Portfolio management using deep reinforcement learning

by Rui Wang, MSc

Student ID: 20396662

Submitted to The University of Nottingham
in September 2022
in partial fulfilment of the conditions for the award of the degree of
MSc Computer Science (Artificial Intelligence)

I declare that this dissertation is all my own work, except as indicated in the text

Contents

Contents ..... *ii*

Abstract ..... *iv*

List of Tables ..... *v*

List of Figures ..... *vi*

Acknowledgements ..... *vii*

Chapter 1: Introduction *1*

1.1 Background **1**
1.1.1 Investment & Portfolio Management 1
1.1.2 Portfolio Management using Algorithmic Trading 1

1.2 Traditional Investment Theories **1**
1.2.1 Modern Portfolio Theory (MPT) 1
1.2.2 Capital Asset Pricing Model (CAPM) 2
1.2.3 Efficient Market Hypothesis (EMH) 2

1.3 Portfolio Management using AI & Machine Learning **3**

Chapter 2: Literature Review *5*

Chapter 3: Problem Definition *7*

3.1 Reinforcement Framework **7**

3.2 Relationship Graph **7**

3.3 Portfolio Vectors **8**

Chapter 4: Methodology *9*

4.1 Top-level reinforcement learning framework **9**

4.2 Algorithm **9**

4.3 Network **10**
4.3.1 Structure I: Simple R-GCN 10
4.3.2 Structure II: LSTM-driven R-GCN 11
4.3.3 Structure III: CNN-driven R-GCN 11
4.3.4 Structure IV: CNN-driven Simple R-GCN 12
4.3.5 Structure V: CNN Stacked Simple R-GCN 12

Chapter 5: Experiment *14*

5.1 Datasets **14**

5.2 Comparison **15**

5.3 Metrics **15**
5.3.1 final Accumulative Portfolio Value (fAPV) 15
5.3.2 Annualized Rate of Return (ARR) 15
5.3.3 Sharpe Ratio (SR) 15
5.3.4 Maximum Draw Down (MDD) 15
5.3.5 Alpha (α) 16
5.3.6 Beta (β) 16

5.4 Training Hyperparameters & Environment **16**
5.4.1 Hyperparameters 16
5.4.2 Environment 17

Chapter 6: Result *18*
Chapter 7: Analysis *20*
7.1 Simple R-GCN and its Derivative Structures **20**
7.2 Analysing from all metrics **20**
7.3 Performance before Covid-19 ..... **21**
Chapter 8: Conclusion *23*
Chapter 9: Future Research *25*
9.1 Adjusting the Optimisation Algorithm **25**
9.2 Adjusting the Neural Networks **25**
9.2.1 Increase the number of network layers 25
9.2.2 Altering the network structure 25
9.3 Adjusting the Reward Function **25**
References *26*

Abstract

Reinforcement Learning as a branch of machine learning has made great developments in various fields in recent years. In particular, in the field of portfolio management, more and more attention is being paid to the reinforcement learning framework and it is believed that reinforcement learning has a highly potential. In this paper, we explore the structure and performance of Relational-Graph Convolutional Networks (R-GCN) in the context of deep reinforcement learning and compare it with common networks. Eventually, our proposed LSTM-driven R-GCN achieves a good performance in terms of trade-off between return and risk, and its return curve is fairly smooth.

List of Tables

Table 1 Overview of the Efficient Market Hypothesis 3
Table 2 Summary of the numerical information in the dataset 14
Table 3 Summary of time span information for the dataset 14
Table 4 Market types of input signals 15
Table 5 Key hyperparameters in experiments 16
Table 6 Hardware environment parameters used for the experiments 17
Table 7 Performance under all metrics across networks 18
Table 8 Performance under all metrics across networks after exclusion of the Covid-19 period 21

List of Figures

Fig.1 The relationship graph constructed using sector-industry relationships 7
Fig.2 Schematic diagram of the Simple R-GCN structure 11
Fig.3 Schematic diagram of the LSTM-driven R-GCN structure 11
Fig.4 Schematic diagram of the CNN-driven R-GCN structure 12
Fig.5 Schematic diagram of the CNN-driven Simple R-GCN structure 12
Fig.6 Schematic diagram of the CNN Stacked Simple R-GCN structure 13
Fig.7 Historical price curves for each asset 14
Fig.8 Performance of fAPV in the R-GCN group 18
Fig.9 Performance of fAPV in the R-GCN and control groups 19
Fig.10 fAPV curves for each network after exclusion of the Covid-19 period 22
Fig.11 The fAPV curves of LSTM-driven R-GCN and control groups 24
Fig.12 The fAPV curves of LSTM-driven R-GCN and controls excluding the covid-19 period 24

Acknowledgements

I would like to acknowledge my parents for encouraging and supporting me to pursue my MSc in Computer Science, as they got me to this place. I would also like to thank my supervisor for giving me advice on my thesis. I would like to thank my friends and classmates for making my journey less lonely. May we all have a bright prospect and good health in the future.

Chapter 1: Introduction

1.1 Background

1.1.1 Investment & Portfolio Management

An investment is a cession of the right to use the assets for a period of time thereafter to receive a future payment that is required to compensate the investor for the expected inflation rate over that period of time and the risk of uncertainty of future payments [1].

Asset classes comprise cash and cash equivalents, fixed income(bonds), equity(stocks), derivatives, commodities, real estate and other investments such as antiques, art and other collectables [2].

A primary progression within the investment field over the last decades has been the realization that you cannot simply aggregate a number of individual securities with ideal risk-return features to build an optimal portfolio. In other words, it has been demonstrated that investors are required to consider the inter-relationships between investments in order to build the best portfolio to meet their investment goals [1].

1.1.2 Portfolio Management using Algorithmic Trading

The process of algorithmic trading involves the execution of financial instruments on a computer using algorithms. Algorithms can be used to trade stocks, bonds, currencies, and a wide variety of financial derivatives. Investing strategies and trading goals also depend on algorithms. New trading techniques provide investors with better executions while lowering transaction costs, improving portfolio performance. Automated, black box, and robot trading are terms used to describe algorithmic trading [3]. It uses a variety of tools for analysis, including the Pre-Trade analysis, the Intraday analysis, and the post-Trade analysis, as well as rules-based trading and quantitative techniques.

The advantages of algorithmic trading are further demonstrated by high-frequency trading (HFT). With the aim of making a profit, it involves using sophisticated mathematical techniques and high-speed computers to trade stocks, bonds, or options. The difference between execution trading algorithms and HFT systems is that the former executes previously established investment decisions, while the latter makes both investment and trading decisions. Typically, HFT strategies fall into three types: Auto Market Making (AMM), Quant Trading/Statistical Arbitrage, and Rebate/Liquidity Trading. Each of them has a distinctive purpose and aim, even though there is some overlap between these types. In brief, high frequency trading is distinguished by these characteristics: Automated trading, No net investment, and Short trading horizons.

1.2 Traditional Investment Theories

1.2.1 Modern Portfolio Theory (MPT)

In 1952, Harry M. Markowitz spearheaded a mathematical introduction of the mean and variance for portfolio returns, defining explicitly and scientifically the preference of investors [4]. He further explained the principles of investment diversification in numerical terms and systematised the portfolio selection process, marking the commencement of MPT. The theory holds that portfolios are identified by the component securities and their weights, that portfolios can mitigate unsystematic risk from the markets and that the selection of uncorrelated securities should be the objective of portfolio construction.

$$r_p = \sum_{i=0}^{n} w_i \cdot r_i$$

Where r_p is the expected return of the portfolio, w_i and r_i are the weight and return of the *i*-th asset only, respectively.

$$\sigma_p^2 = \sum_{i=0}^{n} \sigma_i^2 \cdot w_i$$

Here σ_p^2 is the variance of the portfolio, σ_i^2 and w_i are the variance and weights of the *i*-th asset only, respectively.

1.2.2 Capital Asset Pricing Model (CAPM)

The Capital Asset Pricing Model (CAPM) was proposed by William Sharpe, John Lintner, Jack Treynor and Jan Mossin in 1964 [5]. The model was developed on the foundation of asset portfolio theory and capital market theory. Its main objective is to investigate the relationship between anticipated returns on assets and risky assets in the equity market, and how balance prices are generated. It is the mainstay in the theory of price in the modern financial markets. It is broadly applied to investment decision-making and corporate finance.

The capital asset pricing model presumes that all investors make decisions regarding their own investments in accordance with Markowitz's theory of asset selection, that they have the same estimation of expected returns, variance and covariance, and that they can borrow and lend liberally. Under such assumptions, research of capital asset pricing models concentrates on exploiting the quantitative relationship between returns and risks of risky assets. That is, what rate of returns an investor is expected to obtain as compensation for a particular level of risk.

The formulas for CPAM without α and with α are as follows.

$$E(r_p) = r_f + \beta_p(E(r_m) - r_f)$$

$$E(r_p) = r_f + \beta_p(E(r_m) - r_f) + \alpha$$

Where $E(r_p)$ refers to the expected return of the portfolio, $E(r_m)$ is the expected return of the market portfolio, r_f is the risk-free rate, β_p is the beta of the portfolio and α is the residual term indicating excess return.

1.2.3 Efficient Market Hypothesis (EMH)

The conception of market efficiency has been around since the early 1900s. However, the theory of the Efficient Market Hypothesis emerged in the mid-1960s and started to attract considerable interest. In 1970, Fama presented a paper containing comments on the theory and supporting arguments for the hypothesis. The review developed and refined the theory including definitions of three efficiency forms of financial market. The weak, semi-strong and strong forms of the efficient market hypothesis [6].

Weak-Form Efficient Market Hypothesis: The Weak-Form EMH assumes that current stock prices completely reflect the entire range of stock market information, including the historical series of prices, rate of return, transaction volume information and other information derived from the market. Therefore, we cannot use technical analysis to obtain excess returns.

