Applied Intelligence (2021) 51:8119–8131
https://doi.org/10.1007/s10489-021-02262-0

Portfolio management system in equity market neutral using reinforcement learning

Mu-En Wu[1] · Jia-Hao Syu[2] · Jerry Chun-Wei Lin[3] · Jan-Ming Ho[4]

Accepted: 4 February 2021 / Published online: 26 March 2021
© The Author(s) 2021

Abstract

Portfolio management involves position sizing and resource allocation. Traditional and generic portfolio strategies require forecasting of future stock prices as model inputs, which is not a trivial task since those values are difficult to obtain in the real-world applications. To overcome the above limitations and provide a better solution for portfolio management, we developed a Portfolio Management System (PMS) using reinforcement learning with two neural networks (CNN and RNN). A novel reward function involving Sharpe ratios is also proposed to evaluate the performance of the developed systems. Experimental results indicate that the PMS with the Sharpe ratio reward function exhibits outstanding performance, increasing return by 39.0% and decreasing drawdown by 13.7% on average compared to the reward function of trading return. In addition, the proposed `PMS_CNN` model is more suitable for the construction of a reinforcement learning portfolio, but has 1.98 times more drawdown risk than the `PMS_RNN`. Among the conducted datasets, the PMS outperforms the benchmark strategies in TW50 and traditional stocks, but is inferior to a benchmark strategy in the financial dataset. The PMS is profitable, effective, and offers lower investment risk among almost all datasets. The novel reward function involving the Sharpe ratio enhances performance, and well supports resource-allocation for empirical stock trading.

Keywords Portfolio management · Equity market neutral · Reinforcement learning · Convolutional and recurrent neural network

1 Introduction

Most conventional trading strategies generate trading signals based on predetermined subjective indicators, such as moving averages [1], relative strength index [2], and opening range breakout [3, 4]. However, most indicators provide only long (buy) and short (sell) signals, regardless of position size (the quantity of commodities) and risk management (the relevance of commodities). Thus, portfolio management is used to provide additional control over investments. There has been considerable research into the selection of commodities, position sizing, and resource allocation [5, 6].

Common portfolio strategies, such as modern portfolio theory (MPT) [7] and the Kelly criterion [8], require predictions pertaining to future stock prices as inputs for portfolio management. MPT relies heavily on the accuracy of the future mean and variance, whereas the Kelly criterion depends on the probability distribution of future returns. Nonetheless, a small deviation in the implementation of these strategies can greatly affect the weights used in the portfolio. Michaud [9] claimed that the main problem associated with MPT is its tendency to maximize error effects in input assumptions. Note that difficulties in forecasting the mean and variance of future commodity prices make this approach unsuitable for portfolio management in actual stock markets. Among the variants of portfolio strategies, Equity market neutral

✉ Jerry Chun-Wei Lin
jerrylin@ieee.org

Mu-En Wu
mnwu@ntut.edu.tw

Jia-Hao Syu
f08922011@ntu.edu.tw

Jan-Ming Ho
hoho@iis.sinica.edu.tw

1 Department of Information and Finance Management, National Taipei University of Technology, Taipei, Taiwan

2 Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan

3 Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, Bergen, Norway

4 Institute of Information Science, Academia Sinica, Taipei, Taiwan

(EMN) is a hedging strategy that enhances risk management [10] by balancing the holding of relatively strong stocks with the selling of relatively weak stocks; however, quantifying and classifying stocks according to their relative strengths and weaknesses can be difficult.

Reinforcement learning (RL) is an important topic in machine learning due to its flexibility in many domains [11, 12], and are used to support decision-making through enormous trial-and-test schemes. Neural network (NN) models can also be used as policy agents within the RL architecture. NNs comprise a number of highly-interconnected elements, processing information at multiple levels via dynamic responses to external information [13, 14]. With the rapid development of computing power and parallel computing, numerous layers of NN are then formalized, which is known as deep-learning [15, 16]. Common deep-learning architectures include recurrent neural networks (RNN) [17, 18] and conventional neural networks (CNN) [19, 20]. Financial data comprises time-series information related to prices, i.e., opening, highest, lowest, and closing prices (abbreviated as OHLC prices). Time-series can be regarded as sequential data well-suited to prediction tasks based on the RNN model.

The learning mechanism of RL is similar to human investment activities. Through the interaction between investors (learning agent) and the stock market (environment), investors can learn knowledge for further investment through trial-and-test of actions (investment decision) and rewards (investment return). Therefore, in this paper, we sought to overcome the above-mentioned limitations by developing an efficient Portfolio Management System (PMS) through the implementation of CNN and RNN networks within the RL architecture in order to support decision-making in the allocation of resources. The PMS presented in this paper comprises three modules: (1) data pre-processing; (2) NN-based RL portfolio; and (3) EMN strategy. The first data pre-processing module transforms price data into input tensors for the NN-based RL portfolio module, which in turn learns trading information from the data in order to generate long- and short-term portfolios. The EMN strategy module combines long and short positions to obtain the final states for portfolio management. In addition, we use two NN-based models (CNN and RNN) to deal with spatial and temporal information in order to refine the portfolio strategy. This makes it possible for the `PMS_CNN` and `PMS_RNN` systems to assign appropriate weights to stocks to assist in the allocation of resources for each training day. Previous works based on RL [21–23] employed trading returns as a reward function aimed at optimizing profitability; however, they tend to neglect stability and risk. The Sharpe ratio [24, 25] is a well-known indicator of trading performance used to optimize the trade-off between profitability and risk. In this study, we applied the Sharpe ratio to the reward function to estimate the profitability and stability of given strategy.

In our experiments, we used a number of indicators to evaluate performance of our PMS. Total return measures the profitability of the used strategy. The Sharpe ratio indicates the amount of profit that can be earned for a given unit of risk (volatility), whereas maximum drawdown (MDD) [26] indicates the maximum losses that can be borne during trading. These are the most common terms used to indicate the effectiveness and efficiency of trading strategies. Experiment results indicate that the proposed PMS scheme in conjunction with the reward function of Sharpe ratio can outperform that with the reward function of trading returns, resulting in a 39% improvement in returns and a 13.7% reduction in drawdown (novel and effective reward function). Furthermore, the `PMS_CNN` outperforms `PMS_RNN` in terms of returns and the Sharpe ratio, making it suitable for EMN portfolios. The proposed PMS outperformed existing benchmark strategies (`UCRP` [27], `Winner`, `Loser` [28, 29]) on all measures using industrial and TW50 datasets. Our PMS was outperformed by conventional methods when using datasets from the financial industry; however, it still achieved positive profits and Sharpe ratio. When applied to a dataset from the electronics industry, our PMS suffered large drawdown and fluctuations in the second half; however, it remained more profitable than conventional strategies (scalability in different datasets). Overall, the proposed PMS remains profitable and low-risk regardless of the dataset, and the novel reward functions involving the Sharpe ratio and return factors truly enhance performance. In conclusion, the proposed PMS is a highly valuable tool for resource-allocation for empirical stock trading, and makes three major contributions:

1. Proposed a RL-based portfolio management system, concatenated with CNN and RNN networks to support resource-allocation for empirical stock trading.
2. Provide a novel and effective reward function based on the Sharpe ratio to assess the trade-off between profitability and risk.
3. Experiment results demonstrate the applicability of CNN to the formulation of an EMN portfolio as well as the scalability of the proposed PMS to a variety of datasets.

