Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

Adaptive Hybrid Optimized LSTM Models: A Novel Computational Framework for Algorithmic Financial Trading System

SU, Pei-chiang [1*]

[1] Carnegie Mellon University, USA

**** SU, Pei-chiang is the corresponding author, E-mail: peichias@alumni.cmu.edu***

Abstract: In our modern society, with the development of Internet and information system, pre-programing algorithmic trading strategies for online automatic trading has also flourished, especially in the rapidly fluctuating trading market. Using quantitative trading programs that can automatically trade in response to market conditions has attracted the attention of major financial institutions and governments in the financial market. How to use self-adaptive programs to automatically trade in the ever-changing financial market has become a popular research topic for the development and pursuit of all financial markets in recent years.

This research proposes an online self-adaptive trading algorithm that can be applied to financial markets such as stock market, currency markets, cryptocurrency markets, futures markets, etc. In the first part of the algorithm, the simplified swarm optimization will be used to optimize the parameters of the newly proposed flexible grid in this research. Then the data will be imported into the artificial neural network model for training in the latter part, helping the trading model automatically select the appropriate parameters for construction flexible grid corresponding to the market conditions.

The greatest contribution of the research is to provide a whole new trading algorithm that can adapt to the dynamic trading market, automatically make suitable adjustments to current trading strategy and place real-time orders. The algorithm controlling both profit and risk, which can seem like a balanced trading algorithm that are robust and profitable.

Keywords: Algorithmic Trading, Simplified Swarm Optimization, Artificial Intelligence, Deep Learning, Simplified Swarm Optimization, Artificial Neural Network, Grid Trading.

Disciplines: Computational Science. **Subjects:** Intelligent Computing.

DOI: https://doi.org/10.70393/6a69656173.323338 **ARK:** https://n2t.net/ark:/40704/JIEAS.v2n6a06

1 INTRODUCTION

1.1 RESEARCH BACKGROUND

Since the late 20th century, the continuous advancements in internet and computational technology have significantly transformed trading methods and strategies within financial markets. More financial institutions and major trading entities have gradually shifted from traditional in-person trading to electronic remote trading, even enabling automated trading through pre-programmed computer algorithms (algorithmic trading). Reports indicate that in the U.S. stock market, which is the most developed globally, 60-70% of transactions are executed through automated trading algorithms. Amid this wave of integrating computational technology into financial markets, the combination of quantitative trading and computational algorithms has experienced particularly rapid development [1].

The primary function of financial markets is to provide current market prices for traded assets, enabling participants to conduct transactions and potentially profit. Quantitative trading emerged in the stock market in the late 20th century and has been increasingly applied in recent years to automated trading systems for stocks, currencies, and futures. By analyzing large volumes of historical data, statistical and mathematical models predict high-probability future market scenarios and establish model-based trading logic for executing transactions, aiming for excess returns.

The defining feature of quantitative trading is its use of a fixed set of logical rules to achieve stable, above-average returns, making it widely applicable in stock and futures markets. Over time, quantitative trading has evolved into a comprehensive system, encompassing various models for different financial activities such as asset selection, portfolio allocation, and market timing.

Quantitative trading now considers an increasing array

of factors, including market structure, asset valuation, target asset development potential, and market sentiment. The information age has facilitated the collection and preservation of vast amounts of data, enabling quantitative trading models to analyze and supplement human limitations in processing large data volumes and make more objective, emotion-free decisions.

1.2 Research Motivation

With the evolution of international trade and the internet fostering an active global trading environment, transaction costs have decreased, and the scale and scope of trading have expanded. The relationship between the financial industry and technology has deepened, becoming more mature and widespread [2, 3]. Leveraging advanced computational technology as a tool for financial trading, and even as a critical factor in achieving trading success, has become a central research focus in the financial sector in recent years.

Early applications of electronic commerce focused on transforming physical trading formats into electronic forms, including recording and storing transaction information electronically for future reference and analysis. Basic services, such as account opening, fund transfers, and withdrawals, were automated to reduce human labor significantly.

In recent years, following initial successes in electronic finance, financial institutions have aimed to extend the role of computer systems from simple transaction processing to aiding complex decision-making processes. Notably, advancements in artificial intelligence (AI) have led to numerous breakthroughs [4-7]. The development of computer programs that simulate human decision-making and surpass human speed, scope, and depth has become a prominent area of development in the financial field.

Algorithmic trading emerged in the late 1980s and has evolved from human analysis or statistical decision-making followed by computer execution to increasingly sophisticated models that integrate decision-making processes within trading algorithms, enhancing their intelligence and automation. This field has become highly practical, drawing significant interest and investment from both governments and financial institutions [8].

To date, algorithmic trading research has yielded diverse and rich findings, such as using mean reversion to adjust stock investment weights [9], applying Long Short-Term Memory (LSTM) models for market trend prediction combined with grid trading methods (GTM) for currency trading [10], using box theory and support vector machines (SVM) to aid stock trading decisions [11], conducting forex trading based on Ichimoku Kinkohyo analysis [12], and replacing fixed time sequences with market trends as the unit for trading strategy execution [13]. It is evident that computer-assisted trading, and even fully automated decision-making and execution, has become an irreversible trend in financial development.

1.3 Research Objectives

As discussed earlier, with the rise of e-commerce and quantitative trading, economic activities are increasingly intertwined with technology, with many transactions relying on programs for completion. Looking forward, programs will not only aid in recording transactions and facilitating transfers but will also autonomously make trading decisions, including the timing and pricing of transactions. Given this background, the objectives of this paper are as follows:

- Propose a novel grid trading algorithm to address the shortcomings of existing grid trading models, such as premature entry and exit points.

- Develop an algorithm capable of adapting to changes in market conditions over time, adjusting its parameters to reduce the investor's effort in the trading process.

- Implement a logically structured trading model to minimize subjective and irrational decision-making by investors.

- Achieve a balance between risk and reward to secure a favorable return within a reasonable risk threshold.

1.4 Research Structure

Chapter 1 introduces the current state, significance, and impact of trading algorithms, outlining the background, motivation, and purpose of this study.

Chapter 2 discusses the definitions and characteristics of quantitative trading algorithms, provides an overview of the commonly used grid trading strategy, explains the operation logic of the Simplified Swarm Optimization (SSO) algorithm and its applications, and describes recent developments in artificial intelligence (AI), particularly in deep learning (DL), including the introduction and application of artificial neural networks (ANNs).

Chapter 3 details the model construction and training process, covering the operation of grid trading, the concept and development of a flexible grid, the optimization of flexible grid parameters under various conditions using SSO, and training neural networks using market data and optimized grid parameters. The trained model is then used to select parameters for different market scenarios and develop a flexible grid model for trading.

Chapter 4 presents an experiment using stock market data as a benchmark for quantitative trading, comparing the results with commonly used geometric and arithmetic grid models. Evaluation metrics include annualized percentage yield (APY), Sharpe ratio, and maximum drawdown.

Finally, the conclusion summarizes the model's performance and contributions and suggests potential future improvements. The overall research framework is illustrated in Figure 1-1.

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

2 LITERATURE REVIEW

2.1 QUANTITATIVE TRADING

Quantitative trading primarily involves formalizing investment decision-making processes into a structured, quantifiable operational logic. This approach eliminates emotional and subjective human interference and applies this logic to financial market activities to achieve returns above average market gains. The process of constructing models and rules can leverage large historical datasets for model validation, while the implementation phase employs computer programs to conduct automated financial activities.

The construction process for quantitative trading is often grounded in statistics, using vast historical data to calculate the most probable future outcomes and respond accordingly. The process typically involves metrics like probability, expected value, and standard deviation to evaluate key indicators such as investment returns, value-at-risk (VaR), and the Sharpe ratio, which are used for optimizing and assessing the performance of quantitative logic and models.

Quantitative trading can be categorized into several major strategies, described as follows:

Alpha Strategies: The core idea behind alpha strategies is to select superior assets (e.g., stocks or currencies) to go long while shorting underperforming assets. This type of strategy typically involves a lower trading frequency, associated with lower risk and relatively modest returns.

Arbitrage Strategies: The essence of arbitrage strategies lies in exploiting price differences for profit, such as discrepancies between identical assets in different markets or price differences between different assets within the same market. Another type of arbitrage capitalizes on price variations over time. For example, if an asset is priced at $90 with an expected value of $92 after one week, a quantitative trading model would generate a buy signal. This type of algorithm often includes a pricing model that is pivotal to the strategy's success. The basic assumption of such models is often mean reversion, where a significant deviation from the long-term average price triggers a trading signal, leading to potential profit. This strategy is commonly applied to assets with price volatility and is less effective for those in stable price conditions.

Mean Reversion Strategies: The mean reversion theory, derived from long-term market observations, posits that asset prices, despite short-term fluctuations, tend to revert to a stable mean over time. In quantitative trading, this strategy is implemented by using algorithms to identify markets with mean-reverting characteristics and issuing trading signals when deviations occur. A buy signal is generated when the price falls below the mean, while a sell signal is triggered when the price rises above the mean by a certain amount, creating profit opportunities through this logic.

Trend Following Strategies: Trend following is another mainstream strategy in quantitative trading and one of the most straightforward approaches. It aims to identify and follow market trends from their inception until their conclusion.

Trend Strategies: These strategies rely on collecting information such as news, events, or data that could impact the market to forecast future market directions and seek profit opportunities. Trend strategies are divided into event strategies and liquidity detection strategies.