Semi-Strong-Form Efficient Market Hypothesis: The semi-strong form of EMH believes that stock prices will adjust rapidly in response to all public information released and that the current stock price adequately reflects all public messages. It is noteworthy that public information encompasses all market and non-market messages such as company dividends and EBIT, P/E ratios, P/N ratios, stock splits and redemptions, share allotments, public economic information and political information, etc. With such assumptions, there is no possibility of achieving returns in excess of the market average through technical and fundamental analysis.

Strong-Form Efficient Market Hypothesis: The Strong-Form of EMH asserts that stock prices sufficiently reflect all information from public and private sources. This hypothesis postulates that no particular group of investors can defeat the market.

However, few markets can fully reach weak efficient markets, thus we can use the strong calculation power of computers to analyse market information and non-market information to obtain earnings that exceed the market benchmark.

Furthermore, there are behavioural finance, the random walk theory proposed by M. F. M Osborne and the beauty pageant theory proposed by John Maynard Keynes. However, all of them cannot be quantified and cannot provide assistance for quantitative investment.

Table 1 Overview of the Efficient Market Hypothesis

Effectiveness	Features
Weak-Form Efficient Market	All historical security price information is adequately reflected in market prices
Semi-Strong-Form Efficient Market	The prices completely reflect all publicly available information about the companies
Strong-Form Efficient Market	The prices adequately reflect all information about the companies' activities, both publicly available and internally undisclosed.

1.3 Portfolio Management using AI & Machine Learning

Along with the advancement of artificial intelligence (AI) and machine-learning technologies, a growing number of fields have started to combine with them, including the field of healthcare, security, forecasting, and portfolio management. Machine learning techniques have the advantages to process large amounts of data simultaneously and then making investment decisions, particularly in the area of high-frequency trading, where it can significantly increase the excess return of a portfolio, known as alpha.

Lee (2009) presented a support vector machine (SVM) based hybrid feature selection method called F_SSFS for forecasting equity trends. They compared this method with three commonly used feature selection methods and get an average cross-validation accuracy rate of 87.7% [7].

Kumar et al. (2006) used support vector machines and random forests to forecast the stock index movement of S&P, CNX, and NIFTY. [8]. Compared their performance with three other classification methods, hit ratios of Random Forests and Support Vector Machine archived 67.40% and 68.44, respectively.

Feng et al. (2017) select the determinant with the highest explaining effect on prospective returns among the vast number of return forecasting signals in the market using the LASSO regression [9]. Rapach et al. (2013) use the LASSO framework to search for lead-lag associations among asset groups or markets [10]. For instance, we can investigate which domestic sector or industry returns are the most important forecasters across all other sectors or industries. All of them yields promising results.

Chapados et al. (2001) trained an ANN to manage portfolios based on value-at-risk-adjusted profit criteria within a value-at-risk-control framework [11]. The forecasting and asset-allocation decisions performance of that neural networks and both significantly better than the market benchmark.

Becker et al. (2019) developed a deep learning method for the optimal stop-loss question, which learns optimal stop-loss rules directly from Monte Carlo samples [12]. The method was then tested on the pricing of Bermudan maximum call options and callable multiple barrier inverse convertibles, along with optimal stops for fractional Brownian motion. Across all three scenarios, it yielded remarkably accuracy in the high-dimensional context, with short computation times.

Machine learning methodologies have received considerable attention from academics in finance. Bartram et al. (2021) investigate the most prominent ML approaches and empirical outcomes in the literature on active portfolio management. ML has applications of asset management in signal generating, portfolio structuring and trade executing, with promising results have already been

reported. And they point out that reinforcement learning (RL), especially, is promised to play a further major impact in the financial sector [13].

In this paper, we will introduce the Relational-Graph Convolution Network (R-GCN) within the Deep Reinforcement Learning (DRL) framework to manage portfolios, and propose 5 structures of R-GCN. Then compare their performance with Convolution Neural Network (CNN), simple Recurrent Neural Network (sRNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU) under the DRL framework.

Chapter 2: Literature Review

Almahdi et al. (2017) present a recurrent reinforcement learning method that employs a coherent risk-adjusted performance objective function, the Kalmar ratio, to derive buy and sell signals and weightings of portfolio allocations. With a portfolio comprising the most frequently traded exchange-traded funds, they revealed that compared to previously proposed RRL objective functions (i.e., Sharpe ratio and Stirling ratio), an objective function based on the expected maximum downside risk yielded better return performance, with the variable-weighted RRL long/short portfolio outperforming the equal-weighted RRL long/short investment for different transaction costs [14].

Jiang et al. (2017) present a financial-model-free Reinforcement Learning framework to provide a deep machine learning solution to the portfolio management problem [15]. Their framework contained four nova points: Ensemble of Identical Independent Evaluators (EIIE) topology, Portfolio-Vector Memory (PVM), Online Stochastic Batch Learning (OSBL) scheme, and a fully exploiting and explicit reward function. They realized it using three neural networks, including a Convolutional Neural Network (CNN), a basic Recurrent Neural Network (RNN), and a Long Short-Term Memory (LSTM). Their experiment is based on thirty minutes of trading data from the cryptocurrency market for high-frequency trading. Even with a trading cost of 0.25%, the framework yielded at least quadruple the returns in 50 days.

Deng et al. (2017) proposed a financial trading system combining Fuzzy Learning, recurrent deep neural network (RDNN) and Reinforcement Learning (RL), called Fuzzy deep direct reinforcement (FDDN) [16]. They employed fuzzy learning to reduce the uncertainty of the data and use DNN to perform noise reduction and feature extraction on the data. The processed data is then handed over to the RL module to select portfolios. They tested on CSI 300 futures trading data and high-frequency data of commodity futures for silver and sugar at minute levels respectively. The experimental results show that FDDR is exceptionally profitable.

Lu (2017) used LSTM for feature extraction and incorporates market downside signals in the framework of reinforcement learning, thereby solving the problem of gradient disappearance that often occurs when using DNN for feature extraction. A Dropout layer is also added to avoid over-fitting of the LSTM. In addition, he also tried to use the falling deviation ratio instead of the Sharpe ratio as the loss function [17].

Yao et al. (2022) built on the Inception network and the Bottleneck attention module to implement a novel network structure embedded in a deep reinforcement learning framework [18]. They additionally applied an ensemble framework of identical independent evaluators to realize two filter maps as the identical separate evaluators to optimize the best network parameters. They selected 11 underlying asset origins in the cryptocurrency market to evaluate the validity of their proposed investment policy compared to eight other investment strategies, using accumulated returns and Sharpe ratios as metrics to evaluate the performance of the policy. The back-testing results indicated that their algorithm could achieve a return of at least 98.7% over a 50-day time horizon.

Liang et al. (2018) realized three reinforcement algorithms, including Deep Deterministic Policy Gradient (DDPG), Proximal Policy Optimization (PPO), and Policy Gradient (PG), for portfolio management and compared their performance with ADR, Sharpe, and MMD. Their conclusion is that DDPG and PPO are not better than PG in the field of financial investment, although the formers are updated to some extent. So they proposed Adversarial Policy Gradient to improve the performance of PG [19].

Jiang et al. (2017) present a model-less convolutional neural network under the reinforcement learning framework, using the accumulative return as the reward function [20]. They trained it with around 8.5 months' price data from a cryptocurrency exchange and backtested it in 1.8 months using 30-minute scale transaction data. Their result achieved 10-fold returns and was significantly better than other neural networks. They also stated that this neural network could also be used for high-frequency trading in other financial markets.

Feng et al. (2019) proposed a graph-based deep learning network, called Relational Stock Ranking (RSR), for stock forecasting [21]. The neural network they constructed consists of an input layer, a sequential embedding (LSTM) layer, a relational embedding (Temporal Graph Convolution) layer and a prediction (Fully Connected) layer in that order. Their framework was trained and validated using historical data from NASDAQ and NYSE. Its performance, measured with the average return ratio, reached 71% and 98% on NASDAQ and NYSE, respectively.

With the aim of harnessing the temporal variation of interrelationships between financial instruments, Soleymani et al. (2021) presented a graph convolutional reinforcement learning framework, called DeepPocket [22]. It utilizes an autoencoder to extract features, a CNN for mining information underneath, and an agent critic structure. The agent was primarily trained offline on historical data using online random batch processing. With the availability of incoming data, an online training was carried out using a passive concept drift methodology to address unforeseen variations in its distribution. The authors compared its performance with four market indices, including the Dow Jones Industrial (DJI), Euro Stoxx 50, NASDAQ, and S&P500, over three 30-day periods, covering the Covid-19 crisis. The result was that DeepPocket significantly outperformed the market indices.

Shi et al. (2022) introduced a reinforcement learning framework based on graph convolutional networks used in portfolio management, named GPM [23]. It commences by using a relational graph convolutional network (R-GCN) to extract relational features and incorporates multi-scale temporal features to make decisions. Their experiments were carried out using NASDAQ and NYSE data and by adjusting the number of network layers and other hyperparameters to obtain the best results. At last, compared with other state-of-art portfolio management methods, the result is that their proposed GPM network outperforms other networks when using both fAPV and SR as metrics, but its MDD performance is not optimal.

Chapter 3: Problem Definition

3.1 Reinforcement Framework

Portfolio management is a decision-making process that continuously allocates capital to different assets. In the deep reinforcement learning framework, a trading agent gives the allocation weights of assets in a given transaction environment (containing all the assets that can be allocated). The initially available cash for the agent is set at 1, i.e., all cash. By allocating all capital in equal amounts to all investable assets, this passive investment approach provides a fairly representative return of the average market return. Hence we use it as the market benchmark.

3.2 Relationship Graph

The relationship between allocatable assets is represented by a constant relationship graph G(A, E, R). The relationship graph contains three variables.