2 Literature review

2.1 Equity market neutral (EMN)

EMN is a hedging strategy commonly used for portfolio management, and aims to exploit the differences trends in stock trend and attribute [10]. EMN is based on short

selling relatively weak commodities, holding onto relatively strong commodities, and earning the spread between the two positions. All of the cash from short positions is reinvested in the market and covers all of the long positions, while the equivalent long and short positions are used to reduce systemic risk in the market. Theoretically, EMN does not impose any investment capital, and the net investment money is zero. In fact, short selling requires a certain margin to maintain the balance of EMN, and EMN still requires some investment capital in practice.

Many variants of the EMN model have been developed in recent decades. Alexander and Dimitriu [30] proposed the EMN strategies with enhanced index tracking, wherein portfolios are optimized through cointegration rather than correlation, resulting in 2% annual volatility in the DJIA. Shyum et al. [31] proposed a multi-factor model using fundamental and technical descriptors to construct EMN strategies for the Taiwan stock market. The multi-factor EMN strategies generated 25.4% returns with a Sharpe ratio of 1.80, at a time when the market produced 32.0% returns and Sharpe ratio of 1.26, which enhanced the stables but sanctify the profitability. Vijayalakshmi and Michel [32] solved complex constrained EMN portfolios using differential evolution for optimization.

The above studies demonstrate the effectiveness and profitability of EMN-based strategies for portfolio management; however, it should be noted that quantifying and classifying stocks according to their attributes (strongness and weakness) is not a trivial task. Practically, investors usually treat the stock's past performance as a prediction of the stock's future performance, and use it to measure the attribute of strong. However, it requires a strong assumption that the attribute is stationary, which cannot be proven and is usually wrong in financial markets. Attributes can also be evaluated according to their features, such as the multi-factor model [31] mentioned above; however, feature collection is costly and there is always the problem of overfitting. Thus, we tried to utilize the state-of-art technique of RL to evaluate stock attributes, determine portfolio weights, and establish an EMN portfolio strategy.

2.2 Reinforcement learning (RL)

RL is a major discipline of machine-learning, inspired by the mechanisms underlying human learning, which is the interaction between the environment and the agent [11, 12]. The simple concept of RL is shown in Fig. 1. Software agents take appropriate action within the context of the current state. For infinite number of actions and states, the agents are usually constructed by the policy networks that uses the power of neural networks to memorize and predict appropriate action based on the current state. The rewards deriving from actions are then estimated by the environment

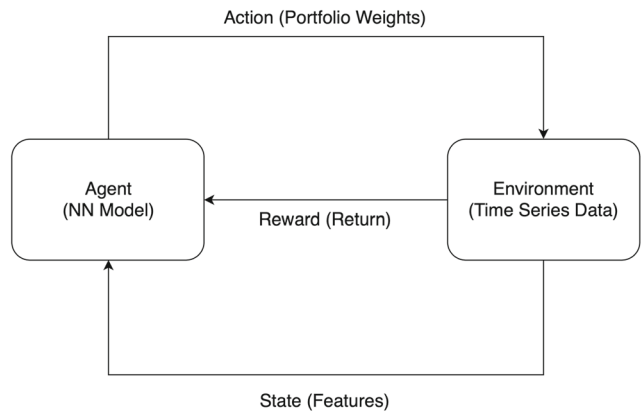

Fig. 1 A simple architecture of reinforcement learning

to guide the agent learning the subsequent actions in an iterative process. This progress can gradually improve the overall performance and find the potential models of the given problem based on the obtained experiences with the several trial-and-error steps.

RL is based on the framework of Markov decision process, which state that the future is independent of the past given the present [33]. The attributes of the Markov decision process can also be found in financial markets. The efficient market hypothesis demonstrates that historical information is fully and efficiently reflected in the present price [34]. Therefore, future information can be extracted from current information, which means that RL and Markov decision processes have the same view as financial markets and are suitable for financial problems.

With the rapid development of RL, two main branches including Q-learning and policy gradient have been developed. Q-learning is a model-free algorithm to learn optimal reaction under Markovian circumstances, and establishes a Q-table to memorize and predict actions with the concept of dynamic programming. Policy gradient is a RL approach to model and determine the policy (action) to obtain optimal rewards, and contains a policy network that predicts a probability distribution over actions based on current state [35]. Several derivative models are also developed, including deep policy gradient, deep deterministic policy gradient [36], etc.

The flexibility and generalizability of RL makes it an attractive scheme for many domains, particularly in planning business strategies [37], implementing automation systems, and controlling robot [38] for industrial applications. In the field of finance, Jiang and Liang [21, 22] formed a cryptocurrency portfolio using deep reinforcement learning, which resulted in 10-fold returns over periods of 1.8 months. However, the cryptocurrency market is quite frequent and volatile, and the trading returns are unstable compared with the stock market. Almahdi and Yang [23] proposed an adaptive portfolio trading system using recurrent RL and

mean-variance optimization. That system makes it possible to estimate drawdown risk and automatically retrain the system. Other relative RL-based works have also been implemented and discussed [39, 40], and the development is still in progress.

2.3 Convolutional and recurrent neural network (CNN & RNN)

CNN and RNN are deep neural-network architectures utilized in a variety of domains [19, 20]. CNN has demonstrated successful performance in the processing of multi-dimensional data (images and videos) [20], which is a subtype of the discriminative deep architecture. It also reduces learning complexity using convolution and pooling strategies. CNN has been used for recognition and classification tasks in multimedia, and is also used for financial data analysis. Tsantekidis et al. [41] proposed a deep CNN scheme for the prediction of price movements using a large dataset of high-frequency transactions records. Their scheme outperformed simple DNN and SVM in terms of recall as well as precision. Chen et al. [42] formulated a deep CNN model using planar feature representation for the analysis and prediction of stock prices. Their scheme achieved accuracy of 57.88% in three-category prediction.

RNN [17] differs from feedforward networks in its use of a feedback loop to maintain a connection with previous decisions to enable iterative progress. Their memory architecture was inspired by the learning and memory processes in humans, wherein past information and decisions held in memory can affect subsequent behaviors. RNN has proven highly effective in natural language processing [43] as well as finance. Gao and Chai [44] predicted stock closing prices by RNN model with principal component analysis (PCA) dimension reduction. Wang et al. [45] developed an Elman RNN architecture for the prediction of price indices on stock markets. Rout et al. [46] used a low-complexity recurrent neural network and evolutionary learning to forecast S&P500 data. Their scheme demonstrated very low variance and good performance in predicting volatility. Their scheme was shown to outperform common neural networks in forecasting financial time series. Most current research on the use of CNN and RNN in the field of finance focuses on price forecasting. By contrast, we focused on developing specific trading strategies and methods for portfolio management.

3 Methodology

The flowchart of the proposed PMS is presented in Fig. 2, including three major modules. The first module involved data preprocessing, which included data normalization and tensor packaging. The normalized training data was packed for use as an input tensor (features) for the RL module. Testing data was then used to verify the performance of the PMS and to generate portfolio weights for testing period.

