and observe the next state, $s_{t+1}$, and reward $r_t$.
  Store transition $(s_t, a_t, r_t, s_{t+1})$ in replay buffer $\mathcal{D}$.
  Sample the mini-batch of transitions $(s_t, a_t, r_t, s_{t+1})$ from replay buffer $\mathcal{D}$.
  Compute target actions $\tilde{a} = \pi(s_{t+1}|\boldsymbol{\psi}) + \varsigma, \varsigma : clip(\mathcal{N}(0,\delta), -\kappa, \kappa)$.
  Compute target values $y = r_t + \gamma \cdot \min(Q_1'(s_{t+1}, a'|\boldsymbol{\theta}_1'), Q_2'(s_{t+1}, a'|\boldsymbol{\theta}_2'))$.
  Update the critic networks by minimizing the mean-squared error loss function:
  $L(\boldsymbol{\theta}_i) = \frac{1}{I}\sum_{t=1}^{t+i}(y_t - Q_\pi(s_t, a_t|\boldsymbol{\theta}_l))^2,\ l = 1,2$.
  If $t\ mod\ d = 0$,
    Update the policy by applying the gradient:
    $\nabla_{\boldsymbol{\psi}} J(\boldsymbol{\psi}) \approx \mathrm{E}_{\boldsymbol{s_t}}[\nabla_{\boldsymbol{a}} \boldsymbol{Q}(s_t, a_t|\boldsymbol{\theta}_l)|_{a_t=\pi(s_t;\psi)} \nabla_{\boldsymbol{\psi}} \pi(s_t|\boldsymbol{\psi})]$.
    Update the target networks:
    $\boldsymbol{\theta}_{l,t}' = \tau\boldsymbol{\theta}_{l,t-1} + (1-\tau)\boldsymbol{\theta}_{l,t-1}';\ \boldsymbol{\psi}_t' = \tau\boldsymbol{\psi}_{t-1} + (1-\tau)\boldsymbol{\psi}_{t-1}'$.
  End for
  End for
```

4. Empirical application

This section provides an empirical illustration of the aforementioned methodology and algorithms to construct investment portfolios. We use the Yahoo Finance database and select the 30 constituent stocks of the Dow Jones Industrial Average (DJIA) and the 100 constituents of the S&P100 index as the trading stock pool. Furthermore, we exploit historical daily closing price data from April 1, 2010 to March 9, 2023 for the performance evaluation. Our empirical application has three stages, including training, validation, and trading testing, where the entire dataset is divided into three parts. The daily closing price data from April 1, 2010 to January 2, 2020 and January 3, 2020 to April 29, 2021 are utilized for the learning agent model training and validation, respectively. Notably, the validation stage assesses the performance of the method using in-sample information to evaluate the quality of the portfolio training model. Subsequently, the trained model was used to test the trading performance based on the testing dataset in an out-of-sample period from April 30, 2021 to March 9, 2023.

Portfolio performance is usually assessed by comparing the metrics of portfolio competitors. The first metric that we consider is the cumulative portfolio value, which measures the increase in portfolio value at the end of the investment period. Here, the cumulative portfolio value P_T is obtained as the net increment of the portfolio over time. Combining (1) and (2), we obtain the net value of the portfolio P_T^{net}, defined as follows:

$$P_T^{\text{net}} = P_0 + \sum_{t=1}^{T} \boldsymbol{p}_t^{\mathrm{T}} (\boldsymbol{k}_t - \xi|\boldsymbol{k}_t|) \tag{14}$$

where P_0 is the initial portfolio trading wealth, which is set to $P_0 = 1$ and T denotes the length of the investment time period. The transaction cost rate (ξ) is taken into account in the cumulative portfolio. The final cumulative return, S_T, after T, is defined as follows:

$$S_T = \frac{P_T^{\text{net}} - P_0}{P_0} \tag{15}$$

The cumulative value neglects the presence of risk in the portfolio as it reports only the cumulative gain. A complementary performance measure widely used in the literature is the Sharpe ratio, which measures the portfolio return per risk unit. The Sharpe ratio is defined as:

$$SR = \frac{\mathbb{E}[\rho_t] - \rho_f}{\sigma_{\rho_t}} \tag{16}$$

where ρ_t denotes the portfolio return, $\rho_t = (P_t - P_{t-1})/P_{t-1}$, ρ_f is the risk-free return, and σ_{ρ_t} is the unconditional volatility of ρ_t, defined as $\sigma_{\rho_t} = \sqrt{\text{Var}(\rho_t)}$.