1. Asset Nodes (A): each of which represents a selectable investment underlying;

2. Edges between nodes (E): where the presence of an edge between two nodes indicates that the two selectable assets are related and the length of the edge represents their similarity;

3. Relationships (R): the relationship between two nodes in the R-GCN that will affect their correlation.

In the classical GCN model [24], the similarity of different nodes is calculated by feeding each node's degree (adjacent nodes) matrix to cluster them. However, in the R-GCN model, the relationship between two nodes is also a consideration before clustering. For example, the relationship between Company A and Company B is very similar to other companies in the market. In GCN, these two companies should be classified in one category, and then holding them concurrently would not reduce the portfolio's unsystematic risk when managing the portfolio. However, in R-GCN, if they are not related to each other or are not in the same industry (perhaps they are upstream and downstream in the industry chain or in a partnership), then it makes sense to hold them in the portfolio at the same time.

In this paper, we use a three-dimensional sparse tensor to represent the relationship graph, which has a shape of (number of assets, number of assets, number of relationships). The first two dimensions of this tensor are stock information and the third dimension is relationship information. If stocks A and B are in the *i*-th sector, then the values of [A, B, *i*] and [B, A, *i*] are 1, otherwise they are 0.

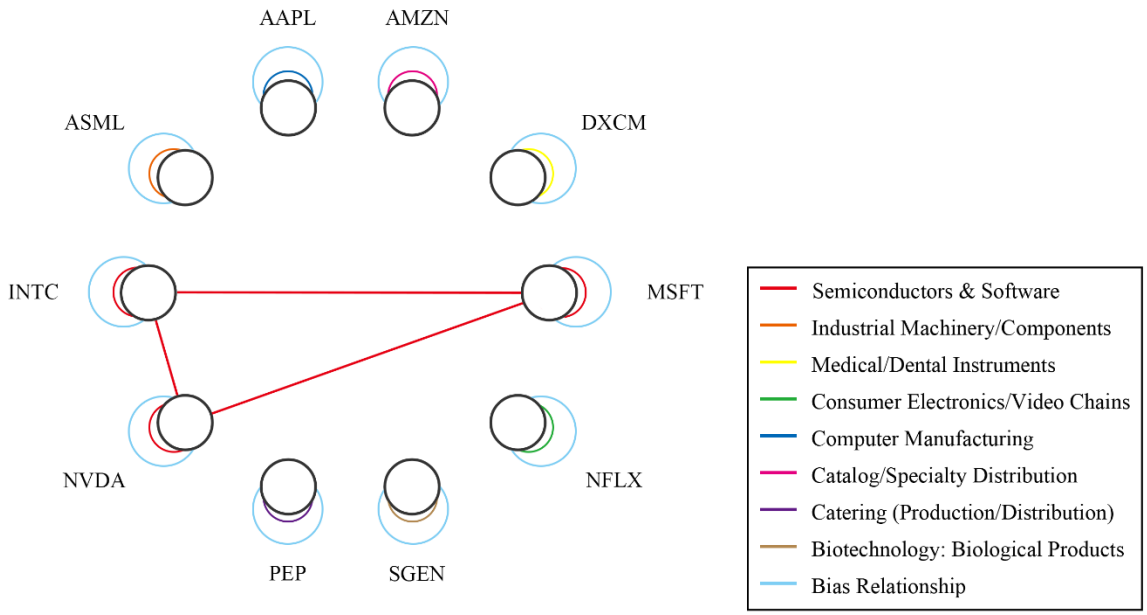

Fig.1 The relationship graph constructed using sector-industry relationships

In our experiments, we build a relationship graph for the portfolio based on sector-industry relationships, the structure of which can be illustrated using a non-directional graph as shown in Figure 1. Each node represents an investment target (company). If there is an edge between the nodes it means that there is a relationship between the two companies, and different coloured edges represent different types of relationships.

3.3 Portfolio Vectors

In the portfolio problem, the agent is required to give the proportion of each asset allocation at each trading point, including cash. So we can use a vector of weights to represent the portfolio. The portfolio for trading day t can then be represented as,

$$w_t = [w_{0,t}, w_{1,t}, w_{0,t}, \dots, w_{10,t}]$$

On any trading day t, the sum of all weights is one, i.e.

$$\sum_{i=0}^{10} w_{i,t} = 1$$

The weights vector is initialized as

$$w_0 = [1,0,0, \dots,0]$$

And shorting is not allowed in our framework, so all weights are greater than zero and less than one, that is $0 \leq w_{i,t} \leq 1$.

Meanwhile, the relative price of any trading day t can be represented by the vector V_t as

$$V_t = [1, \frac{v_{1,t}}{v_{1,t-1}}, \frac{v_{2,t}}{v_{2,t-1}}, \dots, \frac{v_{10,t}}{v_{10,t-1}}]$$

Then the total value of the portfolio at time T, taking into account transaction frictions, is

$$p_t = p_{t-1} \cdot v_t \cdot w_t - \mu \cdot p_{t-1} \cdot \Delta w^T$$

where μ is the transaction cost, which is set as a constant of 0.25‰, i.e., 0.00025, in this experiment

To generate the ultimate accrued profit, the return for period *t* is converted to a logarithmic rate of return, r_t, which is formulated as,

$$r_t = \ln \frac{p_t}{p_{t-1}}$$

Thereafter, the resulting portfolio value can be derived as,

$$p_t = p_0 \cdot e^{\Sigma_1^T r_t}$$

Building on the above statement, the purpose of our portfolio management is to locate a series of weights that would maximize the ultimate value of the portfolio p_T.

Chapter 4: Methodology

4.1 Top-level reinforcement learning framework

The portfolio management problem can be modelled as a Markov Decision Process (MDP) [25] in the form of a 4-tuple (S, A, P_a, R_a), where S is the set of states (s) named state space, A is a set of actions (a) called the action space, $P_a: S \times A \times S \to [0,1]$ is the transition probability distribution, and $R_a: S \times A \to R$ is the reward function. Further details are provided below:

- ***State***: When placing a transaction in period t, the agent will take into account the current signal vector of the market Sg_t, the asset relation graph G, and the previous portfolio vector w_{t-1}. So

$$S_t = A(Sg_t, G, w_{t-1})$$

  where Sg_t denotes the input signal vector of each asset at period t,

$$Sg_t = [MA10_t, MA30_t, MA100_t, Volume_t, Mkt\ Cap_t, P/E_t, LIBOR_t,$$

$$COMEX\ Gold_t, COMEX\ Copper_t, NYMEX\ WTI_t, CBOT\ 10yr\ T-Bonds_t]$$

  G is the constant matrix representing the relational graph, and w_{t-1} represents the weight vector at the previous moment $(t-1)$.

- ***Action***: The action is to establish the vector of portfolio w_t, i.e., $a_t = w_t$.
- ***Transition***: Our agent uses the DDPG algorithm to perform the decision action, i.e., to transfer states. Details will be elaborated in the next sub-section.
- ***Reward***: The reward $R_a(s, s')$ at period t is the trading return r_t. In this paper, the reward function is defined by the following formula:

$$Reward_t = \ln\left(\frac{r_t}{r_{t-1}}\right)$$

4.2 Algorithm

In this paper, our agent uses Deep Deterministic Policy Gradient (DDPG) [26] to give the next portfolio weight.

DDPG is a method that can solve the problem of continuity control. It is mod-free, off-policy, and policy-based, which can ultimately output only one action with certainty. DDPG is structured in the form of Actor-Critic, so it is divided into two large networks, the Strategy Network and the Value Network. It inherits the idea of a fixed target network from DQN and each network is subdivided into a local network and a target network. These four networks in each agent will use the same network and their structure will be given in the following subsection.

The policy network, or Actor, takes a deterministic strategy. It outputs a deterministic action directly through the local network. It is updated based on gradient ascent because the goal of Actor is to find an action a that maximises the output value Q. So the gradient of the optimisation strategy network is to maximise this Q value of the value network output.

The value network, or Critic, is used to fit the value function. Both of its networks returns the q-value of the current state. However, two arguments are given as input to the target network, respectively the observation of the current state and the action output of the actor's target network. The input to the local network, on the other hand, is the action output of the actor's local network. The target network is designed to figure out the Q-Target and subsequently the TD-error of the value network since the update of the value network is based on gradient descent of the TD-error.

Notably, DDQG also draws on DQN's Experience Replay technique. In other words, the sequences are stored in the experience pool for a period of time, and a minibatch is randomly sampled from the experience pool for each training session.

The agent makes decisions according to the actor $\mu(s|\theta^\mu)$ and critic $Q(s, a|\theta^Q)$ with weights θ^μ and θ^Q. The goal of the Actor is to optimize the θ^μ to make the action with the highest Q value, the

goal of the Critic is to optimize the θ^Q to give a more accurate Q value. The detailed algorithm is shown below.

Algorithm 1 DDPG algorithm [26]

Randomly initialize local actor network $\mu(s|\theta^\mu)$ and critic network $Q(s,a|\theta^Q)$ with weights θ^Q and θ^μ.

Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$

Initialize replay buffer *R*

for episode = 1, **M do**

Initialize a stochastic noise n for action exploring

Accept original observation state s_1

for t = 1, **T do**

Choose an action $a_t = \mu(s_t|\theta^\mu) + n$ based on present policy and exploratory noise

Implement action a_t, observe reward r_t and new state s_{t+1}

Store transition (s_t, a_t, r_t, s_{t+1}) in *R*

Sample a random minibatch of *N* transitions (s_i, a_i, r_i, s_{i+1}) from *R*

Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$

Update critic by minimizing the loss:

$$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - Q(s_i, a_i|\theta^Q))^2$$

Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_a Q(s,a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|s_i$$

Update the target networks:

$$\theta^{\mu'} \leftarrow \tau \cdot \theta^\mu + (1-\tau)\cdot\theta^{\mu'}$$

$$\theta^{Q'} \leftarrow \tau \cdot \theta^Q + (1-\tau)\cdot\theta^{Q'}$$

end for

end for

4.3 Network

In this section, we present five structures of the R-GCN. And later on, we will compare their performance.