The second is the RL module, including long and short RL models. RL module learn the portfolio weighting scheme through the interaction between the environment and the agent, and determines the long and short portfolios separately. The final module is EMN, which combines long and short portfolio weights to prevent long and short a same stock from wasting trading costs. At the same time, the EMN module establishes the final EMN portfolio weight.

3.1 Data pre-processing

Since there are differences in the scale of time-series features (different price scales for each stock) , thus the values of features (in this case prices) are required be normalized. The normalization method is shown in (1), and referred from [21]. We design the input tensor, T, as a $M \times 4 \times N$ vector. M is the number of investment assets, 4 is the number of features, and N is the length of the times-series for each feature. In this paper, we set N as 20 to gather the price information in the last 20 trading days.

$$
\begin{aligned}
T_t &= [P_t^1, P_t^2, ..., P_t^{M-1}, P_t^M] \\
P_t^i &= [Open_t^i, High_t^i, Low_t^i, Close_t^i] \\
Open_t^i &= \left[\frac{open_{t-n+1}^i}{close_t^i}, \frac{open_{t-n+2}^i}{close_t^i}, ..., \frac{open_{t-1}^i}{close_t^i}, \frac{open_t^i}{close_t^i}\right] \\
High_t^i &= \left[\frac{high_{t-n+1}^i}{close_t^i}, \frac{high_{t-n+2}^i}{close_t^i}, ..., \frac{high_{t-1}^i}{close_t^i}, \frac{high_t^i}{close_t^i}\right] \\
Low_t^i &= \left[\frac{low_{t-n+1}^i}{close_t^i}, \frac{low_{t-n+2}^i}{close_t^i}, ..., \frac{low_{t-1}^i}{close_t^i}, \frac{low_t^i}{close_t^i}\right] \\
Close_t^i &= \left[\frac{close_{t-n+1}^i}{close_t^i}, \frac{close_{t-n+2}^i}{close_t^i}, ..., \frac{close_{t-1}^i}{close_t^i}, 1\right]
\end{aligned}
\tag{1}
$$

T_t is the input tensor for a period t, which includes M price vectors (P_t) for each investment asset. P_t^i is the price vector ($4 \times N$-dimensional) for each asset i at a period t, including four dimensions, $Open_t^i$, $High_t^i$, Low_t^i, and $Close_t^i$. Each price is then divided by the latest closing price as $close_t^i$, to change the features into relative changes in price. Note that the notation of lower letters ($open$, $high$, low, $close$) represents a price data, and the notation of capital letters ($Open$, $High$, Low, $Close$) represents a N-dimensional price vector.

For example, if the closing prices of an asset i for the previous N days is [77.0, 80.0, 81.5, ... ,90.5], then all of

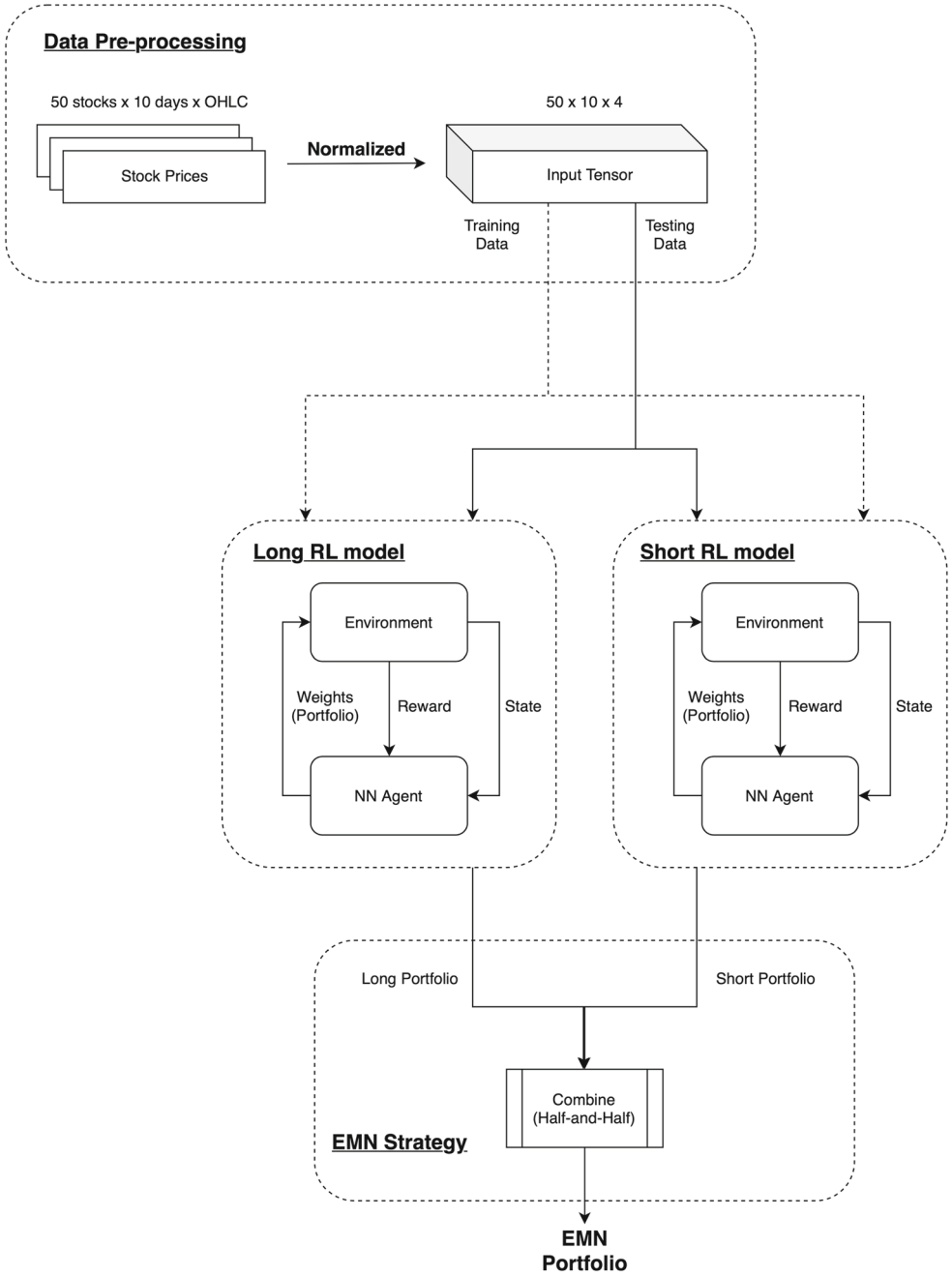

Fig. 2 The flowchart of the designed PMS

the prices are divided by 90.5 (i.e., the latest closing price ($close_t^i$) to become [0.851, 0.884, 0.901, ..., 1]. Following normalization, the price scale of all stocks is the same, such that the features represent the changes in price. The same normalization method is applied to $High_t^i$, Low_t^i, and $Open_t^i$, as shown in (1).

After preprocessing of the price normalization, the input tensors obtain the change rates and fluctuations of stock prices. Since the short-rebalancing investment is considered in the developed systems, the short-term information of price changes should be concerned for the RL model to learn the weighting values of varied portfolios.