A related metric that captures the portfolio maximum potential loss faced by investors is the MDD measure. This is defined as the largest loss from peak to trough:

$$MDD = \max_{t:l>t} \frac{P_t^{\text{net}} - P_l^{\text{net}}}{P_t^{\text{net}}} \tag{17}$$

A mixture of the MDD risk measure and the Sharpe ratio is the Calmar ratio (CR), which provides an alternative characterization of the portfolio risk-adjusted value to evaluate performance. Investors can use this metric to identify portfolios that align with their risk appetite and investment objectives, where CR is defined as:

$$CR = \frac{P_T^{\text{net}}}{MDD} \tag{18}$$

Trading performance comparisons are presented using the following models:

- Our proposed risk and transaction cost-sensitive (RTC) portfolio based on the TD3 algorithm combined with CNN, denoted RTC-CNN-TD3.
- Our proposed RTC portfolio based on the TD3 algorithm combined with LSTM, denoted RTC-LSTM-TD3.
- The risk and transaction cost-sensitive portfolio based on the RTC-DDPG algorithm combined with CNN, denoted RTC-CNN-DDPG.
- The risk and transaction cost-sensitive portfolio based on the RTC-PPO algorithm combined with CNN, denoted RTC-CNN-PPO, where PPO is a common RL portfolio trading method (Aboussalah et al., 2022; Li, Liu, et al., 2021).
- The minimum variance portfolio trading method, which aims to minimize portfolio risk, denoted min-variance (MV).
- The maximum Sharpe ratio portfolio, which maximizes the Sharpe ratio, denoted Max-Sharpe.

The hyperparameter values for portfolio optimization are provided in Table 1. To improve the learning efficiency of the proposed DRL algorithm, a suitable parameter must be selected. For the learning rate, if we set a learning rate that is too small (e.g., a = 0.0001), it will take longer to achieve convergence. On the contrary, if the learning rate is set too large (e.g., a = 0.01), the method leads to significant fluctuations in the model parameters and destabilizes the training processes. Hence, we select a suitable learning rate (i.e., a = 0.001) in the training model.

Similarly, the number of hidden layers and their sizes should be moderated. If we set numerous layers, it may render an overly complex model that demands high computational complexity. However, limited number of layers may result in poor performance because it would lack the capacity to learn effectively. The remaining hyperparameters are standard values for DRL frameworks (Sutton & Barto, 2018).

4.1. Empirical results for DJIA stocks

Fig. 2 and Table 2 present the backtesting portfolio trading performance on 30 DJIA stocks for a risk aversion coefficient of β = 0.005 and a transaction cost rate of ξ = 0.05%. The first scenario hardly penalizes the presence of risk and transaction costs, and all results were performed on the out-of-sample evaluation (testing) period. Fig. 2 illustrates the strong performance of the three DRL-based portfolio methods (i.e., TD3, PPO, and DDPG). These portfolios achieve a higher cumulative return than those managed by

Table 1
Hyperparameter values for portfolio optimization.

Hyperparameter	Value	Hyperparameter	Value
Actor learning rate	10^{-4}	Second hidden layer size	512
Critic learning rate	10^{-4}	Target policy coefficient	10^{-4}
Optimizer	Adam	Max. episode number	1000
Discount factor	0.98	Replay buffer Size	106
Mini-batch size	64	Policy noise	0.02
Number of hidden layers	2	Policy noise clip	0.05
First hidden layer size	512	Exploration noise standard deviation	0.15

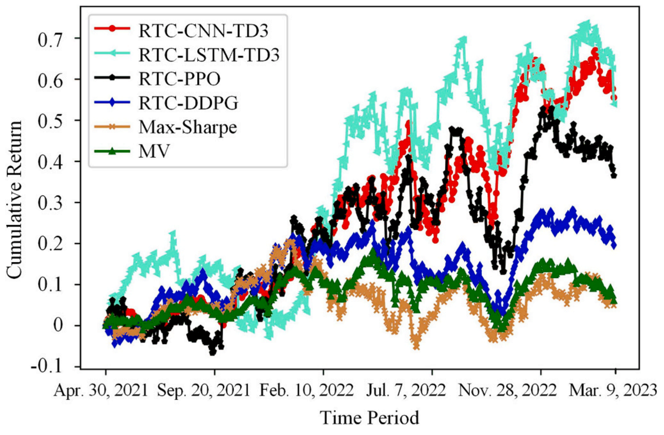

Fig. 2. Cumulative return performance comparisons using different investment strategies for a risk aversion coefficient of $\beta = 0.005$ and a transaction cost rate of $\xi = 0.05\%$.

Table 2
Performance measures of different portfolio methods when $\beta = 0.005$ and $\xi = 0.05\%$.