Event Strategies use an event-triggered mechanism, collecting reports from major financial institutions and governments, company financial statements, statements, and international developments. Past market reactions to similar news are analyzed to predict future trends, enabling immediate responses before the general market.

Liquidity Detection Strategies identify the intentions of key market participants whose actions can significantly impact market prices. The strategy aims to detect potential buying or selling actions of influential traders and act preemptively to capitalize on such behavior.

Overall, quantitative trading enables the rapid processing of data collected from various systems in modern society, replacing traditional trading that relied on individual experience and limited information. By adhering to a consistent logic and mathematical model, it avoids emotional and external influences, reducing the likelihood of trading errors.

2.2 GRID TRADING

The grid trading method is defined as placing buy and sell orders at regular intervals within a specified price range, implementing a strategy of buying low and selling high. This simple yet effective principle allows traders to profit from price fluctuations and has been increasingly applied in various financial markets in recent years.

The success of a grid trading strategy depends on two main factors: price volatility and parameter settings. Price volatility encompasses both upward and downward movements: if prices rise continuously, holding a long position will be more profitable than grid trading; if prices

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

fall continuously, traditional spot trading yields limited gains, whereas short-selling futures can capture profits during declines. The most profitable scenario for grid traders is a market with both rises and falls, where higher volatility enhances profitability. Most markets exhibit significant price fluctuations and mean-reverting tendencies, making grid trading a widely adopted strategy.

The second key factor is parameter settings, which include average grid spacing, upper and lower grid limits, the number of grid levels, activation price, stop-loss, and take-profit points. These parameters must be tailored to the asset's price volatility, transaction costs, risk tolerance, and capital, directly impacting the strategy's performance.

Arithmetic and Geometric Grid Trading

The arithmetic grid setup is based on an initial price P_0, with the upper grid limit G_{ul}, lower grid limit G_{ll}, and the number of grid levels n defined. The grid spacing is then calculated using Equation 2.1.

$$G_s = \frac{G_{ul} - G_{ll}}{n} \qquad \text{Equation (2.1)}$$

After setting up the grid model, it serves as the basis for trading, facilitating the buying and selling of financial assets as prices fluctuate. This process is illustrated in Figure 2-1 below.

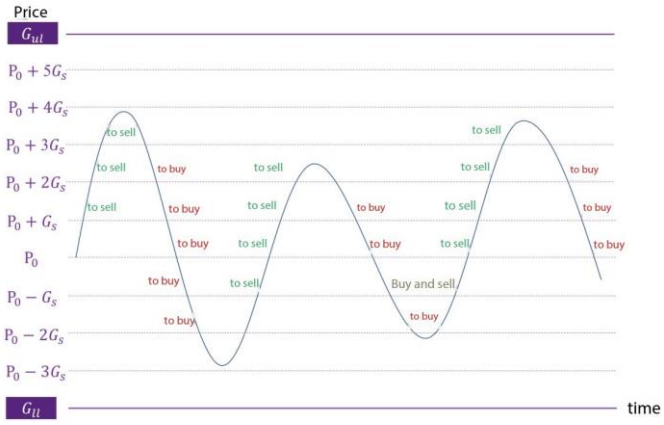

FIGURE 2-1 ILLUSTRATION OF ARITHMETIC GRID TRADING

As for the geometric grid, it uses a fixed ratio, and the grid spacing is calculated based on this ratio. The concept is illustrated in Figure 2-2 below.

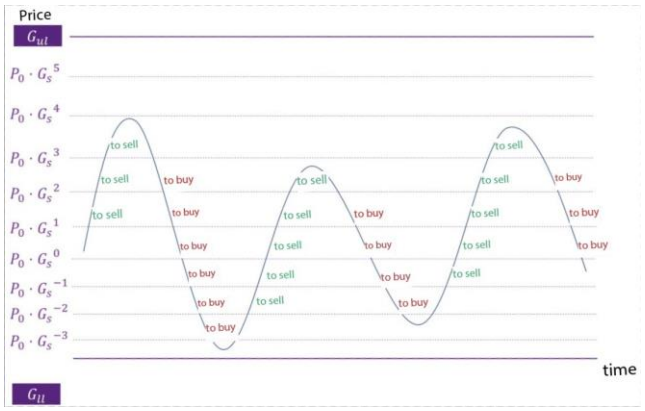

FIGURE 2-2 ILLUSTRATION OF GEOMETRIC GRID TRADING

Note: The detailed process and mechanism of grid trading will be explained in Research Methodology.

2.3 SIMPLIFIED SWARM OPTIMIZATION ALGORITHM

Particle Swarm Optimization (PSO) was proposed by Kennedy, Eberhart et al. [25] in 1950 to model the graceful yet unpredictable flight trajectories of bird flocks. It is a type of heuristic algorithm that can find relatively appropriate solutions to problems within a reasonable amount of time, and is often applied to complex problems where finding the optimal solution is not intuitive.

However, PSO faces issues such as premature convergence and poor performance when solving discrete problems. To address these shortcomings, Yeh [26] introduced the Simplified Swarm Optimization (SSO) algorithm in 2008, based on the core idea of simplicity. SSO improves the efficiency of problem-solving while simplifying the algorithm. It has since been widely used in solving problems across various fields [27-29]. One of the applications of SSO is to determine the optimal parameter combinations, where multiple parameters are simultaneously considered and adjusted to obtain the best-performing set of parameters. This approach has been used to fine-tune hyperparameter combinations in convolutional neural networks [30].

PSO's update mechanism requires two random values and three preset values (w、C_1 and C_2) to update, as shown in equations (2.2) and (2.3). The solution process is influenced by the Personal Best (pbest) and Global Best (gbest) values. SSO extends the concepts of pbest and gbest and incorporates random numbers [30], allowing the solution to escape local optima and enhancing the diversity of the solution.

$$v_{ij}^t = w \cdot v_{ij}^{t-1} + c_1 \cdot \rho_1 \cdot \left(P_{ij}^{t-1} - x_{ij}^{t-1}\right) + c_2 \cdot \rho_2 \cdot (g_j^{t-1} - x_{ij}^{t-1}) \qquad \text{Equation (2.2)}$$

$$x_{i_j}^t = x_{ij}^{t-1} + v_{ij}^t \qquad \text{Equation (2.3)}$$

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

The main difference between SSO and other heuristic algorithms lies in its unique update mechanism, which involves three particularly important parameter settings: C_g、C_p、C_w, where $C_g > C_p > C_w$. The update mechanism, as shown in Equation (2.4) and Figure 2-3, determines the next-generation solution x_{ij}^t based on the relationship between the random number ρ and the parameters C_g、C_p and C_w. This solution could be $pbest$、$gbest$ the current solution, or a random number.

$$x_{i_j}^{t+1} = \begin{cases} g_j \ , if\ \rho_{ij}^t \in [0, C_g) \\ p_{ij} \ , if\ \rho_{ij}^t \in [C_g, C_p) \\ x_{i_j}^t \ , if\ \rho_{ij}^t \in [C_p, C_w) \\ \text{x} \ , if\ \rho_{ij}^t \in [C_w, 1) \end{cases} \qquad \text{Equation (2.4)}$$

Let x_i^t be represented as $x_{i1}^t, x_{i2}^t, \dots, x_{ij}^t$, where x_{ij}^t indicates the ith solution in the tth iteration, consisting of j variables. ρ is a random variable uniformly distributed within the range [0,1]. The update mechanism is as follows: when ρ falls within [0, C_w), the variable x_{ij}^t retains the solution from the previous generation, x_{ij}^{t-1}; when ρ falls within $[C_w, C_p)$, it is replaced by the local best solution (pbest), which is the best solution for this variable in past iterations; when ρ falls within $[C_p, C_g)$, it is replaced by the global best solution (gbest), which is the best solution among all current solutions; and when ρ falls within $[C_g, 1)$, it is replaced by x, a random number generated within the variable's upper and lower bounds, to reduce the chance of being trapped in a local optimum and increase solution diversity.

In the update mechanism, different values of C_g, C_p, C_w are set according to various problems and scenarios, which greatly impact the quality of the final solution. In many past SSO-related studies, Orthogonal Array (OA) has been used to select suitable parameter configurations [29-31].