3.2 RL environment

There are two major tasks for the RL environment as shown in Fig. 1, which are calculating the reward of the received action and transferring the next state to the agent.

The environment of RL architecture receives the actions from agent, and provides the reward of the corresponding actions (portfolio weight, W_t). Firstly, the environment will calculate the asset value and daily return by the following equations. In (2), PC_t is the price change from period $t-1$ to t (closing price of period t divided by the closing price of period $t-1$), and is a transposed M-dimensional vector

for M assets. a_t is the net asset value of period t. W_t is the portfolio weight, which is a M-dimensional vector. $W_{t-1} \times PC_t$ denotes that the portfolio weight multiplies the price change, which is the change of the net asset value compared to the previous period. The 1-norm term represents the change of portfolio weight (how much stocks you would sell or buy), after multiplied by the rate of transaction fee, δ, it becomes the value of transaction fee, which will deduct your net asset value. In the following experiment, we set δ to 0.25%, which is the average transaction fee for the Taiwan stock market. r_t is the daily return at period t, and is the natural logarithm of the change for the net asset value.

$$PC_t = \left[\frac{P^{close}_{1,t}}{P^{close}_{1,t-1}}, \frac{P^{close}_{2,t}}{P^{close}_{2,t-1}}, ..., \frac{P^{close}_{M,t}}{P^{close}_{M,t-1}} \right]^T$$

$$a_t = a_{t-1} \cdot W_{t-1} \times PC_t - \delta \cdot \|a_t \cdot W_t - a_{t-1} \cdot W_{t-1}\|_1$$

$$r_t = ln\left(\frac{a_t}{a_{t-1}}\right) = ln\left(W_{t-1} \times PC_t - \frac{\delta \cdot \|a_t \cdot W_t - a_{t-1} \cdot W_{t-1}\|_1}{a_{t-1}}\right) \tag{2}$$

$$\text{Trading Return} = \frac{a_T}{a_0}$$

$$\text{Sharpe Ratio} = \frac{\text{Annual Return}}{\text{Annualized Standard Deviation of Return}} = \frac{\sqrt[years]{\text{Trading Return}}}{\text{Annualized Standard Deviation of Return}} = \frac{\sqrt[years]{\frac{a_T}{a_0}}}{std(r_t) \times \sqrt{252}} \tag{3}$$

Secondly, the environment will calculate the reward for the corresponding actions. Most previous works used trading return as a reward function of RL to optimize the profitability. However, these studies failed to consider stability or risk. In this paper, we adopted the Sharpe ratio as a novel reward function to optimize profitability while considering the tradeoff with risk. Therefore, we utilize both the trading return (general factor) and the Sharpe ratio (novel factor) as two different reward functions, and the mathematical expressions of which are shown in (3). Trading return is the final asset value (in a given period T, a_T) divided by the initial asset value, a_0. Sharpe ratio is the annual return divided by annualized standard deviation of the return, where the annual return is the $years$ root of the trading return ($years$ is the number of years of the trading process). The annualized standard deviation of return is the standard deviation of the daily return multiplied by $\sqrt{252}$ (252 is the approximate number of trading days in a year, and is the frequently used factor in the financial field).

The RL environment adopts the above formula to provide rewards (trading return or Sharpe ratio) of the corresponding actions (W_t), which will guide the agent to update the prediction model. At the same time, the environment will transfer the next state (normalized price tensor in Section 3.1), for the agent to predict the next action (W_{t+1}). The rebalance period (trading period) in the following experiments was set at one day, which means that the environment sends historical OHLC data to the agent (state) daily in order to obtain a portfolio weight (action) from the agent.

3.3 NN-based policy network for RL agent

In this study, RL is used for portfolio management. The policy network we designed takes the price tensor (T, $M \times 4 \times N$ tensor) as an input, and predicts the weight of the portfolio (W) as an output. W is designed as a M-dimension vector with a sum of 1 (by the softmax layer), representing the portfolio weight of M asset. For each period, the environment transfers a reward by the given action, which will guide the agent to upgrade the policy network. For each updating epoch, the NN agent will obtain better patterns for portfolio management, enabling construction of a suitable portfolio strategy.

In this paper, we utilize two types of neural networks in the designed PMS to obtain the temporal and spatial information. In the proposed system, the PMS with CNN and RNN are respectively named as `PMS_CNN` and `PMS_RNN`, and the networks are shown in Tables 1 and 2. Table 1 presents the designed network of the CNN agent, which determines the spatial coherence of features via convolution. In the layers of Conv_2D, we implement the stride of convolution on the direction of the time series to discover the patterns of time series through the network. In addition, the second layer of Conv_2D has a kernel of (1, 3rd-dim previous output), which can compress the size of the third dimension as 1 of the output tensors regardless of the input size N. After feature extraction by convolution layers, we simply add a dense layer with M neurons and softmax activation function at the end of the network as the portfolio weights. Compared with the basic neural network, the utilized RNN in the design system can obtain past information through the memory states, and the utilized CNN can extract spatial information from different linear-transformations through the learned kernels, which can solve the limitations of basic neural networks. In this way, a better balance between profit and risk can be obtained and achieved in portfolio management. Note that in all convolutional layers of the CNN network, the stride is set as 1, and there is no padding involved in the network.

Table 2 presents the designed network of the RNN agent, which excels at detecting coherence in time series of features using memory states. In the designed RNN, we adopt a single-layer LSTM with a dropout layer, and end at a M-neuron dense layer with softmax to construct

Table 1 The designed network of the CNN agent

Layer	# Filter	Filter size	Activation
Input		$M \times 4 \times N$ Vector	
Conv_2D	2	(1,2)	relu
Conv_2D	48	(1, 3rd-dim previous output)	relu
Conv_2D	1	(1, 3rd-dim previous output)	relu
Flatten	–	–	–
Dense	M	–	softmax
Output		M-dimensional Vector	

a M-dimensional output, which is treated as the weight of portfolio.

3.4 EMN strategy

EMN is a powerful hedging portfolio strategy in which relatively weak commodities are shorted (sold) and relatively strong commodities are bought. We innovatively utilize the EMN strategy as the trading mechanism of the RL-based portfolio management system. Based on the concept of EMN strategy, the size of long and short positions should be equal. We design the PMS to be decomposed into long and short independent RL models, and equally sum the output weights (of each stock) of both models to construct the final portfolio weight.