Method	RTC-CNN-TD3	RTC-LSTM-TD3	RTC-CNN -DDPG	RTC-CNN-PPO	Max-Sharpe	MV
Annual Return (%)	26.91	26.20	10.18	18.28	2.62	3.25
Cum. Return (%)	55.53	53.92	19.62	36.50	4.91	6.10
Annual Volatility (%)	22.01	29.02	17.28	28.08	19.97	13.42
Sharpe Ratio	1.19	0.95	0.65	0.74	0.23	0.31
Max Drawdown (%)	19.11	20.61	18.58	23.52	21.38	16.01
Calmar Ratio	1.41	1.27	0.55	0.78	0.12	0.20

the other two benchmark methods (i.e., Max-Sharpe and MV). Table 2 reports the performance metrics. The RTC-CNN-TD3 portfolio has an annualized return of 26.91%, which is significantly higher than those of Max-Sharpe (2.62%) and MV (3.25%). Moreover, our proposed DRL portfolio method effectively captures the dynamic patterns of each asset price, and the trading performance can be optimized accordingly. From Fig. 2 and Table 2, we can observe that both the RTC-CNN-TD3 and RTC-LSTM-TD3 portfolios have similar performance, where the CNN and LSTM methods have the most structurally comparable structure for portfolio feature extraction.

Table 2 shows that the Sharpe ratio of the RTC-CNN-TD3 method is the highest, indicating that this strategy outperformed the rest in terms of risk/return tradeoff. Furthermore, the RTC-CNN-TD3 method achieves the highest annual return and CR during the testing period. On the downside, this strategy exhibits higher annual volatility than the two benchmark methods. Interestingly, we observe that the RTC-CNN-PPO portfolio may not be a suitable investment strategy because of its poor performance in terms of annual volatility (28.08%) and MDD (23.52%). These results demonstrate that DRL-driven portfolios can adequately handle dynamic financial data. In

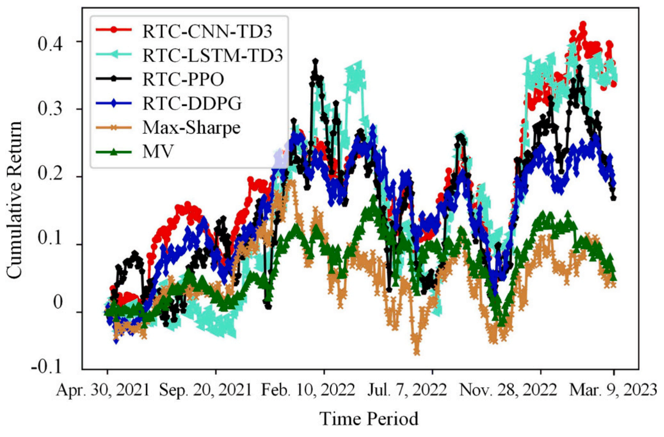

Fig. 3. Cumulative return performance comparisons using different portfolio trading strategies for a risk aversion coefficient of $\beta = 0.01$ and a transaction cost rate of $\xi = 0.1\%$.

particular, the RTC-CNN-TD3 method generally outperforms the others.

We further evaluate the portfolio performance of the six competing methods by increasing the values of the two cost-sensitive parameters (i.e., risk aversion coefficient β and transaction cost rate ξ). Fig. 3 and Table 3 present the performance measures of the six methods when $\beta = 0.01$ and $\xi = 0.1\%$. Compared with the results shown in Fig. 2 and Table 2, the cumulative returns of all portfolios decrease to different extents. This is because the increased risk aversion and transaction cost levels discourage risk-taking positions and high portfolio turnover. Moreover, the three DRL-based portfolio methods have similar cumulative returns in the period from April 30, 2021 to September 20, 2021. Interestingly, the proposed RTC-CNN-TD3-based portfolio achieves the highest value (approximately 33.69%) at the end of the out-of-sample period on March 9, 2023. In contrast, the Max-Sharpe and MV portfolios achieve cumulative returns of only 3.97% and 5.11%, respectively. As expected, as the two cost-sensitive parameters (i.e., β and ξ) increase, the annual volatility of the methods drops. The Sharpe ratio also takes lower values than the ones shown in Table 2; however, our proposed method still significantly outperforms the competitors, indicating that the RTC-CNN-TD3-based portfolio strategy balances risk and return. From Tables 2 and 3 we find that although both RTC-CNN-TD3 and RTC-LSTM-TD3 achieve comparable performances, RTC-LSTM-TD3 exhibits a higher volatility, suggesting that risk-averse investors may prefer RTC-CNN-TD3.