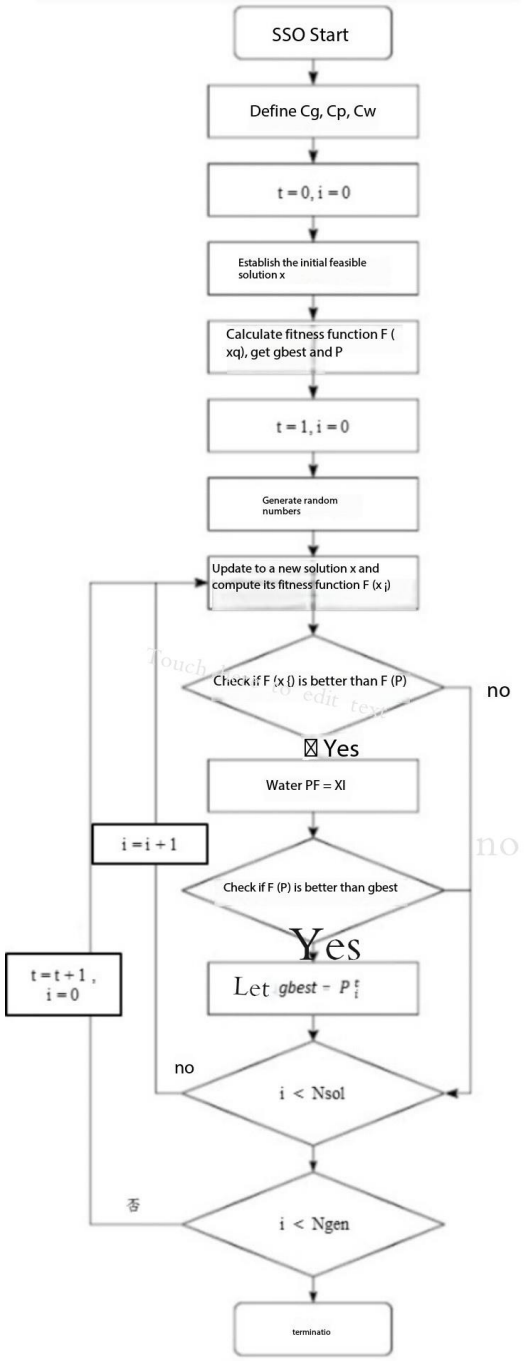

FIGURE 2-3 SSO UPDATE FLOWCHART [26]

2.4 DEEP LEARNING

Recently, the application of deep learning and artificial neural networks in financial activities has become increasingly diverse and widespread. The most common applications are in the prediction of stock markets, exchange rates, financial indices, and more [32]. Many studies are currently exploring various neural network models to assist in financial forecasting and decision-making [33-35], enabling more rational and accurate judgments.

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

2.4.1 Artificial Neural Networks

Artificial intelligence (AI) is a technological field that advanced countries around the world have been actively developing in recent years. Its main purpose is to imbue machines with the computational ability to think and make judgments similar to humans, enabling machines or computers to demonstrate human-like intelligence [36].

Within the field of AI is machine learning (ML), and deep learning (DL) is a branch of machine learning (the relationship between these three can be seen in Figure 2-4 below). In recent years, deep learning has been widely applied in fields such as medicine, industry, and transportation [37]. It facilitates tasks such as speech recognition and computer vision and is an algorithm structured on artificial neural networks (ANN) that learns from data features.

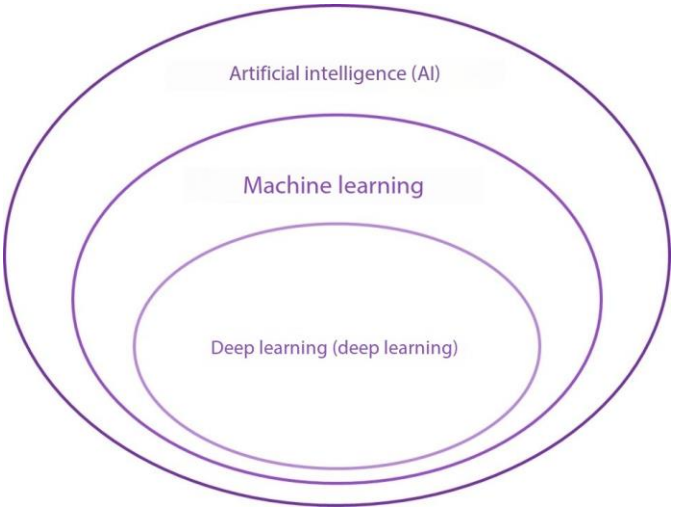

FIGURE 2-4 THE RELATIONSHIP DIAGRAM BETWEEN ARTIFICIAL INTELLIGENCE, MACHINE LEARNING, AND DEEP LEARNING

The computational architecture of artificial neural networks was first proposed by W.S. McCulloch and W. Pitts [38]. Subsequent improvements and modifications by numerous distinguished researchers [39, 40] have turned it into a well-known machine learning model within artificial intelligence. Artificial neural networks are models inspired by the structure of biological neural networks, aiming to mimic the pattern of neural transmission and brain function. Composed of numerous artificial neurons connected together, these networks can adjust themselves according to training data and learn in a semi-supervised or unsupervised manner [41].

The basic structure of artificial neural networks is divided into three main parts: the input layer, the hidden layer, and the output layer, as illustrated in Figure 2-5. Each node in the input layer corresponds to the input predictor variables, withx_j,j representing the node index in the input layer (e.g.x_1, the first node in the input layer). Each node in the output layer corresponds to the target variablesy_j, denoting the node index in the output layer (e.g.y_1, the first node in the output layer). Between these two layers are the hidden layers, where the nodes are represented by h_{ij} .with i indicating the layer number and j representing the node index within that layer (e.g. h_{23} , the third node in the first hidden layer). It is important to note that when i = 1 , this refers to the input layer at the front of the entire neural network. The hidden layers begin from i = 2.

What w_{i,j_1,j_2} represents is the weight between the j_1-th node in the upper layer and the j_2-th node in the i-th and i+1-th layer. In Figure 2-5, the weight between nodes h_{21} and h_{31} is represented by $w_{2,1,1}$. b_i represents the error between the i-th and i+1th layers and the error between the first hidden layer and the second hidden layer is denoted by b_2.

The number of hidden layers can be single layer or more, or even hundreds of layers, and the nodes between layers will be connected to each other, forming a complex neural network, and the reason why artificial neural networks are called deep learning is that.

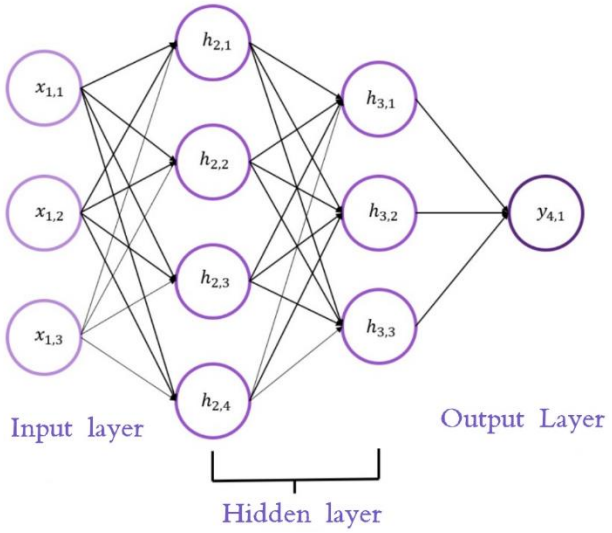

FIGURE 2-5 ARCHITECTURE OF A SIMPLE ARTIFICIAL NEURAL NETWORK

In addition to the nodes of the input layer, each node in the artificial neural network will be connected with several nodes in front of it (this model is also called fully connected neural network, FNN), and each node has a corresponding weight. The value of this node is the product of the value of the previous node and its weight, and then calculated by an activation function.

The node value of the hidden layer and the output value are calculated according to formula 2.5 and 2.6:

$$h_{i+1,j_2} = \Sigma_{j_1} h_{i,j_1} \times w_{i,j_1,j_2} + b_i \quad \text{Formula(2.5)}$$

$$y_{i+1,j_2} = \Sigma_{j_1} h_{i,j_1} \times w_{i,j_1,j_2} + b_i \quad \text{Formula(2.6)}$$

The incentive function is a nonlinear function. Without the introduction of the incentive function, ANN architecture

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

will carry out interactive operation in a simple linear way, and the relationship between nodes is linear. However, in practice, most of the relations between input and output variables are nonlinear. The model trained by ANN is difficult to have substantial meaning and practical problem-solving ability [42].

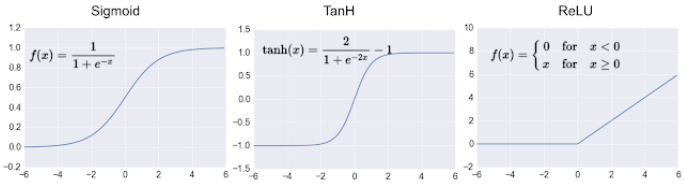

FIGURE 2-6 COMMON INCENTIVE FUNCTIONS

2.4.2 Backpropagation

In deep learning, backpropagation [43] is an extremely important key to model integrity and is often used to train and optimize artificial neural networks. By using backpropagation, the gradient of the loss function to the weight can be efficiently found, and then the gradient descent [44] is used to solve each weight. The "loss" in the loss function means "the residual difference between the actual value and the predicted value". The most important concept of backpropagation is to return the error value, so that the weight can use the error size to gradient descent method to obtain and update more suitable weights, further reduce the error and optimize the weight.

In the back propagation method, it can be divided into two stages: forward propagation and back propagation. For the process, please refer to Figure 2-3 below.

1. The first stage: forward propagation

The final output value is calculated by propagating from input layer, hidden layer to output layer with random output or all current weights.

2. The second stage: Back propagation

The difference between the network output value and the target value is calculated by the loss function, and the difference is returned as the basis of optimization, and the weight optimization is carried out by gradient descent method.