Two independent models are used to learn the patterns associated with taking long and short positions with the aim of recommending appropriate portfolio weights for each asset. Equation (4) shows that the sum of long-weight (W_L) and short-weight (W_S), are both equal to 1, where M is the number of assets. To prevent situations where a given asset is simultaneously bought and sold, the final portfolio weight, W_C , combined with the long-weight and short-weight, W_L and W_S. Equation (5) shows that the sum of the combined weights. The zero W_C indicates that the size of the long position is equal to the short position, and indicates that EMN reinvested all the cash from short position into the market, and EMN-based PMS does not require any investment capital (money) in theory. In the designed system, the long and short RL models are repetitively named as `-Long` and `-Short`.

$$\begin{aligned} W_L &= (W_{L1}, W_{L2}, ..., W_{LM}), \quad W_{Li} \in [0, 1], \quad \sum_{i=1}^{M} W_{Li} = 1 \\ W_S &= (W_{S1}, W_{S2}, ..., W_{SM}), \quad W_{Si} \in [0, 1], \quad \sum_{i=1}^{M} W_{Si} = 1 \end{aligned} \tag{4}$$

$$\begin{aligned} W_C &= (W_{L1} - W_{S1}, W_{L2} - W_{S2}, ..., W_{LM} - W_{SM}) \\ \sum_{i=1}^{M} W_{Ci} &= \sum_{i=1}^{M} W_{Li} - W_{Si} \\ &= \sum_{i=1}^{M} W_{Li} - \sum_{i=1}^{M} W_{Si} = 1 - 1 = 0 \end{aligned} \tag{5}$$

Table 2 The designed network of the RNN agent

Layer	Detail
Input	$M \times 4 \times N$ Vector
LSTM	Neurons = 51
Dropout	0.2
Flatten	–
Dense	Neurons = M, softmax
Output	M-dimensional Vector

4 Experimental results

In this paper, the trading target (stock pool) was constituent stocks of the 50 premium stocks approved by the government of Taiwan, Taiwan 50 (TW50) [47]. The data of stock price was provided by Taiwan Stock Exchange, including the daily OHLC data.

We first split the TW50 data into training and testing datasets. Data from Aug. 1 2015 to Jul. 31 2017 was used as training data, and the remainder (Aug. 1 2017 to Jul. 31 2019) was used as testing data to verify the performance of the proposed systems. **Total return**, **Sharpe ratio**, **maximum draw down** (**MDD**), and **profit factor** (**PF**) were used as performance measures verifying the performance of the proposed PMS. Total return refers to the net profits throughout the entire trading period divided by the initial assets. Sharpe ratio refers to the total return divided by the standard deviation of daily profits [24, 25].

MDD refers to the maximum observed loss from a peak to a trough, before a new peak is reached [26]. Essentially, it indicates the maximum rate of loss during the trading period, where a lower value indicates better performance. PF refers to the net profit divided by the absolute value of the net loss.

4.1 Comparison of different reward function

The reward function in PMS is used to assist RL agents to learn when trading on an actual market. Different to the conventional reward function of trading returns [21], we use the Sharpe ratio as a novel reward function in order to account for investment risk when learning and building the system.

Table 3 compares the performance of the proposed PMS_CNN and PMS_RNN when using average return (Ret) and Sharpe ratio (Sha) as reward functions. As shown in Table 3, the performance of PMS_CNN was similar to that of PMS_RNN; however, the choice of reward function significantly affected the results. Overall, the Sharpe ratio reward function (PMS_CNN-Sha and PMS_RNN-Sha) outperformed the return reward function (PMS_CNN-Ret and PMS_RNN-Ret) in terms of profitability and risk. On average, PMS_CNN-Sha and PMS_RNN-Sha increase, 39.0% return 0.11152 Sharpe ratio, and 0.31781 profit factor (PF), indicating that the Sharpe ratio reward function achieved 11.152% higher profits than those obtained using the return reward function when facing a unit of volatility and 31.781% higher profits when facing a unit of loss. Using the Sharpe ratio reward function also reduced the average MDD value from 27.179% to 13.436%.

Figure 3 shows the equity curves of the compared systems under various reward functions on TW50 dataset. The PMS_CNN-Sha achieved the highest profits, whereas PMS_RNN-Sha was the most stable (i.e., extremely small drawdown). PMS_CNN-Ret and PMS_RNN-Ret presented larger fluctuations, despite achieving profits on par with PMS_RNN-Ret. Overall, the Sharpe ratio reward function outperformed the return reward function. Thus, the Sharpe ratio reward function is used in all subsequent experiments.

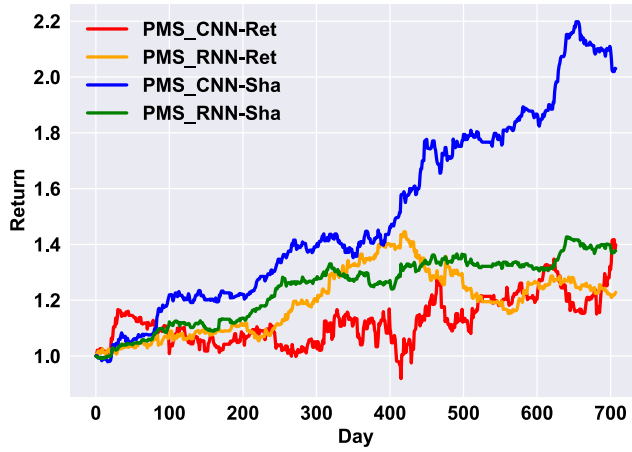

Fig. 3 Equity curves of PMS with different reward functions on TW50. Systems with reward function of Sharpe ratio are relatively profitable and stable

4.2 Performance of long and short models in the PMS

In the proposed PMS, we created two neural networks for the RL agents, respectively referred to as PMS_CNN and PMS_RNN. As described in Section 3.4, we trained and tested a long model to derive buying patterns and a short model to derive selling patterns from actual markets. In the following experiments, we respectively investigated the trading performance of taking long and short models when using our PMS, as shown in Table 4. Please note that the values shown in bold are the best performance for each column.

Table 4 shows that the total return and Sharpe ratios of PMS_CNN-Long and PMS_CNN-Short are similar. Note that the MDD of the short models (PMS_CNN-Short and PMS_RNN-Short) was 1.85 times of the long models (PMS_CNN-Long and PMS_RNN-Long). It also indicates that a short position is subject to higher risks, higher losses, and a smaller PF. The phenomenon is consistent with domain knowledge that short selling is relatively risky but profitable [48], which can be seen as the left figure of Fig. 4. In the end, the long and short models produced similar profits; however, the long model (blue curve) rose steadily,

Table 3 Performance indicators of PMS with different reward functions on TW50

Developed systems	Return	Sharpe	MDD	PF
PMS_CNN-Ret	39.839%	0.38759	0.24989	1.10880
PMS_RNN-Ret	22.859%	0.39872	0.29369	1.12610
PMS_CNN-Sha	**103.041%**	**0.57819**	0.17843	**1.54761**
PMS_RNN-Sha	37.735%	0.43116	**0.09028**	1.32291

The values in bold are the best performance for each column

Table 4 Performance indicators of the long, short, and combined models of the PMS

System	Return	Sharpe	MDD	PF
PMS_CNN-Long	47.971%	0.64282	**0.08999**	**1.46895**
PMS_CNN-Short	55.069%	**0.65812**	0.16640	1.23901
PMS_CNN-Combine	**103.041%**	0.57819	0.17843	1.54761
PMS_RNN-Long	30.680%	**0.57117**	**0.08210**	**1.38457**
PMS_RNN-Short	7.056%	0.44713	0.20648	1.04442
PMS_RNN-Combine	**37.735%**	0.43116	0.09028	1.32291

while the short model (green curve) fluctuated. Despite similar profitability, the short model was subject to greater fluctuations and greater risk.