4.3.1 Structure I: Simple R-GCN

Referring to the TGC in [21], we feed the input to the LSTM layer, then pass it through the R-GCN layer and finally get the portfolio weight vector from a fully connected layer. We call this structure the Basic R-GCN. Fig.1 presents a schematic diagram of the structure of the Simple R-GCN.

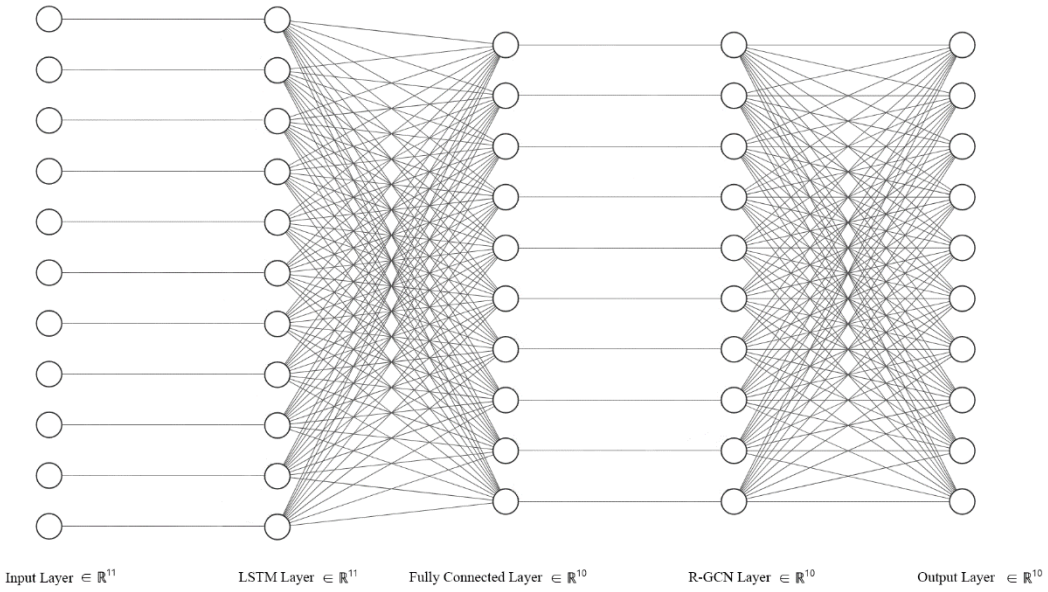

Fig.2 Schematic diagram of the Simple R-GCN structure

4.3.2 Structure II: LSTM-driven R-GCN

In this structure, the input to the fully connected layer is no longer the output of the R-GCN layer, but a summation of the outputs of the LSTM and R-GCN layers, that's why we call it LSTM-driven. Fig.2 presents a schematic diagram of the structure of the LSTM-driven R-GCN.

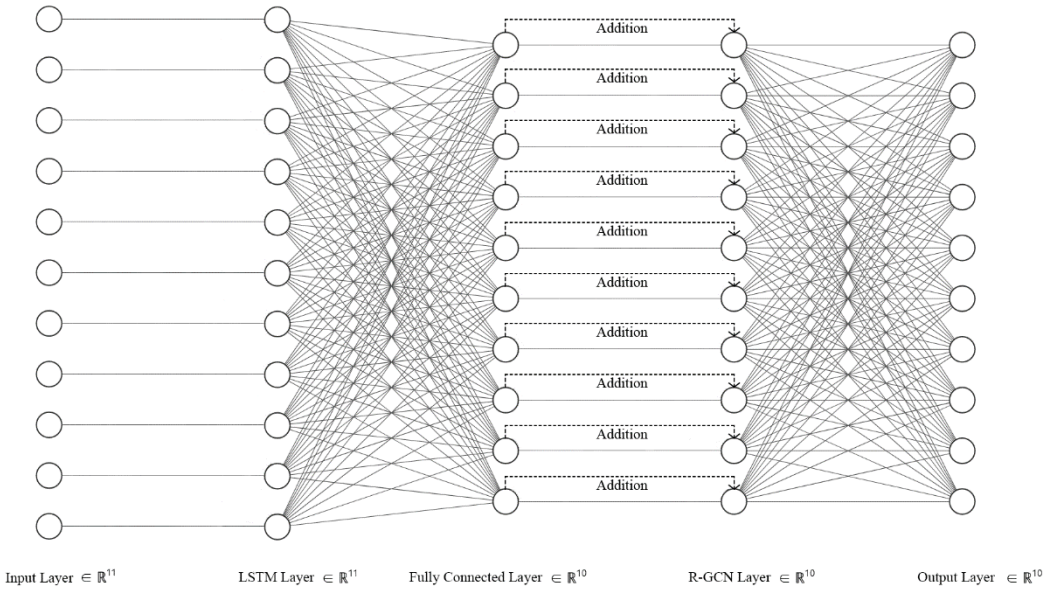

Fig.3 Schematic diagram of the LSTM-driven R-GCN structure

4.3.3 Structure III: CNN-driven R-GCN

This structure no longer uses the LSTM to extract features, but a CNN instead. That is, all the LSTM layers in Structure II are replaced with CNN layers. Fig.3 presents a schematic diagram of the structure of the CNN-driven R-GCN.

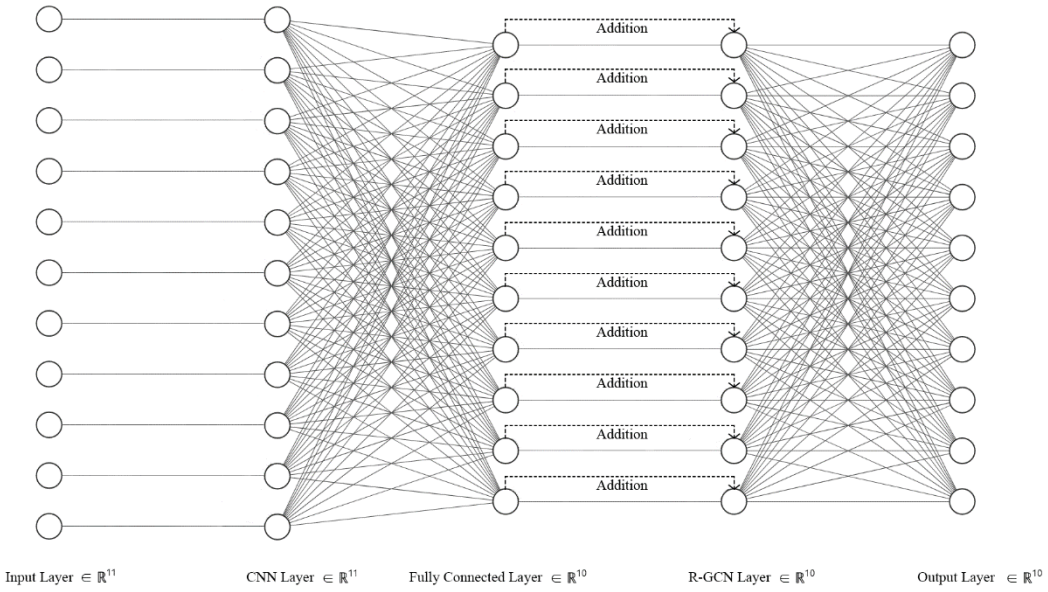

Fig.4 Schematic diagram of the CNN-driven R-GCN structure

4.3.4 Structure IV: CNN-driven Simple R-GCN

In this structure, the data are fed into the LSTM layer and the CNN layer respectively. The output of the LSTM is then fed into the R-GCN layer. Its output is summed with the output of the CNN layer and sent to the fully connected layer to obtain the weight vector. Fig.4 presents a schematic diagram of the structure of the CNN-driven Simple R-GCN.

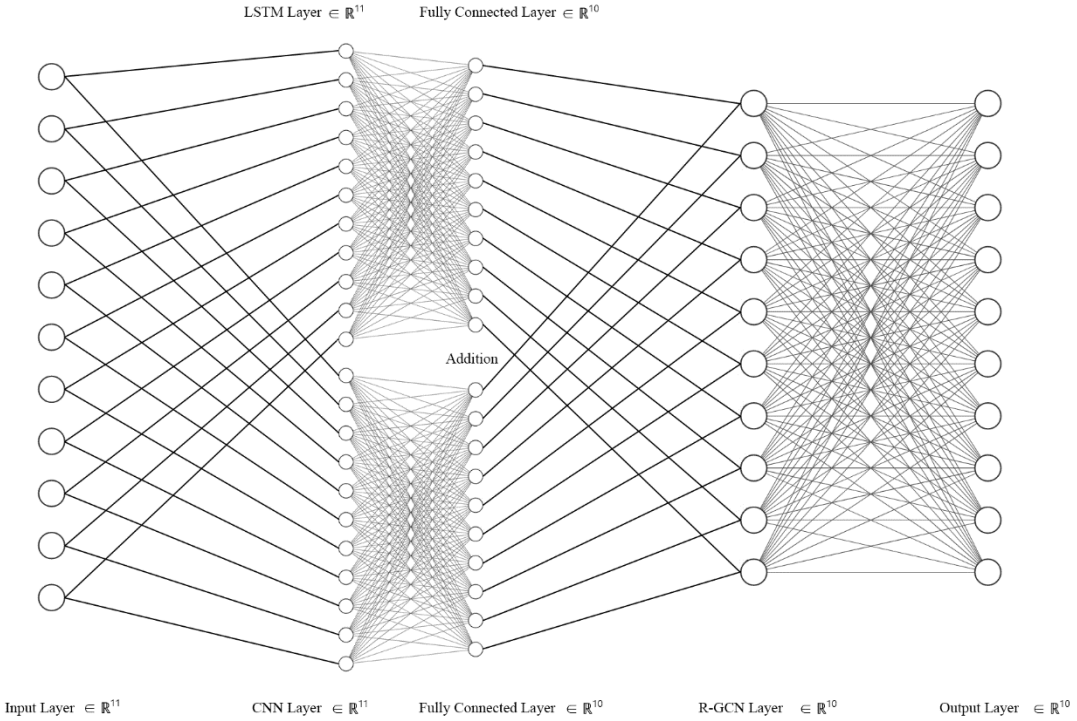

Fig.5 Schematic diagram of the CNN-driven Simple R-GCN structure

4.3.5 Structure V: CNN Stacked Simple R-GCN

This structure consists of a CNN stacked in front of a LSTM-driven R-GCN. Figure 5 presents a schematic diagram of the structure of the CNN Stacked Simple R-GCN.