We further gauge the effect of the transaction cost rate, ξ, on portfolio performance. Unsurprisingly, Fig. 4 and Table 4 reveal that the cumulative return declines as the transaction cost rate increases. Particularly, the RTC-CNN-TD3-based portfolio strategy achieves high cumulative returns when the transaction cost rate is comparatively low (e.g., 0.01% and 0.05%) but reduces by approximately 50% when the transaction cost rate raises to 0.1% and 0.5%. The reason lies in the fact that the transaction cost level has a sizable effect on the profitability of the strategy and investor's behavior. In addition, as shown in Table 4, increases of ξ are accompanied by decreases in Sharpe ratio, MDD, and CR, especially when ξ is relatively large (e.g., $\xi = 0.5\%$). These empirical findings suggest that the investment strategy is very sensitive to frequent changes in portfolio composition.

Finally, we evaluate the effect of the risk aversion coefficient β on the performance of the proposed RTC-CNN-TD3 method (see Fig. 5 and Table 5). As expected, increases in risk aversion are accompanied by decreases in the annual volatility of the proposed portfolio over the testing dataset, indicating the effectiveness of the proposed cost-sensitive method in incorporating investor's attitude towards risk. Furthermore, the MDD value declines for larger values of β. As this value is determined by asset price volatility, the results confirm that constraining the volatility of portfolio returns can assist investors in managing downside risks.

4.2. *Empirical results for S&P100 stocks*

This section compares the portfolio performance of the methods in a high-dimensional setting given by the constituents of the S&P100 index. Fig. 6 illustrates the cumulative return performance of the out-of-sample dataset. As expected, the three portfolio methods based on DRL (i.e., TD3, PPO, and DDPG) achieve better performance than the traditional Max-Sharpe and MV strategies. This is because the proposed learning methods are capable of effectively searching for the optimal portfolio decision strategy in complex, uncertain, dynamic, and large-scale financial trading markets.

Furthermore, as shown in Table 6, the proposed two TD3-based portfolio methods achieve the highest annual return and Sharpe ratio in this high-dimensional setting, confirming their ability to balance risk and return. Similar to the previous results, the MV method shows the worst portfolio performance in almost all indicators, apart from the MDD measure. From Fig. 6 and Table 6, we observe that the RTC-LSTM-TD3 method obtains a slightly higher cumulative return than the RTC-CNN-TD3 portfolio at the expense of higher volatility and MDD, posing increased risk for investors.

Fig. 7 and Table 7 illustrate the effect of considering different transaction cost rates, ξ, on the performance of the proposed RTC-CNN-TD3 portfolio for the constituents of the S&P100 index. The portfolio return and market performance measures decreased with increasing ξ. Because a large transaction cost rate induces investors to reduce trading activities, the portfolio return declines. Moreover, when $\xi = 0.5\%$, the proposed learning-based portfolio had a significant loss compared with the scenario given by $\xi = 0.01\%$.

For completeness, Fig. 8 and Table 8 report the cumulative return and performance measures, respectively, for different risk aversion levels. As before, we observe a decrease in the portfolio return and volatility as β increases. There is an exception, with $\beta = 0.005$. For this level of investor risk aversion, we observe a good cumulative return performance and high Sharpe and CR values, suggesting that this level of intermediate risk aversion is optimal from an investment perspective.

5. Conclusion

This study proposes a DRL method to construct optimal portfolios that performs particularly well, even in high-dimensional

Table 3
Performance measures of different portfolio methods when $\beta = 0.01$ and $\xi = 0.1\%$.

Method	RTC-CNN-TD3	RTC-LSTM-TD3	RTC-CNN-DDPG	RTC-CNN-PPO	Max-Sharpe	MV
Annual Return (%)	16.96	17.40	9.87	8.77	2.12	2.73
Cum. Return (%)	33.69	34.62	19.05	16.86	3.97	5.11
Annual Volatility (%)	18.21	26.26	16.47	26.20	19.78	13.28
Sharpe Ratio	0.95	0.74	0.65	0.45	0.21	0.27
Max Drawdown (%)	19.82	26.84	19.19	25.64	21.20	15.87
Calmar Ratio	0.86	0.65	0.51	0.34	0.10	0.17

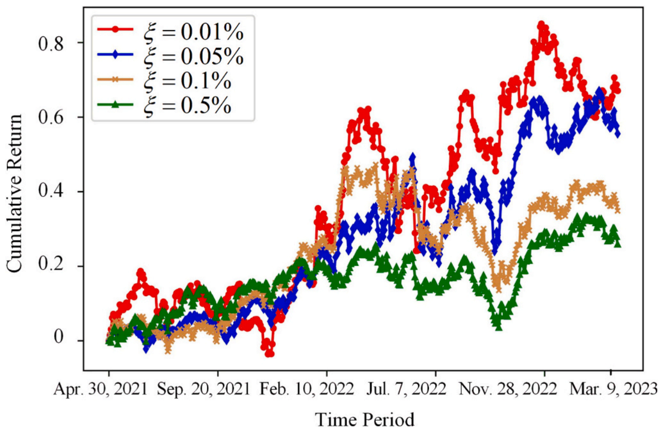

Fig. 4. Cumulative return performance of the RTC-CNN-TD3 portfolio under different transaction cost rates, ξ, for a risk aversion coefficient of β = 0.005.