The second row in Table 4 presents the indicators of PMS_RNN when applied TW50. PMS_RNN-Long achieved better results than that of PMS_RNN-Short regardless of the indicator, which indicated that the portfolio could achieve higher profitability with lower risk. The Sharpe ratio and PF of PMS_RNN-Long (57.117% and 1.38457) are superior to those of the PMS_RNN-Short model (12.404% and 0.34015). In terms of risk, the MDD of the long model was 60% lower than that of the short model. Note that drawdown represents the scale of asset loss, such that a larger drawdown indicates a larger asset loss within a given trading period.

In the PMS_CNN and PMS_RNN systems, the short models always resulted in greater risk and MDD. These results are in line with empirical experience, in which a short position always entails greater risk. Our experiment results indicate that PMS_CNN-Combine outperformed all of the other systems in terms of total return, Sharpe ratio, and PF. In the end, investors are free to choose a system in accordance with their risk tolerance and expected returns.

4.3 Comparison of latest research

Three state-of-art and highly cited papers are compared with our systems and listed in Table 5. All papers adopt RL to form a portfolio on different markets, reward functions and network design. Jiang 2017 Conf. and Jiang 2017 arXiv the performance of designed CNN agent from [21, 22], Almahdi 2017 is the performance of Calmar-RRL from [23] (without stop-loss model and with transaction cost the same as ours). Since comparing models on different markets, we only list indicators that can be compared between trading performance on different datasets, namely the Sharpe ratio (measures the trade-off between profit and risk) and MDD (measures the risk).

Jiang's works focus on high-frequency trading and high volatility markets, aiming to pursue extraordinary profits (reword function of trading return), but ignores trading risks. Compared with Jiang's performance, since we focus on the stableness of trading (reward function of Sharpe ratio), our proposed systems have significantly better Sharpe ratio and MDD. As for Jiang's work, the transaction frequency of their model is much lower, with only 221 transactions in two years (about half of our transactions), and their goal is to optimize risk and return at the same time. Therefore, they have a higher Sharpe ratio, but due to the implementation of the EMN strategy in our systems, we still have the lowest drawdown risk, MDD.

4.4 Scalability of proposed PMS

In real-world application, industries have obtained varied stock attributes. Thus, the scalability of the proposed PMS in various industries (datasets) is then verified and shown in Fig. 5. Also, the trading performance with three benchmark strategies such as symmetrical, momentum, and contrarian are then compared Under the symmetrical strategy, a

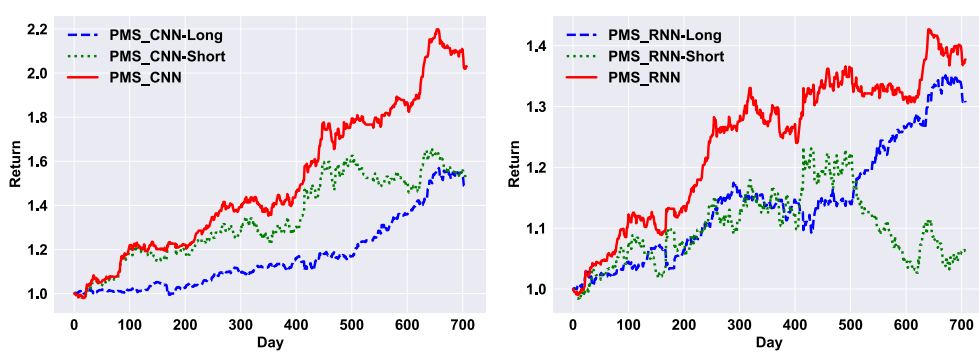

Fig. 4 Equity curves of the long, short, and combined models of the PMS. The left and right figures are for PMS_CNN (left) and PMS_RNN (right) systems

Table 5 Performance indicators of state-of-art researches

System	Sharpe	MDD
PMS_CNN	0.578	**0.178**
PMS_RNN	0.431	**0.090**
Jiang 2017 Conf.	0.037	0.296
Jiang 2017 arXiv	0.059	0.216
Almahdi 2017	**1.0273**	0.9550

The values in bold are the best performance for each column

uniform constant rebalanced portfolio (UCRP) [27] is created to rebalance a portfolio using a constant weight, such that long and short positions are nearly symmetrical. Under the momentum strategy, Winner [28, 29], investors long the stock with the biggest rise the day before, and short the stock with biggest decline the day before. Under the contrarian strategy, Loser [28, 29], the investor longs the stock with biggest decline the day before, and shorts the stock with biggest rise the day before.

We first trained the system on TW50 dataset, test and build the specific portfolios to various industries. The TW50 includes three major industries such as traditional, financial, and electronics. Stocks with ID of 1101, 1102, 1216, 1301, 1303, 1326, 1402, 2002, 2105, and 2207 are classified as the traditional industry. Stocks with ID of 2801, 2823, 2880, 2881, 2882, 2883, 2884, 2885, 2886, 2887, 2888, 2890, 2891, 2892, 5876, and 5880 belong to the financial industry. Stocks with ID of 2301, 2357, 2382, 2395, 4938, 3045, 4904, 2308, 2327, 2317, 2474, and 3008 are the electronics industry.

Figure 5 presents the equity curves of the proposed PMS (PMS_CNN and PMS_RNN) compared with the different benchmark strategies in various industries (datasets). The upper right and upper left subfigures are the trading results on TW50 and traditional industries. The lower right and lower left subfigures are the trading results on finance and electronics datasets. In traditional industries, PMS_CNN and PMS_RNN (red and orange curves) dominated the other curves throughout the entire period, indicating a gradual rise without significant drawdown. Both of them achieved outstanding profitability and outperformed the stand-alone benchmark strategies. In the financial industry, PMS_CNN and PMS_RNN are struggle in the first half, but generate stable profits in the second half, whereas the contrarian strategy Loser (blue curve) generate the largest profit. In the electronics industry, PMS_CNN and PMS_RNN achieved outstanding growth in the first half, but suffered huge drawdown and fluctuations in the second half; the overall result indicates the positive profit. Among four datasets, it