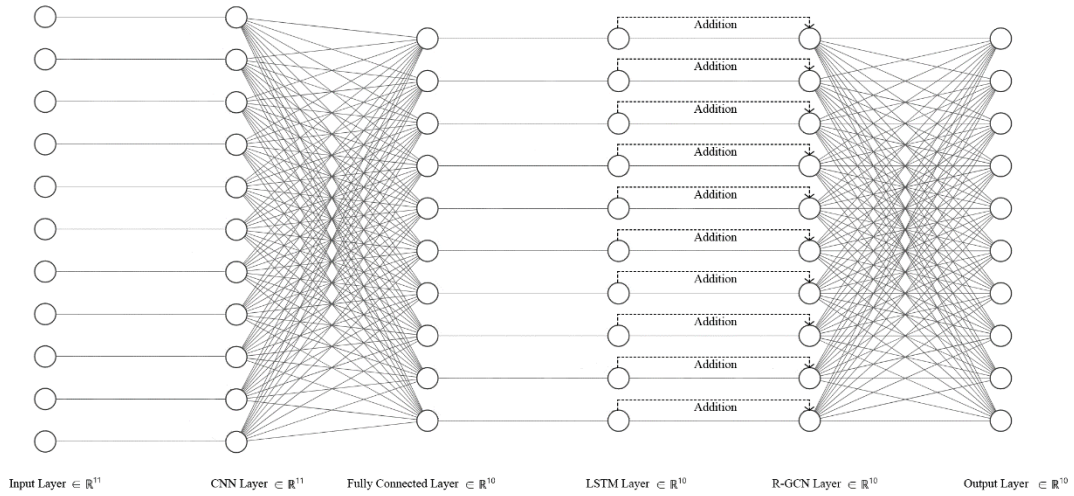

Fig.6 Schematic diagram of the CNN Stacked Simple R-GCN structure

The activation functions used in this paper are all ReLu functions including not only the five R-GCN structures, but also the four structures of the control group. Moreover, with reference to the EIIE structure of [15], we use a separate network structure for each asset.

Chapter 5: Experiment

5.1 Datasets

We train and back-testing our agents on the NASDAQ market. The historical price and signal data used is derived from EMQuantAPI [1] and the relationship data comes from [2] [21]. All the information above is summarised in Table 1. The relational data used in this paper are sector-industry data. In [23], wiki data on inter-company relationships are also utilised. However, the portfolio constructed in this paper contains ten stocks, and the matrix of company relationships (e.g. sub-parent relationships, etc.) between them would be sparse, so we only applied sector-industry relationships in this paper. The historical data spans from 2008-05-23 to 2022-08-23. We use the data from the first 2000 trading days for offline learning and then back test from the 300th trading day for online learning. For the stock signals, we have selected 11 indicators, namely, MA10 (10-day average moving), MA30 (30-day average moving), MA100 (100-day average moving), Volume, Mkt Cap (market capitalization), P/E ratio, LIBOR (London Interbank Offered Rate), COMEX Gold, COMEX Copper, NYMEX WTI and CBOT 10-year Treasury Bonds. These signals include key indicators in money markets, bond markets, capital markets, futures markets and commodity markets, providing agents with clues to the world's financial markets and guiding them in making portfolio decisions. Figs. 5 shows the trend in the price of each asset.

Table 2 Summary of the numerical information in the dataset

Market	Stocks Number	Sector-Industry Relations	
		Relations Number	Edges Number
NASDAQ	10	10	23

Table 3 Summary of time span information for the dataset

Periods	Time Span	Number of Trading Days
Offline Learning	2008-05-23 to 2016-05-03	2000
Online Learning	2009-07-31 to 2022-08-23	3289

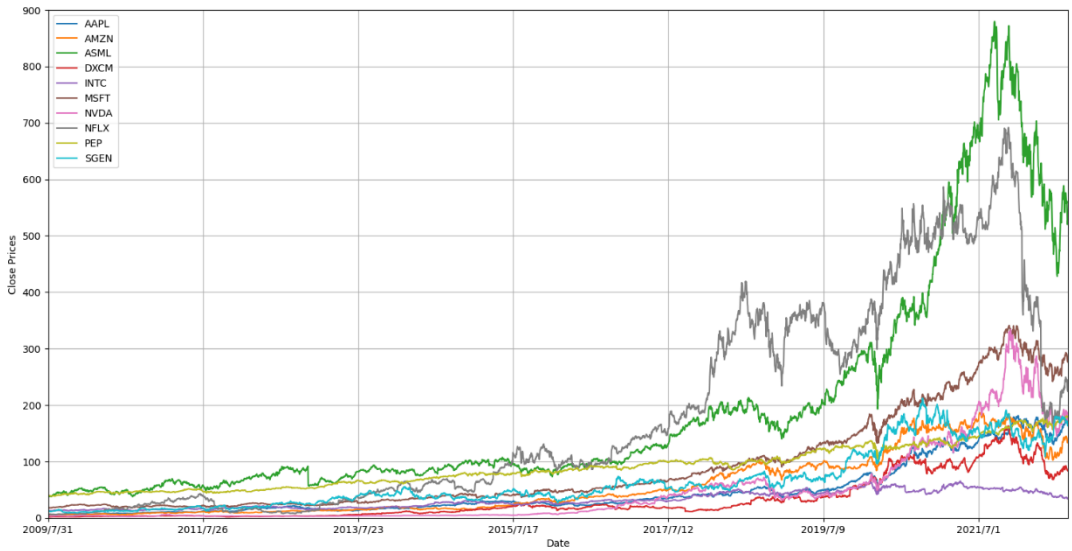

Fig.7 Historical price curves for each asset

[1] https://quantapi.eastmoney.com/

[2] https://github.com/hennande/Temporal_Relational_Stock_Ranking.

Table 4 Market types of input signals

Signals	Money Market	Bond Market	Capital Market	Futures Market	Commodity Market
MA10			⊗		
MA30			⊗		
MA100			⊗		
Volume			⊗		
Mkt Cap					
P/E ratio					
LIBOR	⊗				
COMEX Gold				⊗	⊗
COMEX Copper				⊗	⊗
NYMEX WTI				⊗	⊗
CBOT 10yr T-Bonds		⊗		⊗	

5.2 Comparison

To measure the performance of the different networks, we compare the results of four different structures of R-GCN. Meanwhile, the R-GCN with the best results is compared with CNN, sRNN, LSTM, and GRU.

5.3 Metrics

The performance of the above neural networks was assessed in terms of the undermentioned six metrics.

5.3.1 final Accumulative Portfolio Value (fAPV)

It is a very intuitive and straightforward metric that captures how much our assets have improved at the end of the investment period compared to the beginning of the term, which is similar to the holding period return.

$$\text{fAPV} = \frac{p_T}{p_0}$$

5.3.2 Annualized Rate of Return (ARR)

It is a way of calculating investment returns on an annual basis. It is calculated using the formula,

$$ARR = (\frac{p_T}{p_0})^{\frac{1}{N}} - 1$$

5.3.3 Sharpe Ratio (SR)

It is used to measure the rate of return that can be earned per unit of risk taken.

$$\text{SR} = \frac{E(r_P) - r_f}{\sigma_P}$$

as holding cash earns no return in our framework, the risk-free rate of return (r_f) here is zero.

5.3.4 Maximum Draw Down (MDD)

It is the maximum observed loss from the peak to the trough of the portfolio before a new peak is reached.

$$\text{MDD} = \max_{\tau \epsilon (0,T)} \left(\max_{t \epsilon (0,\tau)} \frac{p_t - p_\tau}{p_t} \right)$$

5.3.5 Alpha (α)

It is calculated by subtracting expected returns from actual returns and represents the excess return, which is independent of market volatility. According to the CAPM model with alpha,

$$r_p = r_f + \beta\left(r_m - r_f\right) + \alpha$$

we can deduce that

$$\alpha = r_p - \left[r_f + \beta\left(r_m - r_f\right)\right] = r_p - E(r_p)$$

As the Alpha measures returns, it is necessary to calculate the annualized Alpha, which is given by,

$$\text{Annualised Alpha} = (1 + \alpha_T)^{\frac{1}{N}} - 1$$

where N represents the number of years within the holding period.

$$N = \frac{the\ total\ number\ of\ trading\ days}{252}$$

252 is the number of trading days in a year.

5.3.6 Beta (β)

It measures the correlation of a stock or portfolio with the overall market.

$$\beta = \frac{Cov(r_p, r_m)}{\sigma_{r_m}}$$

These five metric measures reflect the performance of the network in different ways.
fAPV evaluates the performance of a portfolio purely in terms of returns. The higher fPAV, the higher returns and the better the performance.

The SR takes into account not only returns but also risks. The higher the Sharpe ratio, the higher the return for the same risk. It is also the higher the better.