Table 4
Performance measures of the RTC-CNN-TD3 portfolio under different transaction cost rates, ξ, when β = 0.005.

ξ	0.01%	0.05%	0.1%	0.5%
Annual Return (%)	31.90	26.91	17.51	13.23
Cum. Return (%)	67.06	55.52	34.86	25.89
Annual Volatility (%)	28.42	22.01	20.12	17.38
Sharpe Ratio	1.12	1.19	0.90	0.80
Max Drawdown (%)	23.45	19.11	22.88	17.59
Calmar Ratio	1.36	1.41	0.77	0.75

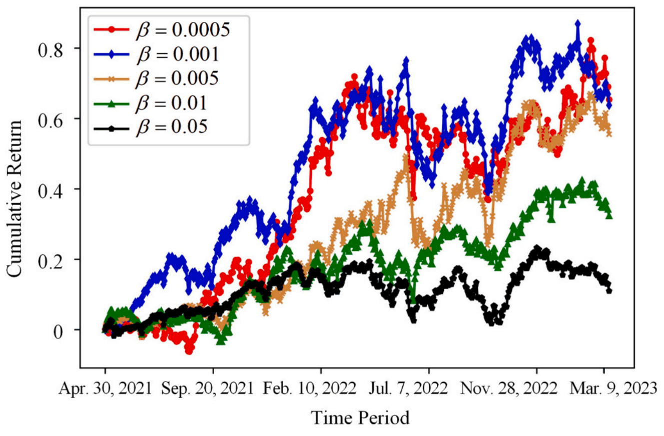

Fig. 5. Cumulative return performance of the RTC-CNN-TD3 portfolio under different risk aversion coefficients, β, for a transaction cost rate of ξ = 0.05%.

Table 5
Performance measures of the RTC-CNN-TD3 portfolio under different risk aversion coefficients, β, when ξ = 0.0005.

β	0.0005	0.001	0.005	0.01	0.05
Annual Return (%)	31.19	30.50	26.91	16.32	5.79
Cum. Return (%)	65.38	63.75	55.53	32.32	11.00
Annual Volatility (%)	27.06	24.52	22.01	20.26	15.91
Sharpe Ratio	1.14	1.21	1.19	0.85	0.43
Max Drawdown (%)	20.39	21.07	19.11	17.05	14.80
Calmar Ratio	1.53	1.45	1.41	0.96	0.39

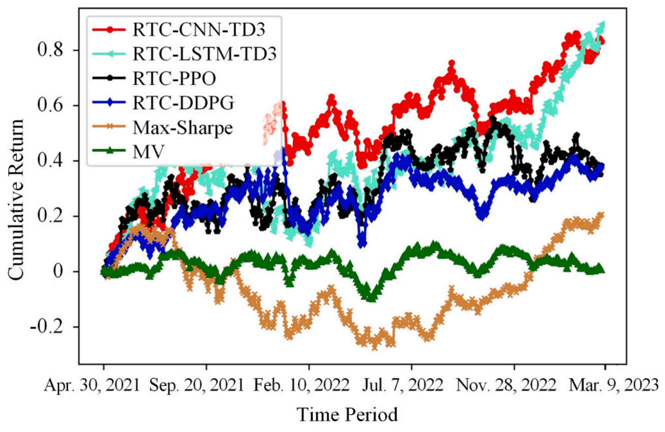

Fig. 6. Cumulative return performance comparisons using different portfolio trading strategies for a risk aversion coefficient of β = 0.005 and a transaction cost rate of ξ = 0.05%.

Table 6
Performance measures of different portfolio methods when β = 0.005 and ξ = 0.05%.