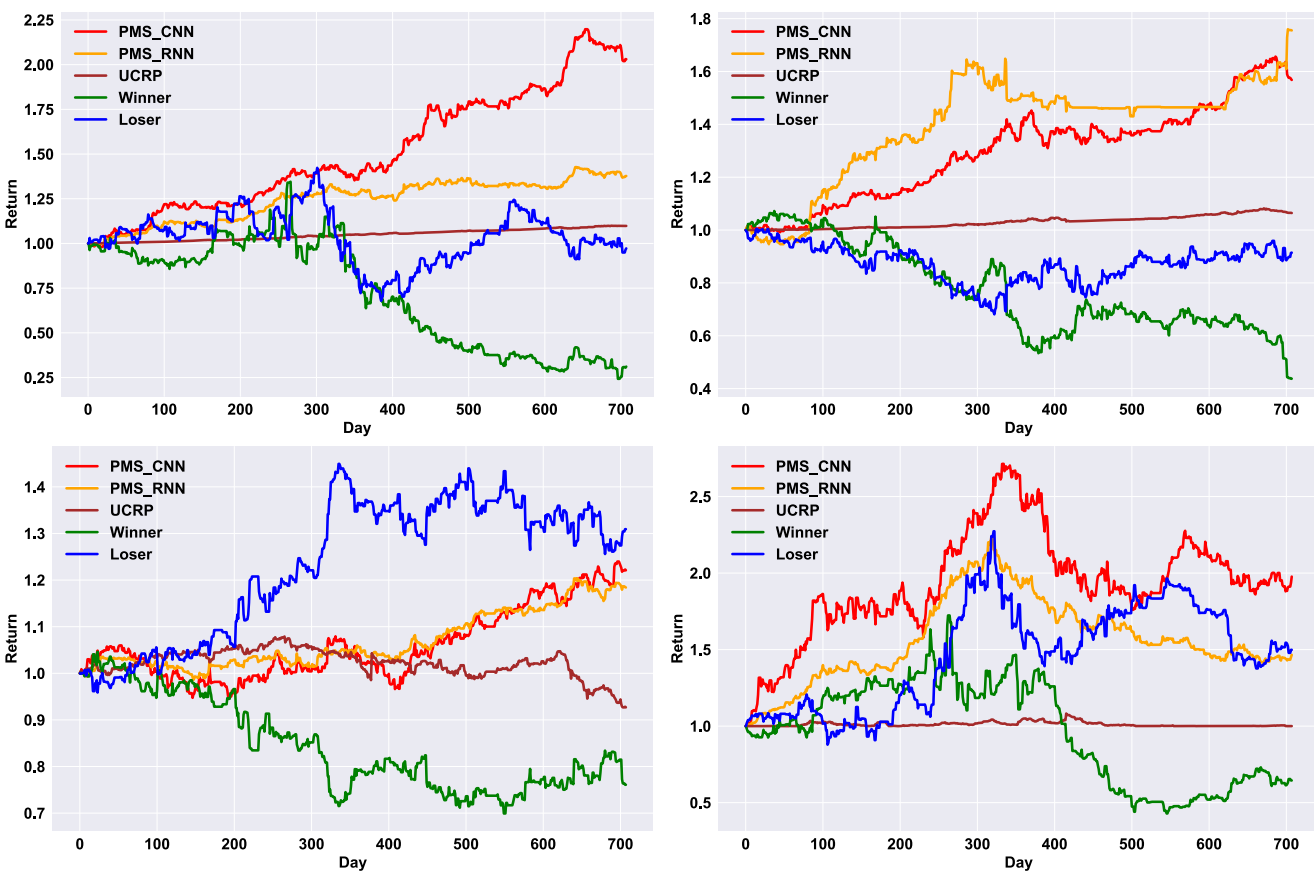

Fig. 5 Equity curves of strategies and systems in different industries

can be seen that the proposed PMS is suitable for the TW50 and traditional industries, whereas the contrarian strategy is more suitable for the finance industry. Note that the equity curves of UCRP were nearly horizontal in all industries, which showed the small profit with a tiny drawdown is then obtained.

5 Conclusions and discussion

Portfolio management is important to investors seeking to optimize commodity selection, position sizing, and resource allocation. Common portfolio strategies, such as MPT and Kelly, require forecasts of future outcomes as model inputs. Obtaining reasonable forecasts is not a trivial task and small deviations can greatly affect investment outcomes. Among the variant portfolio strategies, EMN is a hedging strategy that has proven effective in managing risk by balancing long and short positions. In this study, we established a novel Portfolio Management knowledge-based System (PMS) to support human decision-making and resource allocation in practical trading situations. The system is based on artificial intelligence (RL and NN), which gains trading and portfolio management knowledge by interacting with the market. We integrated two NN models with the proposed RL system to enable the extraction of spatial and temporal information and developed two reward functions to optimize profitability while taking risk into account.

In experiments, the proposed PMS with Sharpe ratio reward function outperformed the conventional return-based reward function, resulting in 39.0% higher profits and 13.7% less drawdown. The PMS_CNN outperformed PMS_RNN in terms of returns and Sharpe ratio; however, the risk of drawdown was 1.98 times higher. The proposed PMS outperformed existing benchmark strategies in terms of all measures in traditional industry or TW50 datasets. Although the PMS did not match the performance of conventional strategies when applied to the finance industry; however, it still achieved positive profits and Sharpe ratio with lower risk. Thus, the PMS is profitable and effective with lower investment risk among almost all datasets, and the novel reward function was also shown to enhance investment performance. In conclusion, the PMS can well support the resource-allocation in the empirical stock trading.

Acknowledgements This research is supported by Ministry of Science and Technology (MOST), Taiwan, and the project code is MOST 109-2218-E-001-004.

Funding Open access funding provided by Western Norway University Of Applied Sciences.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommonshorg/licenses/by/4.0/.

References

1. Gunasekarage A, Power DM (2001) The profitability of moving average trading rules in south asian stock markets. Emerg Mark Rev 2(1):17–33
2. Chong TTL, Ng WK, Liew VKS (2014) Revisiting the performance of macd and rsi sscillators. Journal of Risk and Financial Management 7(1):1–12
3. Tsai YC, Wu ME, Syu JH, Lei CL, Wu CS, Ho JM, Wang CJ (2019) Assessing the profitability of timely opening range breakout on index futures markets. IEEE Access 7:32061–32071
4. Syu JH, Wu ME, Lee SH, Ho JM (2019) Modified orb strategies with threshold adjusting on taiwan futures market. In: IEEE conference on computational intelligence for financial engineering & economics, pp 1–7
5. Reilly FK, Brown KC (2011) Investment analysis and portfolio management. Cengage Learning
6. Cooper RG, Edgett SJ, Kleinschmidt EJ (1999) New product portfolio management: practices and performance. Journal of Product Innovation Managemen 16(4):333–351
7. Harry M (1952) Portfolio selection. The Journal of Finance 7(1):77–91
8. Kelly JL (1956) A new interpretation of information rate. The Bell System Technical Journal 35(4):917–926
9. Michaud RO (1989) The markowitz optimization enigma: is optimized optimal? Financial Analysts Journal 45(1):31–42
10. Patton AJ (2009) Are market neutral hedge funds really market neutral? The Review of Financial Studies 22(7):2495–2530
11. Busoniu L, Babuska R, De Schutter B, Ernst D (2010) Reinforcement learning and dynamic programming using function approximators, vol 39. CRC Press, Boca Raton
12. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT Press, Cambridge
13. Haykin S (2007) Neural networks: a comprehensive foundation. Prentice-Hall, Inc., Englewood Cliffs
14. Gately E (1995) Neural networks for financial forecasting. Wiley, New York
15. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
16. Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117
17. Zaremba W, Sutskever I, Vinyals O (2014) Recurrent neural network regularization. arXiv:1409.2329
18. Lin JCW, Shao Y, Djenouri Y, Yun U (2021) ASRNN: a recurrent neural network with an attention model for sequence labeling. Knowl-Based Syst 212:106548
19. Qin Z, Yu F, Liu C, Chen X (2018) How convolutional neural network see the world-a survey of convolutional neural network visualization methods. arXiv:1804.11191
20. Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE (2017) A survey of deep neural network architectures and their applications. Neurocomputing 234:11–26