The MDD is an evaluation of a portfolio from a risk perspective; a larger MDD indicates a more volatile portfolio. Investors are risk averse, so the smaller it is, the better, indicating a stable portfolio performance.

Alpha is calculated as excess return and obviously the higher the Alpha, the better the portfolio's active investment strategy. A positive alpha indicates that the portfolio has a superior active investment strategy, zero shows that the active investment has the same return as a passive investment that follows the market, and less than zero denotes that the strategy is ineffective.

Beta reflects the sensitivity of a security's or portfolio's yield performance to changes in the average level of market returns. In a bull market, with little difference in valuation advantage, investors will choose stocks or portfolios with a large beta in the hope of achieving higher returns. Conversely, in a bear market, investors will instead choose stocks or portfolios with small beta coefficients to reduce losses and volatility.

5.4 Training Hyperparameters & Environment

5.4.1 Hyperparameters

Table 5 Key hyperparameters in experiments

	Actor Learning Rate	Critic Learning Rate	Mini batch size	Discount factor (Gamma)	Soft Update of Target Parameter (Tau)
Values	0.00056	0.0059	256	0.93	0.0083

5.4.2 Environment

The experiments in this paper were carried out on the following hardware environment.

Table 6 Hardware environment parameters used for the experiments

CPU	GPU	RAM	OS
Intel i5 10500	1080Ti with 11Gb Memory	32Gb with 2666Mhz	Windows 10

Chapter 6: Result

All the results of the experiments are summarized in Table 7.

Table 7 Performance under all metrics across networks

Metrics	fAPV	ARR	SR	MDD	Alpha (annualized)	Beta (averaged)
Simple R-GCN	26.3026	0.2849	1.2855	**0.3708**	0.1181	1.1491
LSTM-driven R-GCN	**47.3653**	**0.3442**	**1.3912**	0.4445	**0.2814**	**1.5748**
CNN-driven R-GCN	24.5992	0.2783	1.1548	0.3951	0.0755	1.0906
CNN-driven Simple R-GCN	36.2768	0.3170	**1.3353**	0.4367	0.2261	1.4296
CNN Stacked Simple R-GCN	46.5526	**0.3424**	1.2773	0.4413	**0.2872**	**1.5224**
CNN	**57.1339**	**0.3636**	1.2168	0.5240	**0.3138**	**2.1905**
sRNN	46.8406	0.3430	1.2699	0.4779	0.2793	1.7366
LSTM	33.2460	0.3082	1.0967	0.4595	0.2038	1.3729
GRU	29.8543	0.2974	1.0907	0.4549	0.1711	1.2091
Market	23.0147	0.2718	**1.2998**	**0.3541**	0	1

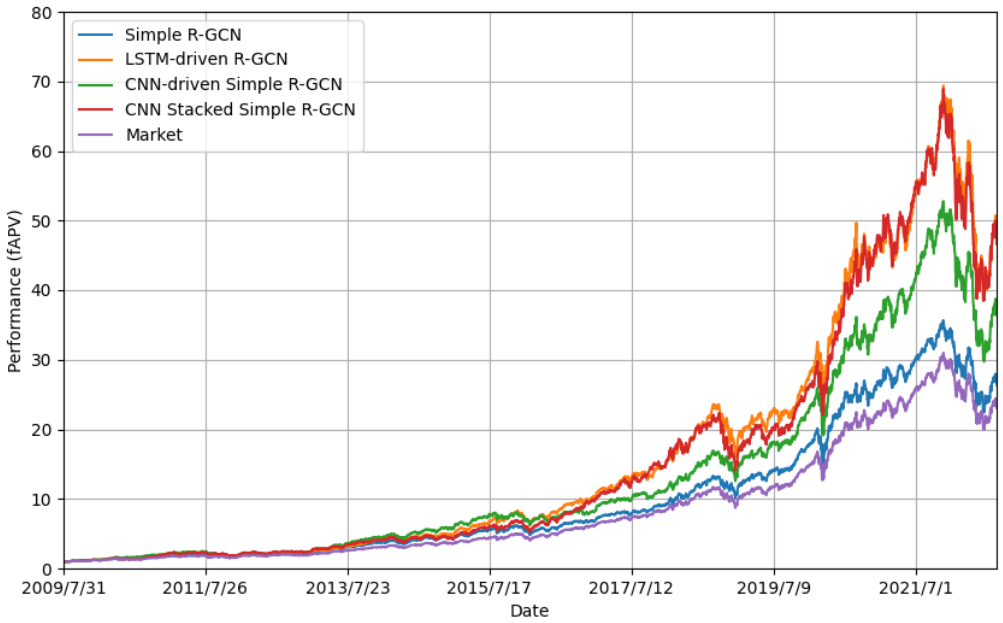

Fig.8 Performance of fAPV in the R-GCN group

From Fig.2, we can see that among the five proposed R-GCN structures, Structure II LSTM-driven R-GCN and Structure V CNN Stacked Simple R-GCN can perform comparatively well.

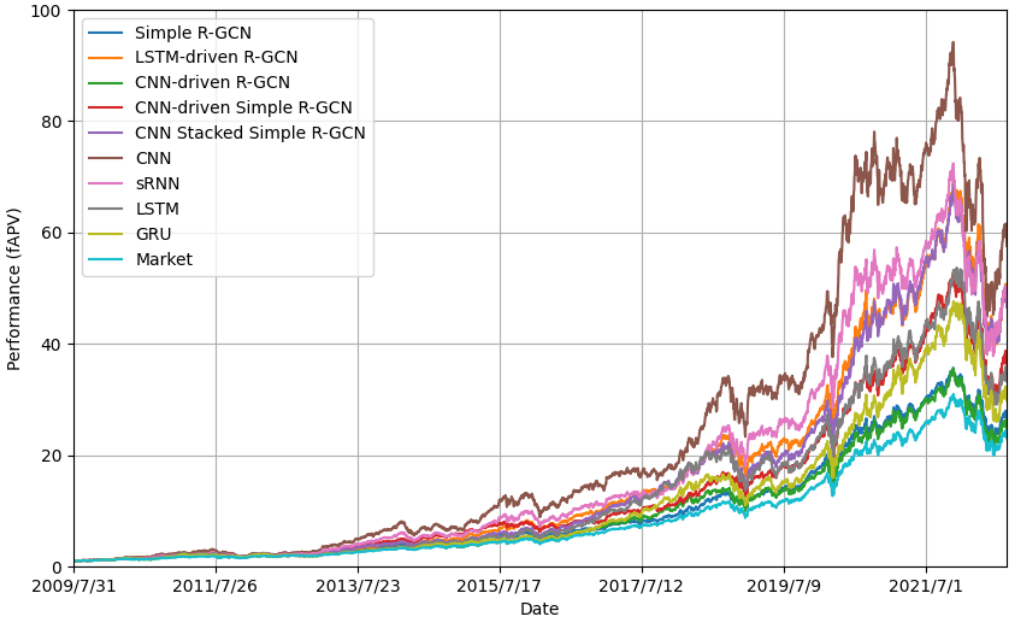

Fig.9 Performance of fAPV in the R-GCN and control groups

From Fig.3, we can roughly find that the CNN performs best under the fAPV metric, followed by the LSTM-Driven R-GCN, CNN Stacked Simple R-GCN and sRNN.

Chapter 7: Analysis

7.1 Simple R-GCN and its Derivative Structures

Simple R-GCN is the first R-GCN structure we have given. After analyzing its fAPV curve, we found that it behaves very smoothly. Its Beta is very close to 1, which indicates that its fluctuations are quite similar to the market's volatility. But this equally means that it will gain less. The fact that it achieves the lowest active investment return (alpha) of all the networks involved in this paper shows that its ARR is only 1.32% higher than that of the market portfolio. However, we do not conclude from this that it is an unsuitable network for application to portfolio management. Considering the very stable performance of this network, we propose the next four networks which combine R-GCN with the better-performing CNN and LSTM. By blending the CNN and LSTM with the R-GCN, it is hoped that their performance can be better traded off between return and stability.

By analyzing Figure 1, we noticed that using CNNs and LSTMs to drive the R-GCN did significantly improve its performance, in other words, we achieved our aim of designing these network structures. Also, it is readily apparent that the antecedent network to the R-GCN layer must be an LSTM layer, otherwise the R-GCN layer will be ineffective.

7.2 Analysing from all metrics

In the following paragraphs, we discuss the performance of each network in terms of each metric.

- fAPV: From the fAPV point of view, the LSTM-driven R-GCN performed the best among the R-GCNs, with an fAPV of 47.3653. However, when compared with the control group, it did not perform as well as the CNN network, which had an fAPV of 57.1339.
- ARR: In terms of ARR, both LSTM-driven R-GCN and CNN Stacked Simple R-GCN performed remarkably well, with ARRs of 0.3442 and 0.3424, respectively. Likewise, none of them performed better than the CNN network when compared to the control group, which had an ARR of 0.3636.

Both fAPV and ARR consider a strategy or neural network from the perspective of returns only. fAPV simply takes into account the value of the portfolio at the beginning and end of the investment period and is a relatively static indicator. The reason why CNNs perform relatively better in these two metrics, we believe, is mainly due to the fact that CNNs are a highly promising network, but their performance in the field of portfolio management can vary greatly depending on the richness of the input signals. The reason why CNNs perform relatively better in these two metrics, we believe, is mainly due to the fact that CNNs are a highly promising network, but their performance in the field of portfolio management can vary greatly depending on the richness of the input signals. In this paper, our input signals cover the money market, bond market, capital market, commodity market, and futures market to provide as comprehensive a picture of the financial markets as possible. It is like in the field of computer vision where we feed a more complete image into the CNN instead of a fragmented image, then it will naturally get a significant performance boost.