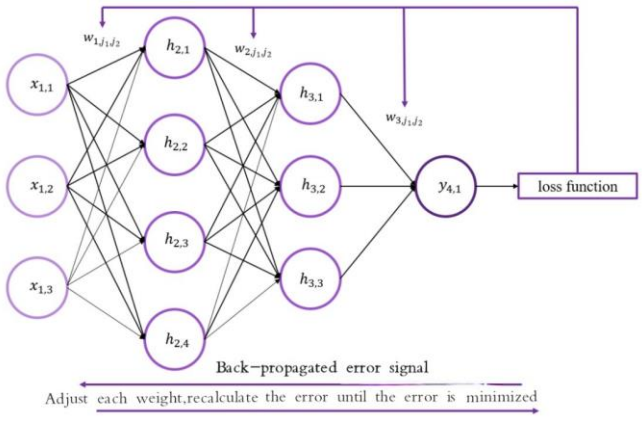

FIGURE 2-7 BACKPROPAGATION METHOD

2.4.3 Recurrent neural network

recurrent neural networks (RNNS) are a type of ANN that can deal with problems related to time series, that is, present or future situations are related to things that happened in the past. RNN stores the characteristics of the problem in the way of weights in each neuron in the network architecture, and uses the historical data that has happened in the past to predict the situation that will occur in the future. Its architecture is shown in Figure 2-8. Where A is the neural network, by reading the state x of the time t at the input layer, and then output a predicted value h_t from the output layer, cycling this process can make the information from the current time unit to the next time unit. The detailed calculation formulas of node values can be referred to equations 2.7 and 2.8[45].

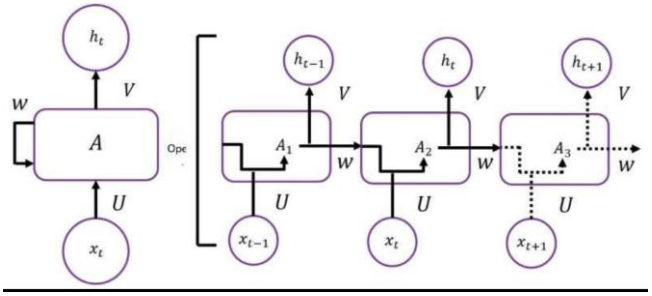

FIGURE 2-8 SIMPLE RNN MODEL ARCHITECTURE

Hidden layer node value calculation:

$$h(t) = f\left(Ux(t) + Wh(t-1)\right) \qquad \text{Formula(2.7)}$$

Output layer node value calculation:

$$y(t) = f\left(Vh(t)\right) \qquad \text{Formula(2.8)}$$

In the table,h(t) is the value of the hidden layer at time t, y(t) is the value of the output layer at time t, f (t) is the excitation function, U is the weight between the input layer and the hidden layer, W is the weight between the hidden layer and the hidden layer, V is the weight between the hidden layer and the output layer.

Because of the above characteristics, RNN has been widely used in speech recognition, text analysis, weather and stock price prediction and other fields, and is a well-known artificial neural network model in deep learning today.

2.4.4 Long short-term memory model

Although RNN can use previous historical data to predict future state performance, that is, it has information memory, but it is limited to short-term memory, and information that occurred long ago is often forgotten with the training of the model. Long Short-term Memory model (LSTM) is a model produced to improve the short-term memory of RNN, which is mainly composed of four units. They are composed of memory cell, input gate, output gate

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

and forget gate respectively. The architecture of these gates can be seen in Figure 2-9 [46].

The input gate controls whether the calculation is entered into the memory unit, the memory unit is responsible for storing the calculated value, the forgetting gate control clears the memory, and the output gate control outputs the calculation result (formula 2.9- Formula 2.14).

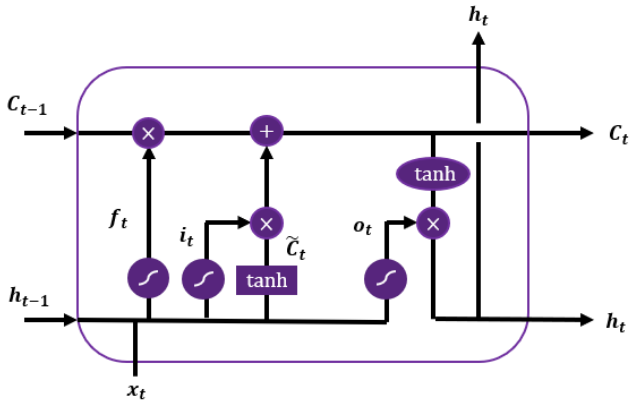

FIGURE 2-9 LSTM MODEL ARCHITECTURE [46]

Decide how much of the upper and local information needs to be forgotten:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad \text{Formula(2.9)}$$

Determine the information to be left and update the memory unit:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad \text{Formula(2.10)}$$

$$\tilde{C}t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c \quad \text{Formula(2.11)}$$

$$C_t = f_t \times C_{t-1} - i_t \times \tilde{C}_t \quad \text{Formula(2.12)}$$

Information that is finally decided to be exported

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad \text{Formula(2.13)}$$

$$h_t = o_t \times \tanh(C_t) \quad \text{Formula(2.14)}$$

2.5 RECENT RESEARCH OVERVIEW

Recent methods that apply machine learning to quantitative trading mainly include reinforcement learning, support vector machine, genetic algorithms, and random forest forest, artificial neural networks, etc.

In the field of neural networks, a large number of applications and studies have been conducted in the financial field. For example, the application of multi-layer neural networks and multi-layer perceptrons in gold and silver price prediction is superior to the traditional linear ARMA prediction model in results [47]. Use RNN for market prediction, train several different neural networks, analyze at the beginning, and select the neural network suitable for the current market situation for prediction [48]; LSTM is used to forecast the market, and the appropriate investment portfolio is constructed according to the result, and the result is better than the investment portfolio constructed by linear regression and SVM [49]. To construct decision models with enhanced learning and LSTM,input market financial information and output trading decisions appropriate to the current market situation [50]; To enhance learning and RNN to construct a trading model, the result is robust, can get a good return under relatively stable risk. [51]

In summary, this study uses a newly proposed grid trading model to obtain parameters suitable for various market conditions as input values and labels by a simplified group algorithm, and trains fully connected neural networks and long short-term memory models. Finally, a quantitative trading model can automatically calculate and adjust the best trading parameters for trading after inputting the existing market conditions.

The comparison method will not only be compared with the existing differential grid and equimetric grid in the market, but also with the technical analysis of the equilibrium chart proposed by Hosoda Goichi [12] and the prediction of market trends by the long short-term memory model combined with the grid trading method [10]. It is worth noting that the grid height and grid width set by the grid exchange here are set on the basis of the arithmetic grid.

3 RESEARCH METHODS

In Chapter 3, the following are introduced: The initial setting and operation mechanism of grid transactions (Section 3.1), the calculation of isometric and isometric grid parameters by SSO as a result comparison method (Section 3.2), the concept and setting method of elastic grid (Section 3.3) are introduced, and the optimal elastic grid parameters under different conditions are obtained by SSO in Section 3.4 as the training basis in deep learning in Section 3.5. Finally, a neural network can be trained to obtain excess returns from market fluctuations by inputting recent market information and automatically outputting optimal grid parameters for trading.

3.1 OPERATION OF GRID TRANSACTIONS

Following the basic concept of grid trading mentioned in 2.2, this section will explain in more detail how to practice grid trading in this study, including the initial parameter setting and calculation, as well as the subsequent operation mechanism and process.

3.1.1 Grid transaction initial Settings

Before running a grid trading model, there are five basic parameters that need to be set, namely, F_0 ,the total investment capital,P_0,the initial price, G_{ul},the upper limit of the grid, G_{ll},the lower limit of the grid and n, the number of grids. The total input capital and initial price are set as control variables in this study, that is, when comparing results, the total input capital and initial price used in any method will be set to the same fixed value and compared on the same basis.

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

Before the operation of the grid trading model, if the grid is an arithmetic grid, it is necessary to calculate the single-cell spread G_s, that is, the grid spacing, according to formula 2.1 in Chapter 2. G_s is a certain value in the isometric grid, but it is a fixed proportion in the isometric grid, for example, when$P_0 = 100$, $G_s = 1.1$,the value of the next cell of the grid is$P_0 \times {G_s}^1 = 110$, and the next cell is$P_0 \times {G_s}^2 = 121$, and this proportion can be calculated from the following formula 3.1.

$$G_s = \sqrt[n]{\frac{G_{ul} - G_{ll}}{G_{ll}} + 1} \qquad \text{Formula(3.1)}$$

In addition, we also need to calculate how much money should be used before the grid trading to buy the spot, that is, to buy the spot initially, to sell when the price rises for profit; And how much money must be kept on hand to buy spot when prices fall, where the initial cash holding is indicated. In other words, the total investment will be divided into two parts, which is expressed in Formula 3.2 below. The funds and spot held in each subsequent period are indicated, where j represents the phase j.

$$F_0 = S_0 + C_0 \qquad \text{Formula(3.2)}$$

To calculate the initial spot purchase S_0 and initial cash C_0, we also need to calculate the number of initial squares n_u and the number of initial squares n_l, the sum of the total number of grids is n , as shown in formula 3.3, and the values of the two can be obtained by formulas 3.4 and 3.5 respectively (here the arithmetic difference is taken as an example).

$$n = n_u + n_l \qquad \text{Formula(3.3)}$$

$$n_u = (G_{ul} - P_0) \div G_s \qquad \text{Formula(3.4)}$$

$$n_l = (P_0 - G_{ll}) \div G_s \qquad \text{Formula(3.5)}$$

After calculating the number of initial squares n_u and the number of initial squares n_l, formulas 3.6 and 3.7 are further used to calculate the initial spot purchase S_0 and initial cash C_0 · at the time, the single-cell trading volume G_v is still unknown.

$$S_0 = G_v \times n_u \times P_0 \qquad \text{Formula(3.6)}$$

$$C_0 = G_v \times [(P_0 - G_s) + G_{ll}] \div 2 \times n_l \qquad \text{Formula(3.7)}$$

Finally, by substituting formula 3.6 and 3.7 back to formula 3.2, the single cell trading volume can be obtained, and the calculation can refer to formula 3.8.

$$G_v = F_0 \div \{[(P_0 - G_s) + G_{ll}] \div 2 \times n_l + n_u \times P_0\} \qquad \text{Formula(3.8)}$$

Finally, formula 3.9 (isometric mesh) or formula 3.10 (isometric mesh) is used to calculate the price of each grid g_i, $i\ from\ 1\ to\ n$.

$$g_i = G_{ll} + G_s \times (\mathrm{i} - 1) \qquad \text{Formula(3.9)}$$

$$g_i = G_{ll} \times {G_s}^{(\mathrm{i}-1)} \qquad \text{Formula(3.10)}$$

At this point, we can use the above formula to calculate various parameters required for the operation of a grid trading model through the initial grid parameter setting, including: single-cell spread G_s, single-cell trading volume G_v, initial spot purchase S_0, initial cash C_0, initial number of upper squaresn_u, initial number of lower squares n_l and each cell price g_i. The symbol arrangement can be referred to the following table.