Method	RTC-CNN-TD3	RTC-LSTM-TD3	RTC-CNN-DDPG	RTC-CNN-PPO	Max-Sharpe	MV
Annual Return (%)	36.92	39.34	17.99	18.37	10.25	3.08
Cum. Return (%)	83.09	89.35	37.49	38.34	20.67	5.93
Annual Volatility (%)	24.79	26.10	22.92	31.72	28.72	14.73
Sharpe Ratio	1.39	1.40	0.84	0.69	0.48	0.09
Max Drawdown (%)	15.90	27.45	23.52	18.95	38.85	15.94
Calmar Ratio	2.32	1.43	0.76	0.97	0.27	0.02

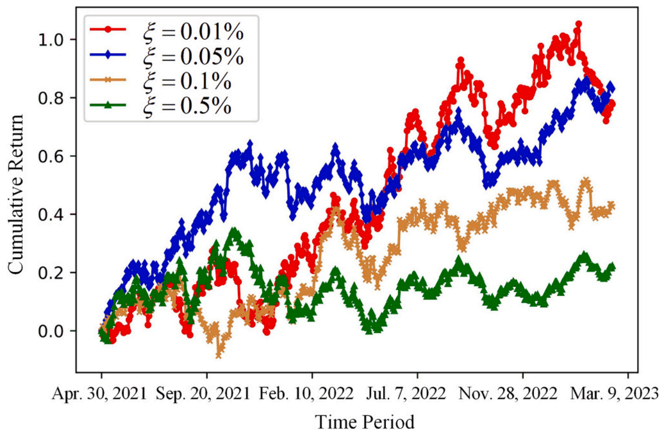

Fig. 7. Cumulative return performance of the RTC-CNN-TD3 portfolio under different transaction cost rates, ξ, for a risk aversion coefficient of β = 0.005.

Table 7
Performance measures of the RTC-CNN-TD3 portfolio under different transaction cost rates, ξ, when β = 0.005.

ξ	0.01%	0.05%	0.1%	0.5%
Annual Return (%)	34.85	36.92	20.23	11.03
Cum. Return (%)	77.78	83.09	42.57	22.30
Annual Volatility (%)	30.49	24.79	26.99	25.20
Sharpe Ratio	1.13	1.39	0.82	0.54
Max Drawdown (%)	22.42	15.90	23.90	25.62
Calmar Ratio	1.55	2.32	0.85	0.43

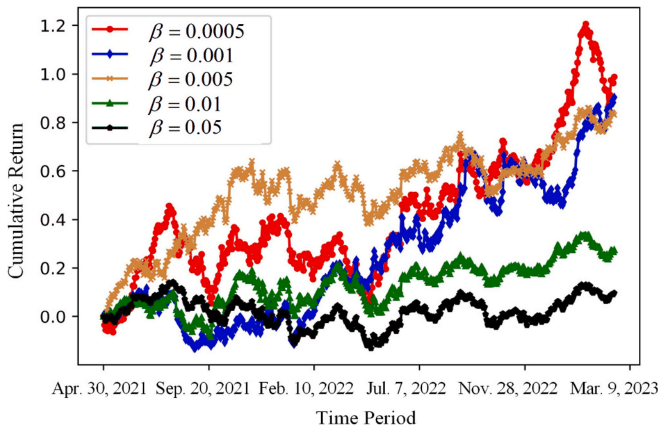

Fig. 8. Cumulative return performance of the RTC-CNN-TD3 portfolio under different values of the risk aversion coefficient, β, for ξ = 0.05%.

Table 8
Performance measures of the RTC-CNN-TD3 portfolio under different risk aversion coefficients, β, when ξ = 0.05%.

β	0.0005	0.001	0.005	0.01	0.05
Annual Return (%)	42.91	39.71	36.92	13.43	4.9
Cum. Return (%)	98.80	90.34	83.09	27.444	9.73
Annual Volatility (%)	33.62	29.97	24.79	21.83	20.26
Sharpe Ratio	1.22	1.27	1.39	0.69	0.34
Max Drawdown (%)	28.27	22.14	15.90	16.69	23.93
Calmar Ratio	1.51	1.793	2.32	0.80	0.21

settings. Our proposal combines DL and RL methods into a new DRL model that optimizes portfolio allocation. Investor risk aversion and transaction cost constraints are embedded using an extended Markowitz's mean-variance reward function, implemented using a TD3 algorithm.

We applied these strategies to the constituents of the DJIA and S&P100 and found extremely encouraging results. In particular, our proposed DRL method outperforms traditional investment strategies widely used by practitioners and recent models proposed in the deep reinforcement literature.

Author statement

Yifu Jiang: Conceptualization, Methodology, Software, Empirical Application, Data curation, Write up.
Jose Olmo: Literature review, Investigation, Supervision, Reviewing.
Majed Atwi: Literature review, Investigation, Supervision.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the writing of this study, the authors did not use any generative AI or AI-assisted technology. All research work was done independently by the authors to ensure originality and academic integrity of the research.

Declaration of competing interest

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.

Data availability

Data will be made available on request.

Acknowledgements

The authors acknowledge the financial support from Ministerio de Ciencia e Innovacion/Agencia Estatal de Investigación (Grant: PID2019-104326GB-I00), and from Fundación Agencia Aragonesa para la Investigación y el Desarrollo.