- SR: Regarding SR, the LSTM-driven R-GCN is the best performing R-GCN with 1.3912. It is followed by the CNN-driven Simple R-GCN with an SR of 1.3353. Meanwhile, they are also the best two networks in both the R-GCN group and the control group. We also noticed that only these two networks exceeded the SR of the Market Portfolio by 1.2998 among all the networks, the rest of the networks were below this value. CNN is only 1.2168 in this metric, which is not yet as high as sRNN's 1.2699. The Sharpe ratio is an indicator that considers both return and risk. fAPV and ARR are higher than the market portfolio, but SR is lower than the market portfolio, implying that these networks are taking unreasonable risks in order to achieve higher returns and rewards. Perhaps we could improve the reward function by incorporating a Sharpe ratio, allowing our agents to weigh the benefits against the risks.
- MDD: Observing the MDD, there is no doubt that Simple R-GCN is the best performer with an MDD of 0.3708, compared to 0.3541 for the market portfolio, which is fairly close. Following this is the CNN-driven Simple R-GCN, whose MDD has already reached 0.4367. Apart from this, the R-GCN group all outperformed the control group. In other words, the largest MDD in the R-GCN

group is smaller than the smallest MDD in the control group. It also demonstrates that the robustness of our proposed network structure has indeed been improved and that our design objectives have been met.

- Alpha (α): In the case of Alpha, it measures the return of the portfolio over the market portfolio, which means it evaluates performance from a return perspective. Therefore, similar to fAPV and ARR, the CNNs in the control group should perform well. The results also confirm our inference. The LSTM-driven R-GCN and CNN Stacked Simple R-GCN in the R-GCN group gave the best performance in the group with 0.4281 and 0.4230 respectively. However, they were both unsurprisingly lower than the 0.4803 for CNN in the control group.
- Beta (β): Finally, turn to Beta, it depicts the sensitivity of a portfolio relative to the market, thus a high or low value does not indicate a successful performance of the investment. Among all the networks involved in the paper, the CNN has a maximum beta of 2.1905, which indicates that CNN networks are very sensitive to the market. In the R-GCN group, the LSTM-driven R-GCN has the largest beta value of 1.5748. The historical data we use spans the period 2008-5-23 to 2022-8-23, and if we put this decade-plus of data into a nearly 100-year duration, we would see that the last decade or so has been in a long-term upward trend. And when the market is in a bull market, stocks or portfolios with high beta tend to yield higher returns for investors. This explains why CNN networks can outperform in our experiments. Combined with SR, as the market is in an upward trend, the CNN tends to take more risk to expand returns, even though its marginal return from risk-taking is already considerably low (its SR is substantially lower than that of the market portfolio).

7.3 Performance before Covid-19

The vast majority of investors are risk averse, who can accept high returns but are weary of slightly larger drawbacks. This is why MDD is a highly common and important metric in the financial sector. However, a maximum draw down over 40% is still very significant. Nevertheless, we take note of the concentration of significant retracements occurred towards the end of the online study, which is 2022. It is possible that this is due to the increase in black swan events since the onset of covid-19, leading to a more unpredictable equity price and consequently an increase in dropdowns. If we concentrate only on the performance prior to the Covid-19 (2009-07-31 to 2019-7-31), we would obtain the following results.

Table 8 Performance under all metrics across networks after exclusion of the Covid-19 period

Metrics	fAPV	ARR	SR	MDD	Alpha (annualized)	Beta (averaged)
Simple R-GCN	14.2473	0.3050	**1.5324**	**0.2636**	0.1279	1.1409
LSTM-driven R-GCN	**22.3784**	**0.3654**	**1.6402**	0.3409	**0.2769**	**1.4337**
CNN-driven R-GCN	13.7538	0.3004	1.3805	0.2913	0.1098	1.0778
CNN-driven Simple R-GCN	18.0244	0.3361	**1.5647**	0.3651	0.2171	1.3803
CNN Stacked Simple R-GCN	20.3802	**0.3526**	1.4413	0.3707	**0.2526**	1.3839
CNN	**33.8048**	**0.4230**	1.4592	0.4814	**0.3686**	**1.9674**
sRNN	26.1492	0.3868	1.5080	0.4167	0.3138	1.5824
LSTM	18.4460	0.3392	1.3452	0.3625	0.2241	1.3013
GRU	15.3059	0.3144	1.3035	0.3397	0.1596	1.1536
Market	11.9258	0.2819	**1.5209**	**0.2550**	0	1

The performance of each network under all measures is shown in the table 8.

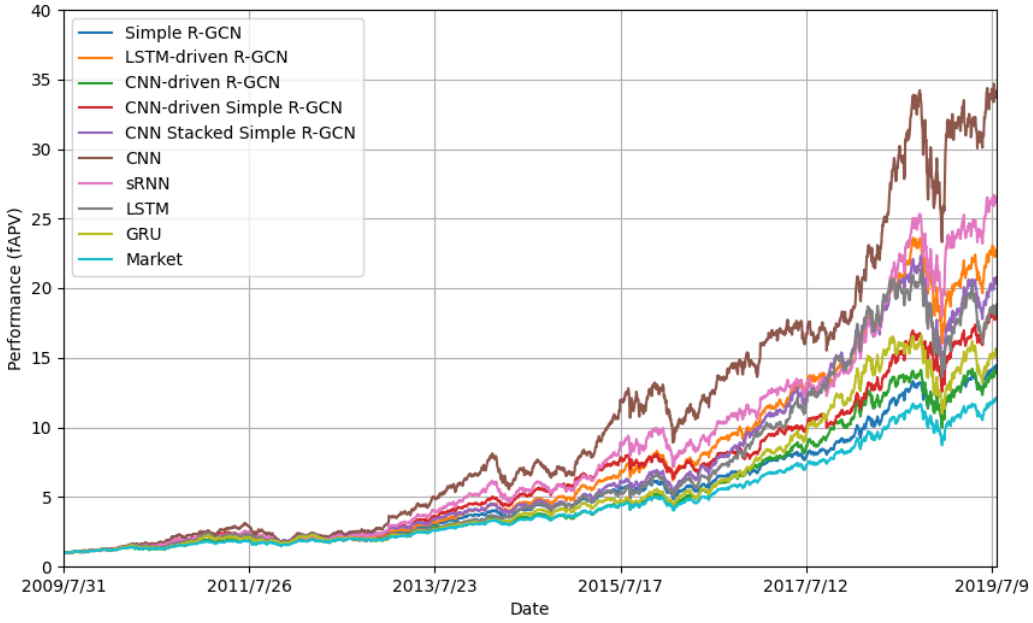

Fig.10 fAPV curves for each network after exclusion of the Covid-19 period

The fAPV curves for each network are summarised in Fig.10

Comparing Table I and Table II, we find that when we focus on the performance before covid-19 crisis, the MDD of each network drops. This supports our suspicions.

In the R-GCN group, the MDD decreased from a mean of 0.4177 to 0.3263, a reduction of more than 0.09. As a comparison, the MDD of the control group decreased from 0.4791 to 0.4001. Comparing the best networks in both groups, the MDD of the CNN decreased by only 0.04, while the MDD of the LSTM-driven R-GCN declined by 0.1.

Chapter 8: Conclusion

Among the five R-GCN structures we present, Simple R-GCN is the most basic, and it slightly outperforms the market portfolio under all metrics. However, it is characterised by consistent excess returns, so we propose the latter four structures with the hope of smoothing out the network's fAPV curve. Where CNN-driven R-GCN performs even worse than Simple R-GCN, but CNN-driven Simple R-GCN outperforms Simple R-GCN, we give the conclusion that the R-GCN layer needs to be connected after the LSTM layer (or presumably the RNN layer) to be valid. Across all R-GCN structures, the LSTM-driven R-GCN and the CNN Stacked Simple R-GCN performed best and similarly. Comparing them, the gaps in performance are very small and almost negligible when investigating the three return metrics fAPV, ARR, and Alpha, as well as Beta, a sensitivity metric. When inspecting SR, a return per unit risk indicator, and MDD, a volatility indicator, however, differences in their performance emerge. The LSTM-driven R-GCN outperformed the CNN Stacked Simple R-GCN on both SR and MMD. It demonstrates that the LSTM-driven R-GCN can achieve the same return as the CNN Stacked Simple R-GCN while taking less risk, and the fAPV curve is even smoother. In a nutshell, the LSTM-driven R-GCN is the best-performing of the five R-GCN structures in this paper.

In a comparison of LSTM-driven R-GCN with four common networks CNN, Simple RNN, LSTM and GRU, LSTM-driven R-GCN beats the networks except CNN in all aspects. Nevertheless, it does not mean that LSTM-driven R-GCN is inferior to CNN. The CNN outperforms the LSTM-driven R-GCN on three performance metrics, fAPV, ARR and Alpha, but is weaker than the LSTM-driven R-GCN on SR and MDD. We have discussed this question in the previous chapter, where we argued that the CNN is a very resilient network and that due to the sensible choice of market signals we use as input, the CNN performs remarkably well in our experiments. Yet the high returns of the CNN derive from excessive risk-taking, which leads to its SR being significantly lower than that of the market portfolio. And the MDD of the CNN is substantial, even after excluding the effects of the Covid-19 crisis. The LSTM-driven R-GCN has the highest SR of all networks and only higher MDD than the Simple R-GCN, despite the fact that its revenue metric is not comparable to that of the CNN. This illustrates that the LSTM-driven R-GCN balances risk and gain fairly well, which achieves our original intention of devising the structure.

In summary, the R-GCN is a high-potential neural network in the framework of deep reinforcement learning. Our proposed LSTM-driven R-GCN is able to trade-off well between earnings and risk, and its fAPV curve is relatively smoother.