TABLE 1 GRID TRANSACTION MODEL SYMBOLS AND DEFINITIONS

Symbol	Definition
F_0	Initial total investment capital
P_0	The initial market price of a commodity
n	Grid total
G_{ul}	Upper limit of grid
G_{ll}	Lower limit of grid
G_s	The single-cell price difference is the numerical difference between the two grid lines in the case of an isometric grid and the multiple difference between the two grids in the case of an isometric grid, and the definition of an elastic grid is explained in detail in section 3.2.
G_v	Single cell volume, the amount of spot needed to sell or buy in a single cell, is a certain value.
S_0	Initial spot purchase volume
C_0	Initial cash
n_u	The number of squares at the beginning
n_l	The number of squares under the beginning
gi	Prices range from 1 to n

3.1.2 Grid transaction operation mechanism

The following illustrations detail how the grid trading model is updated and adjusted with the market price after the initial parameters are set, and how it is settled at the end.

When the grid is initially running, the above grid price will be placed on the sell order based on the current price, and the following grid will be placed on the pay.

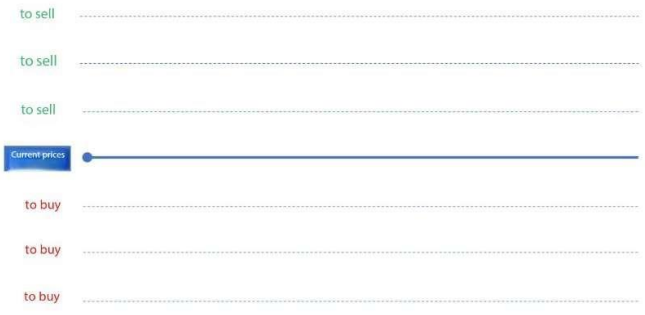

FIGURE 3-1 GRID TRANSACTION DIAGRAM (1)

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

1. if the price rises to meet the first grid line, make a sell action, update the stock and funds, and hang the bill in the original grid position.

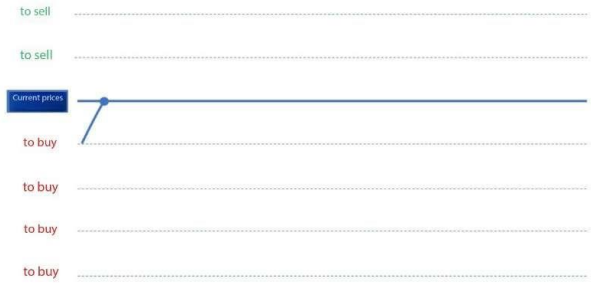

FIGURE 3-2 GRID TRANSACTION DIAGRAM (2)

2. If the price falls back to the original grid line, a buy action is made, the stock and funds are updated, and the sell order is placed at the original grid position.

FIGURE 3-3 GRID TRANSACTION DIAGRAM (3)

3. if the price continues to fall to the next grid line, continue to make a buying action, update the stock and capital, and put a sell order at the original grid position.

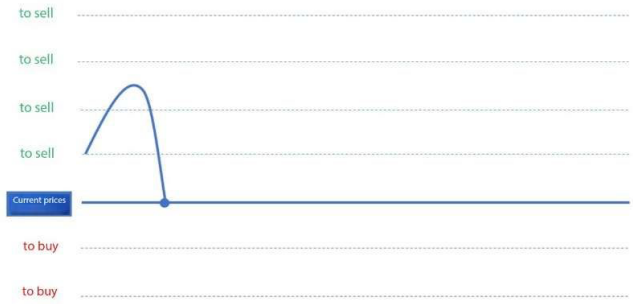

FIGURE 3-4 GRID TRANSACTION DIAGRAM (4)

4. The above mechanism continues to trade, as shown in Figure 3-5. Although the price has returned to the starting point of grid trading, it has successfully arbitraged 7 times, which is equivalent to obtaining 7 grid spread profits

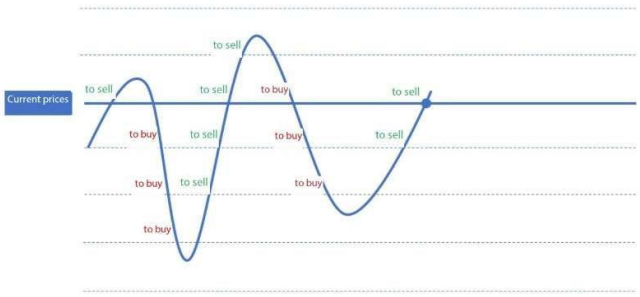

FIGURE 3-5 GRID TRANSACTION DIAGRAM (5)

When you want to close the grid trading model, there are two ways to end: one is to directly retain the current holding spot and funds, and the other is to sell the spot at the current price and convert it into cash. The former is recommended for use when the market price is low, and the latter is vice versa. In this study, the second method is used to close and settle the grid.

Pseudocode for Grid Transaction Operation I	
Input: Initial holding spot S_0, initial cash C_0, j period each period price $P_0 \sim P_j$	
Output: Final holding spot S_j、 Final cash C_j	
x from 0 to j	1
if $P_{x+1} > P_x$	2
set n = 0(Let n be the number of cells passed in this price change)	3
y from 0 to i	4
if $g_y < P_{x+1}$ *and* $g_y > P_x$	5
$n = n + 1$	6
$C_x = C_x + g_y \times G_v$	7
$S_{x+1} = S_x - n \times G_v$	8
if $P_{x+1} < P_x$	9
set n = 0(Let n be the number of cells passed in this price change)	10
y from 0 to i	11
if $g_y > P_{x+1}$ *and* $g_y < P_x$	12
$n = n + 1$	13
$C_x = C_x - g_y \times G_v$	14
$S_{x+1} = S_x + n \times G_v$	15

The overall process can be referred to the grid transaction operation flow chart in Figure 3-6 below, and after S_0 and C_0 entering the price of phase j, its update process is virtual code as above.

After entering the new price each time, updating the stock holding and funds holding, it is also necessary to do the action of reordering, and its virtual code is as follows:

Pseudocode for Grid Transaction Operation II	
if $P_{x+1} > P_x$	1
A1 = [ ] (Set an empty array to store prices that need to be updated)	2

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

y from 0 to i	3
find all $g_y \geq P_x$ and $g_y < P_{x+1}$, deposit A1	4
Remove the g_y recorded in A1 and place the sell order	5
if $P_{x+1} < P_x$	6
A2 = [ ] (Set an empty array to store prices that need to be updated)	7
y from 0 to i	8
find all $g_y \leq P_x$ and $g_y > P_{x+1}$, deposit A2	9
Remove the g_y recorded in A2 and place the sell order	10

If this price update does not encounter any pending order price, the update will not be triggered. It should be noted here that after entering the new price to complete this transaction, no buy or sell order will be placed on the last purchase price, that is, this price will not trigger any trading activity in the next update.

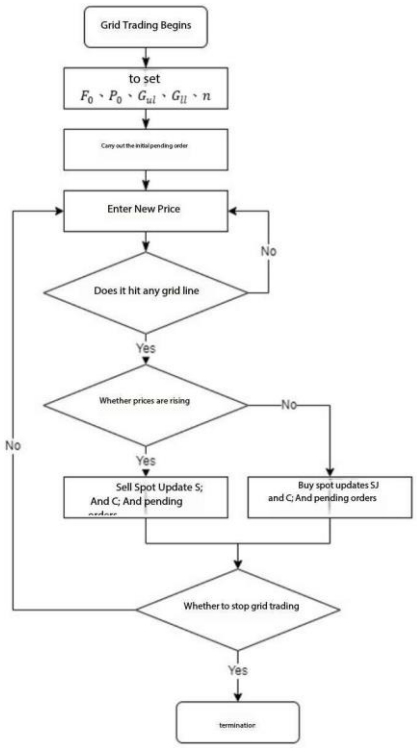

FIGURE 3-6 GRID TRANSACTION FLOW CHART

3.2 CONCEPT AND FRAMEWORK OF ELASTIC MESH

Based on the iso-difference and iso-ratio grid trading models used in today's financial markets, this study proposes a new and adaptive grid trading model-Elastic grid, hoping to construct a grid trading model that can adjust its parameters with market changes and adapt to various external conditions through a more elastic grid framework, combined with simplified group algorithms and deep learning.