References

Aboussalah, A. M., & Lee, C. G. (2020). Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization. *Expert Systems with Applications, 140*, Article 112891.

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

Almahdi, S., & Yang, S. Y. (2017). An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems with Applications, 87*, 267–279.

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.

Almgren, R., & Chriss, N. (2001). Optimal execution of portfolio transactions. *Journal of Risk, 3*, 5–40.

Betancourt, C., & Chen, W. H. (2021). Deep reinforcement learning for portfolio management of markets with a dynamic number of assets. *Expert Systems with Applications, 164*, Article 114002.

Bühler, H., Gonon, L., Teichmann, J., & Wood, B. (2018). Deep hedging. *Quantitative Finance, 19*(8), 1271–1291.

Chambers, R. G., & Quiggin, J. (2007). Dual approaches to the analysis of risk aversion. *Economica, 74*(294), 189–213.

Chaouki, A., Hardiman, S., Schmidt, C., Sérié, E., & Lataillade, J. (2020). Deep deterministic portfolio optimization. *The Journal of Finance and Data Science, 6*, 16–30.

Choi, J. H., Larsen, K., & Seppi, D. J. (2019). Information and trading targets in a dynamic market equilibrium. *Journal of Financial Economics, 132*(3), 22–49.

Cui, T., Ding, S., Jin, H., & Zhang, Y. (2023). Portfolio constructions in cryptocurrency market: A CVaR-based deep reinforcement learning approach. *Economic Modelling, 119*, Article 106078.

Fereydooni, A., & Mahootchi, M. (2023). An algorithmic trading system based on a stacked generalization model and hidden Markov model in the foreign exchange market. *Global Finance Journal, 56*, Article 100825.

Fernandez-Arjona, L., & Filipovic, D. (2022). A machine learning approach to portfolio pricing and risk management for high-dimensional problems. *Journal of Mathematical Economics, 32*(4), 982–1019.

Gaivoronski, A. A., & Pflug, G. C. (2005). Value-at-risk in portfolio optimization: Properties and computational approach. *Journal of Risk, 7*(2), 1–31.

Garcia-Galicia, M., Carsteanu, A. A., & Clempner, J. B. (2019). Continuous-time reinforcement learning approach for portfolio management with time penalization. *Expert Systems with Applications, 129*, 27–36.

Goodell, J. W., Kumar, S., Lim, W. M., & Pattnaik, D. (2021). Artificial intelligence and machine learning in finance: Identifying foundations, themes, and research clusters from bibliometric analysis. *Journal of Behavioral and Experimental Finance, 32*, Article 100577.

Halperin, I. (2019). The QLBS Q-learner goes NuQLear: Fitted Q iteration, inverse RL, and option portfolios. *Quantitative Finance, 19*(9), 1543–1553.

Henriques, I., & Sadorsky, P. (2023). Forecasting NFT coin prices using machine learning: Insights into feature significance and portfolio strategies. *Global Finance Journal, 23*, Article 100904.

Huang, Z., & Tanaka, F. (2022). MSPM: A modularized and scalable multi-agent reinforcement learning-based system for financial portfolio management. *PLoS One, 17*(2), Article e0263689.

Jang, J., & Seong, N. Y. (2023). Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory. *Expert Systems with Applications, 218*, Article 119556.

Kircher, F., & Rsch, D. (2021). A shrinkage approach for sharpe ratio optimal portfolios with estimation risks. *Journal of Banking & Finance, 133*(7), Article 106281.

Li, B., Wang, J., Huang, D., & Hoi, S. C. (2018). Transaction cost optimization for online portfolio selection. *Quantitative Finance, 18*(8), 1411–1424.

Li, Y., Jiang, S., Wei, Y., & Wang, S. (2021). Take bitcoin into your portfolio: A novel ensemble portfolio optimization framework for broad commodity assets. *Financial Innovation, 7*(63). https://doi.org/10.1186/s40854-021-00281-x

Li, Z., Liu, X. Y., Zheng, J., Wang, Z., Walid, A., & Guo, J. (2021). FinRL-podracer: High performance and scalable deep reinforcement learning for quantitative finance. In *, 48. Proceedings of the Second ACM International Conference on AI in Finance. New York, NY, USA* (pp. 1–9).

Lillicrap, T. P., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., … Wierstra, D. (2015). Continuous control with deep reinforcement learning. In *International Conference on Learning Representations, San Diego, CA, USA, May 7–9*.

Lin, F., Wang, M., Liu, R., & Hong, Q. (2020). A DDPG algorithm for portfolio management. In *2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science, Xuzhou, China* (pp. 222–225).