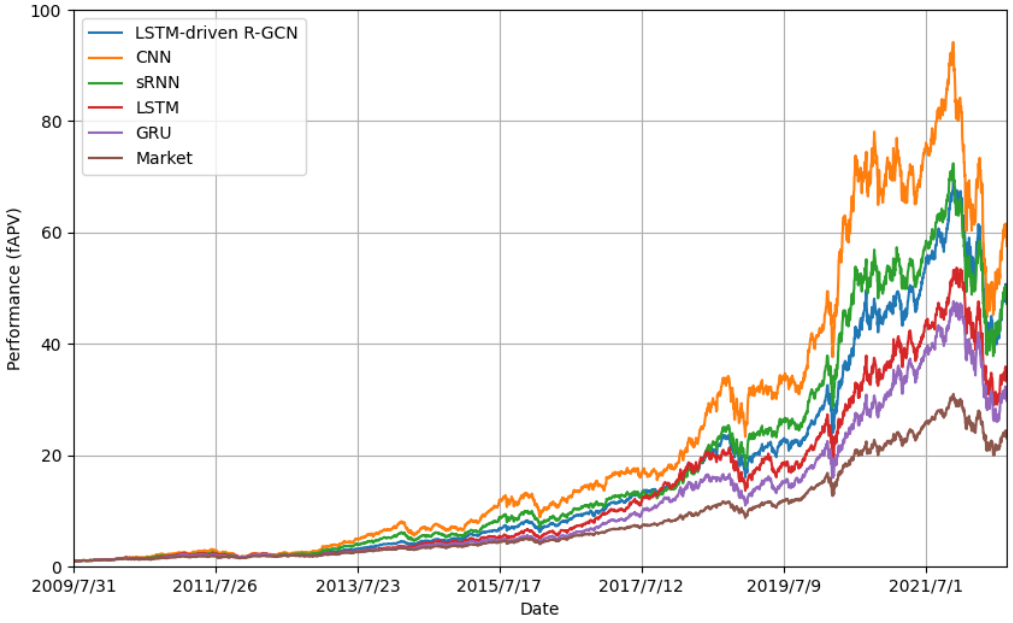

Fig.11 The fAPV curves of LSTM-driven R-GCN and control groups

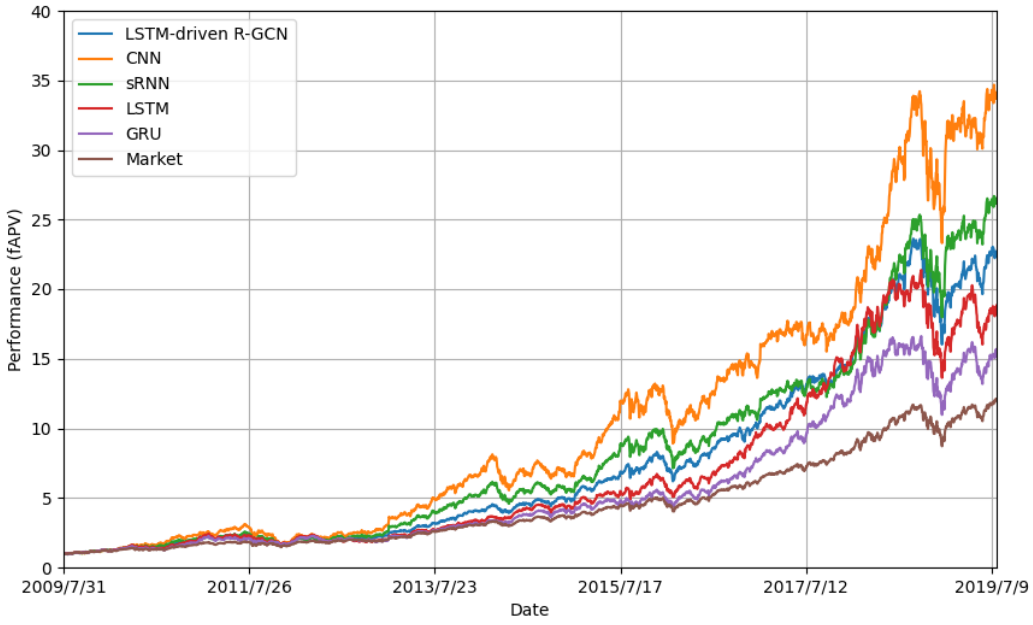

Fig.12 The fAPV curves of LSTM-driven R-GCN and controls excluding the covid-19 period

Chapter 9: Future Research

Our future research could improve performance by tuning various parts of the deep reinforcement learning framework.

9.1 Adjusting the Optimisation Algorithm

In this paper, we have chosen to use the DDPG algorithm in view of outputting only one definite action. In [19], Liang et al. states that the PG algorithm will outperform both PPO and DDPG, so we can adjust the optimisation algorithm used for our agents to replace DDPG with PG and PPO, and even some more advanced algorithms such as, TD3 [27] and D4PG [28].

9.2 Adjusting the Neural Networks

9.2.1 Increase the number of network layers

In this paper, in order to reduce the time complexity of the computation, the same type of network hierarchy is stacked with no more than 1 layer in the network structure we use. In [23], Shi et al. discovered through experiments that better average performance can be obtained when the number of layers of R-GCN is 2 to 3. So we can appropriately increase the number of layers of the network to achieve better results in extracting features.

9.2.2 Altering the network structure

In our experiments we have found that CNNs can produce exceptionally high performance, but their MDD is likewise significant. If we can effectively combine the respective strengths of CNN and R-GCN, we should get a network structure that would perform well. Unfortunately, Structures III, IV and V proposed in this paper fail to unleash the advantages of the CNN. Therefore, we can adjust the network structure to stimulate the potential of the CNN. For example, the addition operations in Structures IV and V in this paper can be changed to a concatenation operation, which is then passed through a CNN using a 1x2 convolutional kernel and finally fed into the output layer. In light of the performance of the sRNN, it is also a good idea to try using the sRNN instead of the LSTM to drive the R-GCN.

9.3 Adjusting the Reward Function

In this paper we use $\ln(r_t)$ as the reward function, which causes agents to consider only the reward and ignore the risk. We can therefore adjust our reward function so that agents also focus on the risk and volatility of the portfolio. For example, the reward function would be adjusted to,

$$Reward\ Function = w_r \cdot \ln(r_t) + w_\sigma \cdot \ln(SR_t) + w_{mdd} \cdot \ln(MDD_t)$$

where w_r, w_σ and w_{mdd} denote the weights of return, Sharpe Ratio and MDD respectively, $\ln(r_t)$, $\ln(SR_t)$ and $\ln(MDD_t)$ denote the natural logarithmic values of return, Sharpe Ratio and MDD at time t respectively.

References

[1] Reilly, F.K. and Brown, K.C., 2011. Investment analysis and portfolio management. Cengage Learning.

[2] Macroption, 2022. Asset Classes: List, Characteristics, Asset Allocation [online]. Available at: https://www.macroption.com/asset-classes/ [Accessed 31 August 2022]

[3] Kissell, R.L., 2013. The science of algorithmic trading and portfolio management. Academic Press.

[4] Markowitz, H.M., 1968. Portfolio selection: efficient diversification of investments.

[5] French, Craig W., The Treynor Capital Asset Pricing Model. Journal of Investment Management, Vol. 1, No. 2, pp. 60-72, 2003, Available at SSRN: https://ssrn.com/abstract=447580

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

[7] Lee, M.C., 2009. Using support vector machine with a hybrid feature selection method to the stock trend prediction. Expert Systems with Applications, 36(8), pp.10896-10904.

[8] Kumar, M. and Thenmozhi, M., 2006, January. Forecasting stock index movement: A comparison of support vector machines and random forest. In Indian institute of capital markets 9th capital markets conference paper.

[9] Feng, G., Giglio, S. and Xiu, D., 2017. Taming the factor zoo. Fama-Miller Working Paper, 24070.

[10] Rapach, D.E., Strauss, J.K. and Zhou, G., 2013. International stock return predictability: what is the role of the United States?. The Journal of Finance, 68(4), pp.1633-1662.

[11] Chapados, N. and Bengio, Y., 2001. Cost functions and model combination for VaR-based asset allocation using neural networks. IEEE Transactions on Neural Networks, 12(4), pp.890-906.

[12] Becker, S., Cheridito, P. and Jentzen, A., 2019. Deep optimal stopping. Journal of Machine Learning Research, 20, p.74.

[13] Bartram, S.M., Branke, J., De Rossi, G. and Motahari, M., 2021. Machine Learning for Active Portfolio Management. The Journal of Financial Data Science, 3(3), pp.9-30.

[14] Almahdi, S. and Yang, S.Y., 2017. An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. Expert Systems with Applications, 87, pp.267-279.

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

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

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

[18] Yao, W., Ren, X. and Su, J., 2022, March. An Inception Network with Bottleneck Attention Module for Deep Reinforcement Learning Framework in Financial Portfolio Management. In 2022 7th International Conference on Big Data Analytics (ICBDA) (pp. 310-316). IEEE.

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

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

[21] Feng, F., He, X., Wang, X., Luo, C., Liu, Y. and Chua, T.S., 2019. Temporal relational ranking for stock prediction. ACM Transactions on Information Systems (TOIS), 37(2), pp.1-30.

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

[23] Shi, S., Li, J., Li, G., Pan, P., Chen, Q. and Sun, Q., 2022. GPM: A graph convolutional network based reinforcement learning framework for portfolio management. Neurocomputing, 498, pp.14-27.

[24] Kipf, T.N. and Welling, M., 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

[25] Bellman, R., 1957. A Markovian decision process. Journal of mathematics and mechanics, pp.679-684.

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

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

[28] Barth-Maron, G., Hoffman, M.W., Budden, D., Dabney, W., Horgan, D., Tb, D., Muldal, A., Heess, N. and Lillicrap, T., 2018. Distributed distributional deterministic policy gradients. arXiv preprint arXiv:1804.08617.