Under the structure of the arithmetic difference grid trading model, it can give full play to its maximum model benefits when the market is volatile. Even if the price returns to the origin, it can still arbitrage from the volatile market situation. The main basis is to adopt the average price regression strategy mentioned in Chapter 2 quantitative trading (refer to section 2.1). On the other hand, equal-ratio grid trading can obtain better returns in the volatile rise. The concept combines the two strategies of average price regression and trend following. The detailed grid architecture can be seen in section 2.3.

The elastic grid proposed in this study captures the advantages of the isometric grid and the isometric grid at the same time, hoping to outperform the traditional grid trading architecture no matter the market is moving sideways, rising or even falling.

During the initial setting of the elastic grid, it is also necessary to set its total investment capitalF_0, initial priceP_0, upper and lower bounds of the gridG_{ul}, G_{ll}, and the number of initial upper and lower squaresn_u,n_l. However, the initial setting of the elastic grid is different from other models in two main aspects:

1. The number of initial upper squares n_u and lower squares n_l in formula 3.3 is no longer calculated by formula 3.4 and 3.5, but can be set initially.

The grid is divided into two parts with the initial price P_0 as the boundary. The upper part and the lower part can each set the number of grids, and have their own grid spacing ratio, The upper grid spacing ratio is G_{su} · the lower grid spacing G_{sl},it should be noted that G_{su} is a number greater than 0 and less than 1, and G_{sl} needs to be a number greater than 1. Its feature is that the upper part of the grid spacing, the higher the price will become smaller and smaller, that is, the transaction frequency will become more and more frequent; Similarly, the spacing of the bottom half of the grid will become smaller and denser as the price is lower。

For details, see Formula 3.11 and 3.12. For the overall architecture, see Figure 3-7 below.

$$G_{su} = \frac{1}{\sqrt[n_u]{\frac{G_{ul} - P_0}{P_0} + 1}} \quad \text{Formula(3.11)}$$

$$G_{sl} = \sqrt[n_l]{\frac{P_0 - G_{ll}}{G_{ll}} + 1} \quad \text{Formula(3.12)}$$

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

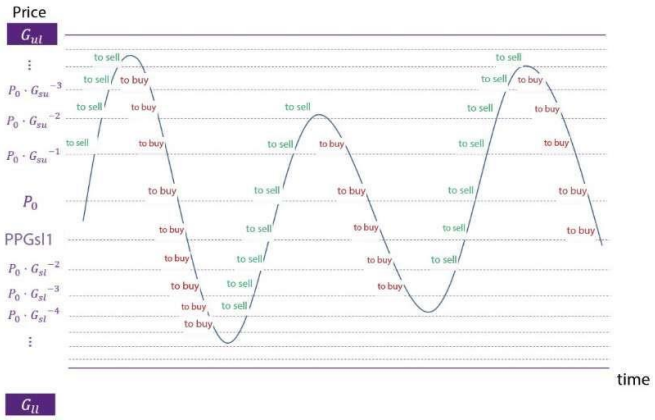

FIGURE 3-7 SCHEMATIC DIAGRAM OF ELASTIC GRID TRANSACTION MODEL

The core concept in the design process of elastic mesh is that it is hoped to perform more frequent selling actions at high prices, and buy cheaper spot more frequently at lower prices. In the middle section of the model, the elastic grid has a relatively similar architecture to the isometric mesh, and the lower part of the model is similar to the isometric mesh. In addition, the upper part of the model is different from the common isometric isometric mesh trading model, and the grid density is higher when the price is higher. This model is built to capture the advantages of arithmetic and arithmetic ratios, and to outperform the current model regardless of whether the market moves in a volatile, continuous rise or fall.

However, in order to give full play to the advantages of the elastic grid model architecture, good parameter setting and self-adjustment in accordance with the market situation are crucial decisive factors in determining the success or failure of the model. Therefore, the simplified group algorithm will be used to determine the parameters suitable for the elastic grid in different situations (the process will be detailed in section 3.3). The combination of the calculated parameters and the corresponding market situation are input into the artificial neural network for model training (the process is detailed in section 3.4). Finally, we will produce a trained deep learning model that can be automatically adjusted to the mesh parameters most suitable for the current market situation by simply inputting the current market situation.

3.3 TO SIMPLIFY THE GROUP ALGORITHM TO FIND THE OPTIMAL PARAMETERS

Based on the elastic grid constructed in the previous section, this section uses the simplified population algorithm to adjust the optimal solution of the elastic grid under different market conditions. In Section 3.4, the market conditions and optimal parameters are input into the artificial neural network as a training set for model training. For details, see Figure 3-8 below.

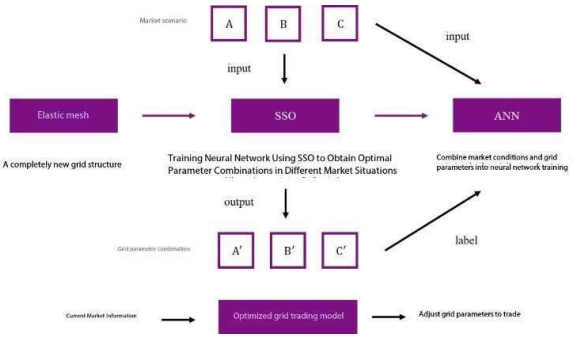

FIGURE 3-8 SCHEMATIC DIAGRAM OF RESEARCH PROCESS

3.3.1 Target type and limit type

The main goal of this research is to maximize the return on investment. Therefore, in the simplified group algorithm, the final goal formula is set as follows after the market investment commodity price is updated in phase j:

$$\max \quad S_j \times P_j + C_j \qquad \text{Formula(3.13)}$$

Where S_j represents the spot amount held in the last period, $P_j$is the commodity market price at the last period, C_j is the fund held at the last period. After S_0 and C_0entered through the price of period j , the updating process can be referred to section 3.1.2.

In addition, in the elastic grid trading model, it is necessary to ensure that the profit of each transaction is greater than the transaction cost $h\%$ (usually the transaction procedure rate), so the following restriction pairs must be set, from 0 to n:

$$s.t. \quad g_{i+1} - g_i > h\% \times g_{i+1} \qquad \text{Formula(3.14)}$$

The calculation can refer to formula 3.9 and Formula 3.10. After substituting, it can be seen that this restriction formula is actually a restriction on grid parameters and valuesG_{ul} , G_{ll} , n_u and n_l.

On the other hand, we will also make additional restrictions on the upper and lower limits of the gridG_{ul} , G_{ll} , n_u and n_l, and, from 0 to j:

$$s.t.\,(Experiment\ 1) \quad P_0 \times 105\% < G_{ul} < P_0 \times 130\% \qquad \text{Formula(3.15)}$$

$$P_0 \times 70\% < G_{ll} < P_0 \times 95\% \qquad \text{Formula(3.16)}$$

$$s.t. \quad (Experiment\ 2)\ P_0 \times 105\% < G_{ul} < P_0 \times 150\% \qquad \text{Formula(3.15)}$$

$$P_0 \times 50\% < G_{ll} < P_0 \times 95\% \qquad \text{Formula(3.16)}$$

$$10 < n_l < [P_0 \times 100/(maximum\ P_x \times 1.3)] - 10 \qquad \text{Formula(3.18)}$$

The above four restrictions are to preserve the error space, avoid the subsequent artificial neural network training overfitting, resulting in the price easily exceed the upper and lower limits of the grid and the loss of arbitrage opportunities,

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

and control the parameters in a reasonable range.

3.3.2 Uncoding Mode

As mentioned in section 3.3.1, the construction of a grid transaction model must be set by the total investment capitalF_0, initial priceP_0, grid upper boundG_{ul}, grid lower bound G_{ll} and grid number. In this study, $F_0 and P_0$ are control variables, so the parameters to be solved by SSO are respectivelyG_{ul}, G_{ll} and n. In elastic grids, grid number can be divided into upper half grid number n_u and lower half grid number n_l.

The solution range can be referred to the table below, where P_0 is the starting price of the commodity.

TABLE 2 UPPER AND LOWER BOUNDS OF GRID TRANSACTION PARAMETERS

Use range	parameter	variable upper bound	variable lower bound
Experiment 1	Grid upper bound G_{ul}	$P_0 \times 130\%$	$P_0 \times 105\%$
	Grid lower bound G_{ll}	$P_0 \times 95\%$	$P_0 \times 70\%$
Experiment 2	Grid upper bound G_{ul}	$P_0 \times 150\%$	$P_0 \times 105\%$
	Grid lower bound G_{ll}	$P_0 \times 95\%$	$P_0 \times 50\%$
Universal	number of upper grids n_l	$[P_0 \times 100 /(maximum\ P_x \times 1.3)] - 10$	10
	Lower grid numbe n_u	$[P_0 \times 100 /(maximum\ P_x \times 1.3)] - 10$	10

In this study, x_1 is set as the upper bound of the grid G_{ul}, $x_2$is set as the lower bound of the grid G_{ll}, x_3 is set as the number of the upper grid n_l and x_4 is set as the number of the lower grid n_u.

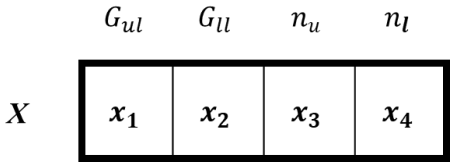

FIGURE 3-9 CODEC STRUCTURE

For example, when X = (150, 100, 10, 30), the upper bound of the grid is 150, the lower bound of the grid is 100, the number of upper grids is 10, the number of lower grids is 30, and the total number of grids is 40.