Ma, G., Siu, C. C., & Zhu, S. P. (2019). Dynamic portfolio choice with return predictability and transaction costs. *European Journal of Operational Research, 278*(3), 976–988.

Markowitz, H. (1952). Portfolio selection. *The Journal of Finance, 7*(1), 77–91.

Mavruk, T. (2022). Analysis of herding behavior in individual investor portfolios using machine learning algorithms. *Research in International Business and Finance, 62*, Article 101740. https://doi.org/10.1016/j.ribaf.2022.101740

Moallemi, C. C., & Saglam, M. (2015). Dynamic portfolio choice with linear rebalancing rules. *Journal of Financial and Quantitative Analysis, 52*(3), 1247–1278.

Moody, J., & Saffell, M. (2001). Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks and Learning Systems, 12*(4), 875–889.

Moody, J., Wu, L., Liao, Y., & Saffell, M. (1998). Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting, 17*, 441–470.

Ngo, V. M., Nguyen, H. H., & Van Nguyen, P. (2023). Does reinforcement learning outperform deep learning and traditional portfolio optimization models in frontier and developed financial markets? *Research in International Business and Finance, 65*, Article 101936.

Park, H., Min, K. S., & Dong, G. C. (2020). An intelligent financial portfolio trading strategy using deep q-learning. *Expert Systems with Applications, 158*, Article 113573.

Park, S., Song, H., & Lee, S. (2019). Linear programing models for portfolio optimization using a benchmark. *European Journal of Finance, 25*(5), 435–457.

Peck, J., & Yang, H. (2011). Investment cycles, strategic delay, and self-reversing cascades. *International Economic Review, 52*(1), 259.

Pigorsch, U., & Schafer, S. (2022). High-dimensional stock portfolio trading with deep reinforcement learning. In *2022 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics (CIFEr)* (pp. 1–8). https://arxiv.org/abs/2112.04755.

Qureshi, F., Kutan, A. M., Ismail, I., & Gee, C. S. (2017). Mutual funds and stock market volatility: An empirical analysis of Asian emerging markets. *Emerging Markets Review, 31*, 176–192.

Roni, M., & Jean-Luc, V. (1996). Trading volume with private valuation: Evidence from the ex-dividend day. *Review of Financial Studies, 2*, 471–509.

Rubesam, A. (2022). Machine learning portfolios with equal risk contributions: Evidence from the Brazilian market. *Emerging Markets Review, 51*, Article 100891. https://doi.org/10.1016/j.ememar.2022.100891

Sebastian, O., Linan, D., & Jörg, R. (2021). Reinforcement learning about asset variability and correlation in repeated portfolio decisions. *Journal of Behavioral and Experimental Finance, 32*, Article 100559.

Shavandia, A., & Khedmati, M. (2022). A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets. *Expert Systems with Applications, 208*, Article 118124.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.

Ta, V. D., Liu, C. M., & Tadesse, D. A. (2020). Portfolio optimization-based stock prediction using long-short term memory network in quantitative trading. *Applied Sciences, 10*(2), 437.

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

Wang, M., & Ku, H. (2022). Risk-sensitive policies for portfolio management. *Expert Systems with Applications, 198*(15), 11680.

Wiering, M., & Otterlo, M. (2012). *Reinforcement learning: State of the art. Computational intelligence and complexity*. Springer.

Xu, W., & Dai, B. (2022). Delta-gamma-like hedging with transaction cost under reinforcement learning technique. *Journal of Derivatives, 29*(5), 60–82.

Yang, H., Liu, X. Y., Zhong, S., & Walid, A. (2020). Deep reinforcement learning for automated stock trading: An ensemble strategy. In *Proceedings of the First ACM International Conference on AI in Finance* (pp. 1–8).
Zeng, Y., & Klabjan, D. (2018). Portfolio optimization for American options. *Journal of Computational Finance, 22*(3), 37–64.
Zhang, J., & Maringer, D. (2016). Using a genetic algorithm to improve recurrent reinforcement learning for equity trading. *Computational Economics, 47*, 551–567.
Zhang, W. G., Zhang, X., & Chen, Y. (2011). Portfolio adjusting optimization with added assets and transaction costs based on credibility measures. *Insurance: Mathematics & Economics, 49*(3), 353–360.
Zhang, Y., Zhao, P., Wu, Q., Li, B., Huang, J., & Tan, M. (2022). Cost-sensitive portfolio selection via deep reinforcement learning. *IEEE Transactions on Knowledge and Data Engineering, 34*(1), 236–248.
Zhao, T. L., Ma, X., Li, X. M., & Zhang, C. M. (2023). Asset correlation based deep reinforcement learning for the portfolio selection. *Expert Systems with Applications, 221*, Article 119707.