21. Jiang Z, Liang J (2017) Cryptocurrency portfolio management with deep reinforcement learning. In: Intelligent systems conference, pp 905–913
22. Jiang Z, Xu D, Liang J (2017) A deep reinforcement learning framework for the financial portfolio management problem. arXiv:1706.10059
23. Almahdi S, Yang SY (2017) An adaptive portfolio trading system: a risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. Expert Syst Appl 87:267–279
24. Sharpe WF (1994) The sharpe ratio. J Portf Manag 21(1):49–58
25. Bailey DH, Lopez de Prado M (2012) The sharpe ratio efficient frontier. Journal of Risk 15(2):3–44
26. Magdon Ismail M, Atiya AF (2004) Maximum drawdown. Risk Magazine 17(10):99–102
27. Kozat SS, Singer AC (2007) Universal constant rebalanced portfolios with switching. In: IEEE international conference on acoustics, speech and signal processing, vol 3, pp 1129–1132
28. Richards AJ (1997) Winner-loser reversals in national stock market indices: can they be explained? The Journal of Finance 52(5):2129–2144
29. Siganos A (2007) Momentum returns and size of winner and loser portfolios. Appl Financ Econ 17(9):701–708
30. Alexander C, Dimitriu A (2002) The cointegration alpha: enhanced index tracking and long-short equity market neutral strategies. ISMA Finance Discussion Paper
31. Jeng Y, Ton W, Lee KJ, Chuang HM (2006) Taiwan multi-factor model construction: equity market neutral strategies application. Managerial Finance
32. Pai GV, Michel T (2012) Differential evolution based optimization of risk budgeted equity market neutral portfolios. In: IEEE congress on evolutionary computation, pp 1–8
33. Van Otterlo M, Wiering M (2012) Reinforcement learning and markov decision processes. In: Reinforcement learning. Springer, pp 3–42
34. Malkiel BG (1989) Efficient market hypothesis. In: Finance, society for financial studies, pp 127–134
35. Sutton RS, McAllester D, Singh S, Mansour Y (1999) Policy gradient methods for reinforcement learning with function approximation. Advances in Neural Information Processing Systems 12:1057–1063
36. Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M (2014) Deterministic policy gradient algorithms. In: International conference on machine learning
37. Huang Z, van der Aalst WM, Lu X, Duan H (2011) Reinforcement learning based resource allocation in business process management. Data & Knowledge Engineering 70(1):127–145
38. Smart WD, Kaelbling LP (2002) Effective reinforcement learning for mobile robots. In: IEEE international conference on robotics and automation, vol 4, pp 3404–3410
39. Moody J, Saffell M (2001) Learning to trade via direct reinforcement. IEEE Transactions on Neural Networks 12(4):875–889
40. Lee JW (2001) Stock price prediction using reinforcement learning. In: IEEE international symposium on industrial electronics proceedings, vol 1, pp 690–695
41. Tsantekidis A, Passalis N, Tefas A, Kanniainen J, Gabbouj M, Iosifidis A (2017) Forecasting stock prices from the limit order book using convolutional neural networks. In: IEEE conference on business informatics, vol 1, pp 7–12
42. Chen JF, Chen WL, Huang CP, Huang SH, Chen AP (2016) Financial time-series data analysis using deep convolutional neural networks. In: International conference on cloud computing and big data, pp 87–92
43. Mesnil G, He X, Deng L, Bengio Y (2013) Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In: Interspeech, pp 3771–3775
44. Gao T, Chai Y (2018) Improving stock closing price prediction using recurrent neural network and technical indicators. Neural Comput 30(10):2833–2854
45. Wang J, Wang J, Fang W, Niu H (2016) Financial time series prediction using elman recurrent random neural networks. Computational Intelligence and Neuroscience 2016
46. Rout AK, Dash PK, Dash R, Bisoi R (2017) Forecasting financial time series using a low complexity recurrent neural network and evolutionary learning approach. Journal of King Saud University-Computer and Information Sciences 29(4):536–552
47. Lin CC, Chiang MH (2005) Volatility effect of etfs on the constituents of the underlying Taiwan 50 index. Appl Financ Econ 15(18):1315–1322
48. Engelberg JE, Reed AV, Ringgenberg MC (2018) Short-selling risk. The Journal of Finance 73(2):755–786

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Mu-En Wu is currently an Associate Professor in the Department of Information and Finance Management, National Taipei University of Technology, Taipei, Taiwan. He obtained his bachelor degree from the Department of Mathematics, National Tsing-Hua University, Taiwan, in 2002. He received his master degree from the Department of Applied Mathematics, National Chiao-Tung University in 2014. Then, he received the Ph.D. degree from the Department of Computer Science, National Tsing-Hua University in 2009. He has authored or co-authored over 90 papers in referred journals and conferences in the area of financial data analysis, information security, and cryptography. His research interests are mainly in the areas of money management, financial data analysis, and cryptography.

Jia-Hao Syu is currently a PhD student in the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan. He obtained his bachelor degree from the Department of Computer Science and Information Engineering, National Taiwan University, in 2019. His research interests are mainly in the areas of data science, artificial intelligence, and quant finance.

Jerry Chun-Wei Lin received his Ph.D. from the Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan in 2010. He is currently a full Professor with the Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, Bergen, Norway. He has published more than 400 research articles in refereed journals (IEEE TKDE, IEEE TCYB, IEEE TII, IEEE TITS, IEEE TIAS, IEEE TETCI, IEEE SysJ, IEEE SensJ, ACM TKDD, ACM TDS, ACM TMIS, ACM TOIT) and international conferences (IEEE ICDE, IEEE ICDM, PKDD, PAKDD). His research interests include data mining, soft computing, artificial intelligence and machine learning, and privacy preserving and security technologies. He is the Editor-in-Chief of the International Journal of Data Science and Pattern Recognition, the Guest Editor/Associate Editor of IEEE TFS, IEEE TII, ACM TMIS, IEEE Access, JIT, PlosOne, IDA, IJIMAI, AIHC, and EVIN. He is the Fellow of IET (FIET), senior member for both IEEE and ACM.

Jan-Ming Ho received his Ph.D. degree in electrical engineering and computer science from Northwestern University in 1989. He received his M.S. at Institute of Electronics of National Chiao Tung University in 1980 and his B.S. in electrical engineering from National Cheng Kung University in 1978. Dr. Ho joined the Institute of Information Science, Academia Sinica as an Associate Research Fellow in 1989, and was promoted to Research Fellow in 1994. In 2000-2003, he served as Deputy Director of the institute. In 2004-2006, he had served as Director General of the Division of Planning and Evaluation, National Science Council. He visited IBMs T. J. Watson Research Center in summer 1987 and summer 1988, the Leonardo Fibonacci Institute for the Foundations of Computer Science, Italy, in summer 1992, and the Dagstuhl Seminar on Applied Combinatorial Methods in VLSI/CAD, Germany, in 1993. Dr. Hos research interests cover the integration of theory and applications, including combinatorial optimization, information retrieval and extraction, multimedia network protocols, bioinformatics, and digital library and archive technologies. Dr. Ho also published results in the field of VLSI/CAD physical design. He has served as board member of several NPOs including Institute of Information and Computing Machinery (IICM), Frontier Foundation, Taiwan, Y.T. Lee Foundation Science Education for All, WuSanLien Foundation on Taiwanese History. He had served as President of IICM in 2007-2009 and President of Software Liberty Association Taiwan (SLAT) in 2004-2008.