3.3.3 Update mechanism parameter setting and scope

In this study, SSO is used for solving, ρ_{ij}^t is a random number bounded between 0 and 1, and the solution of the problem will be updated according to this number: when ρ_{ij}^t is between 0 to C_g, gBest will be the solution of $x_{i_j}^{t+1}$; when ρ_{ij}^t is between $C_g$andC_p, pBest is taken as the solution; When ρ_{ij}^t falling between C_p and C_w · $x_{i_j}^{t+1} will\ renew\ to\ x_{i_j}^t$,It's the previous generation; When $\rho_{ij}^t$is between $C_w$and 1, a random new solution is randomly generated between the upper and lower bounds. The update mechanism can be referred to Formula 2.4.

The following table is a list of symbols and definitions required for SSO operation:

TABLE 3 SSO SYMBOLS AND DEFINITIONS

Symbol	Definition
N_{var}	The number of variables, in this study, is the upper bound of the grid G_{ul}、the lower bound of the grid G_{ll} *and the grid number* n。
N_{sol}	Solution total
N_{gen}	The maximum number of iterations, that is, the termination condition of the algorithm, and the appropriate set value should be able to converge to an appropriate value in a finite time.
X_i^t	X_i^t = ($G_{uli}^t, G_{lli}^t, n_i^t$)represents the i th solution in the t iteration, where t = 1,2,…, N_{gen} , i = 1,2,…,N_{sol}。
$\widehat{G_{ul}^p}, \widehat{G_{ul}^g}, \widehat{G_{ll}^p}, \widehat{G_{ll}^p}, \widehat{n^p}, \widehat{n^g}$	pBest and gBest of each variable in the update process.
C_g, C_p, C_w	The three key parameters used to determine the update value in SSO can be adjusted accordingly for different situations.
LB	LB = (lb_1 , lb_2, … , $lb_{N_{var}}$)is the lower bound of each variable, i.e.$x_{ij}^t \geq lb_j$。
UB	UB = (ub_1 , ub_2, … , $ub_{N_{var}}$)is the lower bound of each variable, i.e.$x_{ij}^t \leq ub_j$。

3.4 TRAINING ARTIFICIAL NEURAL NETWORK TO AUTOMATICALLY ADJUST ELASTIC MESH PARAMETERS

After using SSO to obtain the optimal grid configuration in various market conditions, the market conditions and calculated grid parameters are used as the training problems and solutions of artificial neural network training. One of the market conditions is interpreted in terms of the following values, which are entered into the neural network: period high,

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

period low, average market price, average volume, price change (starting price minus final price), volume change (starting volume minus final volume), price standard deviation, and volume standard deviation.

At the output end of the neural network, the relevant parameters needed to construct a grid transaction model are output, including: upper grid number, lower grid number, upper grid bound, and lower grid bound. The specific artificial neural network architecture can be referred to the following figure.

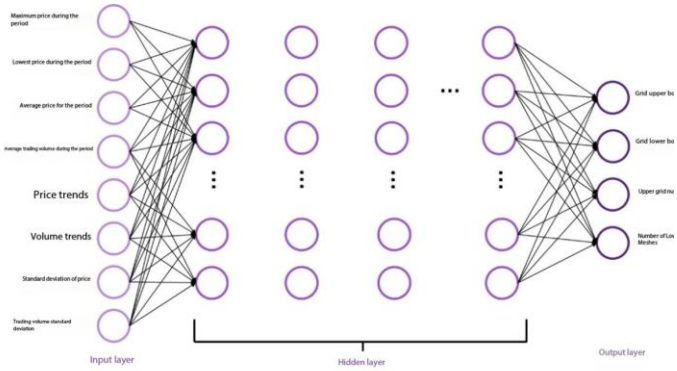

FIGURE 3-10 ARTIFICIAL NEURAL NETWORK ARCHITECTURE

In the process of neural network training, the weights in the artificial neural network nodes will self-adjust through the error value calculated by the loss function until they converge to a state with the least error, that is, the training is completed. Then we can use this trained neural network to input the recent market state and trend, generate the best grid trading model parameters, and automatically construct an elastic grid model for market trading activities.

And in this study, we're going to use two kinds of neural networks for training, They are Fully connect Neural Network (FNN) and Long Short-Term Memory (LSTM), which are often used to predict time series in recent years. To observe and compare which neural networks have better performance in learning mesh parameters.

Due to the different architecture of the model, the data processing of the fully connected neural network model will be rearranged too randomly, so that the pairing of market conditions and corresponding parameters will be trained as a single case, rather than as a time series problem. In the case of long short-term memory model data, since the model is born to deal with time series problems, the data will be input and trained in a chronological manner when the data is input. It is also possible to observe here which of the two data processing methods is better in terms of problem solving results.

4 RESEARCH RESULTS

The data set used for verification and comparison in this study is the Standard & Poor's 500 index, 500), the NASDAQ Composite, the Dow Jones Industrial Average, DJIA, Euro Stoxx 50 and Shanghai Composite, a total of five broad market indexes from 2011 to 2022.

4.1 THE PERFORMANCE OF ELASTIC MESH IS VERIFIED WITH FIXED PARAMETERS

First, based on the same number of grids and the upper and lower limits of the grid, we compared the performance of the elastic mesh with the isometric mesh and the isometric mesh. The results can be seen in Table 4-Table 6. The grid number is calculated according to Formula 4.1, in order to be close to the real investment situation and ensure the adequacy of the use of funds. Each data set is divided into two groups according to the upper and lower bounds condition. The first group takes 1.3 times of the initial price as the upper limit of the grid, and 0.7 times of the initial price as the lower limit of the grid. The second group takes 1.5 times the initial price as the upper limit of the grid, and 0.5 times the initial price as the lower limit of the grid.

From the results in Table 4, it can be verified that under the fluctuation of five different composite indexes in ten years, trading with the same grid parameters shows the best performance in the elastic grid return rate, accumulated wealth and sharpe rate. On the whole, the elastic grid achieves relatively high return on investment and sharpe ratio because its trading structure appropriately delays the entry and exit time.

TABLE 4 COMPARISON OF RETURN RATE OF ELASTIC MESH WITH ISOMETRIC MESH AND ISOMETRIC MESH (FIXED PARAMETER VERSION)

Type	elasticity	equal difference	equal radio
S&P 500			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**31.692 %**	27.934 %	22.234 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**43.727 %**	39.556 %	30.174 %
Nasdaq 100			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**53.795 %**	49.261 %	40.369 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**71.888 %**	66.024 %	51.180 %
Dow Jones Industrial Average			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**21.935 %**	18.332 %	13.629 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times$	**33.541 %**	29.741 %	21.648 %

0.5)			
Euro Stoxx 50			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.601 %**	-4.122 %	-7.284 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**17.499 %**	12.520 %	7.087 %
Shanghai Composite			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**-33.904 %**	-38.944 %	-38.534 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**-12.955 %**	-19.936 %	-18.290 %

TABLE 5. COMPARISON OF ACCUMULATED WEALTH OF ELASTIC MESH WITH ISOMETRIC MESH AND ISOMETRIC MESH (FIXED PARAMETER VERSION)

Type	elasticity	equal difference	equal radio
S&P 500			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**13169**	12793	12223
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**14373**	13956	13017
Nasdaq 100			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**15380**	14926	14037
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**17189**	16602	15118
Dow Jones Industrial Average			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**12194**	11833	11363
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**13354**	12974	12165
Euro Stoxx 50			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**10060**	9588	9272
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**11750**	11252	10709
Shanghai Composite			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**6610**	6106	6147
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**8705**	8006	8171

TABLE 6 COMPARISON OF SHARPE RATIO OF ELASTIC MESH WITH ISOMETRIC MESH AND ISOMETRIC MESH (FIXED PARAMETER VERSION)

Type	elasticity	equal difference	equal radio
S&P 500			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.118**	0.103	0.092
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.160**	0.145	0.137
Nasdaq 100			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.187**	0.172	0.161
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.234**	0.219	0.215
Dow Jones Industrial Average			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.079**	0.065	0.054
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.119**	0.105	0.095
Euro Stoxx 50			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.002**	-0.011	-0.021
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.046**	0.033	0.023
Shanghai Composite			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**-0.082**	-0.095	-0.106
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**-0.030**	-0.047	-0.053

4.2 SSO PARAMETER SETTINGS

We then used part of the dataset to select the appropriate SSO parameters for subsequent experiments.

In the first group of experiments, C_g, C_p, C_w is configured as shown in Table 2, which means that the range of random number ρ is divided into four parts according to 7:1:1:1, and the part with a proportion of 7 is assigned to gbest, pbest , $x_{i_j}^{t+1}$and new solutions in turn in the experiment, so as to detect which solution has a more critical influence on generating the solution with better quality.

TABLE 7 BEST AND WORST SOLUTIONS UNDER DIFFERENT PARAMETER COMBINATIONS (EXPERIMENT 1)

(C_g, C_p, C_w)	Maximum proportional solution	ROI (max)	ROI (min)
(0.7,0.8,0.9)	gbest	69.74%	66.83%
(0.1,0.8,0.9)	pbest	69.62%	67.18%
(0.1,0.2,0.9)	$x_{i_j}^{t+1}$	69.69%	66.01%
(0.1,0.2,0.3)	new random	69.41%	66.69%

As can be seen from Table 7, when gbest is maximum, it can produce a solution with better quality and is robust at the same time, so the probability of gbest will be set to the maximum in the final parameter configuration.

After determining that gbest will be set as the maximum probability, then the range of random number ρ is divided into four parts according to 5:3:1:1. Since gbest has been determined as the key factor to produce a good quality solution in the previous step, the probability of taking gbest

Journal of Industrial Engineering and Applied Science
Journal Home: http://jieas.suaspress.org/ | CODEN: JIEAAE
Vol. 2, No. 6, 2024 | ISSN 3005-6071 (Print) | ISSN 3005-608X (Online)

as the solution is set to the maximum 0.5. The opportunity of 0.3 will be allocated to pbest x_{ij}^{t+1} and new solutions in turn, and according to the experimental results, which solution is the second key factor to produce a solution with good quality can be determined, and the results can be referred to Table 8.

As can be seen from Table 8, when the new random solution is larger, it can produce a solution with better quality, and it can also be seen that its performance is robust in terms of the worst return, so the probability of the new random solution will be set as the second largest in the final parameter configuration.

TABLE 8 BEST AND WORST SOLUTIONS UNDER DIFFERENT PARAMETER COMBINATIONS (EXPERIMENT 2)

(C_g, C_p, C_w)	Maximum proportional solution	ROI (max)	ROI (min)
(0.5,0.8,0.9)	pbest	69.77%	66.22%
(0.5,0.6,0.9)	x_{ij}^{t+1}	69.77%	66.91%
(0.5,0.6,0.7)	new random	69.86%	67.83%

After it is determined that the new random solution will be set as the second highest probability, then the range of random ρ is divided into four parts according to 3:3:3:1, and the following steps are based on the previous two steps, and Table 5 is obtained. It can be seen that x_{ij}^{t+1} and pbest have little difference in the quality of the solution, so both probabilities will be set as the minimum

TABLE 9 BEST AND WORST SOLUTIONS UNDER DIFFERENT PARAMETER COMBINATIONS (EXPERIMENT 3)

(C_g, C_p, C_w)	Maximum proportional solution	ROI (max)	ROI (min)
(0.3,0.6,0.7)	pbest	69.85%	67.99%
(0.3,0.4,0.7)	x_{ij}^{t+1}	69.90%	67.81%

4.3 SSO SELECTION PARAMETERS WERE USED TO VERIFY THE PERFORMANCE OF ELASTIC MESH

After the SSO parameters were set, we connected the elastic grid, isometric grid and isometric grid to SSO respectively, and searched for solutions at the setting of 10 run, 20 generations per run, and 100 groups of solutions per generation. Similar to the previous elastic grid architecture verification, we also conducted two sets of experiments with the upper and lower bounds of different risk degrees, and the comparison of the results can be seen in Table 10-12.

TABLE 10 COMPARISON OF RETURN RATE OF ELASTIC MESH WITH ISOMETRIC MESH AND ISOMETRIC MESH (SSO PARAMETER EDITION)

Type	elasticity	equal difference	equal radio
S&P 500			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**82.859 %**	66.384 %	58.394 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**90.269 %**	77.198 %	60.774 %
Nasdaq 100			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**111.649 %**	66.384 %	82.662 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**127.268 %**	110.179 %	86.208 %
Dow Jones Industrial Average			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**74.509 %**	60.591 %	51.893 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**80.559 %**	70.063 %	54.760 %
Euro Stoxx 50			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**77.220 %**	56.650 %	48.365 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**88.844 %**	72.293 %	55.856 %
Shanghai Composite			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**58.389 %**	39.178 %	33.363 %
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**80.954 %**	58.934 %	43.639 %

From Table 10 to Table 12, it can be seen that elastic grid is still the best performing model in terms of return on investment, cumulative wealth and Sharpe ratio compared with arithmetic and arithmetic grid in solving grid transaction parameters by SSO. In addition, the grid transaction results after SSO connection are compared with the fixed parameter version, we can see that the return rate is significantly increased. In terms of the overall return on investment, the elastic grid is the best, the arithmetic difference is the second, and the arithmetic ratio is the worst.

TABLE 11 COMPARISON OF ACCUMULATED WEALTH OF ELASTIC MESH WITH ISOMETRIC MESH AND ISOMETRIC MESH (SSO PARAMETER EDITION)

Type	elasticity	equal difference	equal radio
S&P 500			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**18286**	16638	15839

$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**19027**	17720	16077
Nasdaq 100			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**21165**	16638	18266
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**22727**	21018	18621
Dow Jones Industrial Average			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**17451**	16059	15189
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**18056**	17006	15476
Euro Stoxx 50			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**17722**	15665	14837
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**18884**	17229	15586
Shanghai Composite			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**15839**	13918	13336
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**18095**	15893	14364

TABLE 12 COMPARISON OF SHARPE RATIO OF ELASTIC MESH WITH ISOMETRIC MESH AND ISOMETRIC MESH (SSO PARAMETER EDITION)

Type	elasticity	equal difference	equal radio
S&P 500			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.439**	0.350	0.351
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.442**	0.391	0.388
Nasdaq 100			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.498**	0.350	0.437
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.524**	0.464	0.462
Dow Jones Industrial Average			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.377**	0.310	0.301
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.412**	0.344	0.341
Euro Stoxx 50			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.300**	0.219	0.211
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.331**	0.265	0.257
Shanghai Composite			
$(G_{ul}, G_{ll}) = (P_0 \times 1.3, P_0 \times 0.7)$	**0.189**	0.134	0.131
$(G_{ul}, G_{ll}) = (P_0 \times 1.5, P_0 \times 0.5)$	**0.220**	0.179	0.172

4.4 TRAINING ARTIFICIAL NEURAL NETWORK TO AUTOMATICALLY ADJUST ELASTIC MESH PARAMETERS

After confirming the excellent performance of elastic grid with SSO in searching transaction parameters, we used elastic grid with SSO to record the best transaction parameters of each index for ten years, and obtained a set of transaction parameters every 30 days. In order to expand subsequent training and data, we set the moving pace of the model to 5. In each index, about 500 parameters were obtained respectively for training and verification of the artificial neural network, in which the data of the first nine years were used as the training set, and the data of the last year was used as the verification set.

In order to avoid affecting the model training results, the data of the training set in this study will be re-randomly ordered and then trained in the fully connected neural network. On the neural network of long short-term memory, because of the model design, the input training data should be sequential, so the training data should not be re-randomly sorted.

After several experiments of the architecture of fully connected neural network and long short-term memory neural network, the final architecture and the setting of hyperparameters are shown in Table 13 and Table 14.

The input parameters are: period maximum price, period minimum price, period average price, period average trading volume, difference before and after trading volume, difference before and after price, price standard deviation, trading volume standard deviation, and a total of eight variables to describe a market situation, and the output variables are set as the grid upper bound, grid lower bound, upper grid number and lower grid number used by grid exchanges.

Among them, the number of hidden layers is set to three layers, the optimizer is set to adam, which is common in recent years, has fast convergence speed and excellent solution finding performance, the excitation function uses sigmoid and relu, and the loss function uses mean squared error, which is commonly used in regression problems.

TABLE 13 HYPERPARAMETERS AND ARCHITECTURE-RELATED SETTINGS OF FULLY CONNECTED NEURAL NETWORKS

Item	set value
Number of input variables	8
Hidden layers	3
Number of output variables	4
Number of hidden layer nodes optimizer	500
optimizer	adam
Excitation function	sigmoid
Loss function	mean squared error

Iteration number	300
Lot size	40

TABLE 14 HYPERPARAMETERS AND ARCHITECTURE-RELATED SETTINGS OF LONG SHORT-TERM MEMORY NEURAL NETWORKS

item	set value
Number of input variables	8
Hidden layers	3
Number of output variables	4
Number of hidden layer nodes	256, 128, 64
optimizer	adam
Excitation function	relu
Loss function	mean squared error
Iteration number	300
Lot size	32

After training the neural network with the above Settings, the training and comparison results are presented in Table 15 and Table 16. From the results of Table 15 root-mean-square difference (Formula 4.1), it can be seen that except for the upper and lower limits of the grid in Nasdaq 100 index, LSTM has a small root-mean-square error, and the root-mean-square difference of the four output variables in other indexes is that FNN performs better.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \quad \text{Formula(4.1)}$$

TABLE 15 COMPARISON OF MEAN SQUARE ERROR BETWEEN FULLY CONNECTED NEURAL NETWORK AND LONG SHORT-TERM MEMORY MODEL

Types of neural networks	FNN	LSTM
S&P 500		
Mesh upper boundMesh G_{ul}	359.87	431.70
lower bound G_{ll}	244.51	329.58
Upper grid number n_l	14.39	15.87
Lower grid number n_u	17.55	21.02
Nasdaq 100		
Mesh upper boundMesh G_{ul}	2173.88	719.85
lower bound G_{ll}	1296.80	1078.14
Upper grid number n_l	11.53	17.20
Lower grid number n_u	15.68	23.83
Dow Jones Industrial Average		
Mesh upper boundMesh G_{ul}	2452.86	4944.37
lower bound G_{ll}	1418.22	4426.61
Upper grid number n_l	12.81	15.39
Lower grid number n_u	17.86	20.05
Euro Stoxx 50		
Mesh upper boundMesh G_{ul}	324.07	578.32
lower bound G_{ll}	184.36	326.22
Upper grid number n_l	11.49	14.21
Lower grid number n_u	14.11	17.03
Shanghai Composite		
Mesh upper boundMesh G_{ul}	255.34	304.74
lower bound G_{ll}	183.31	161.83
Upper grid number n_l	9.22	16.73
Lower grid number n_u	14.63	21.42

As shown in Table 16, the decision coefficient (R squared) of the two models, which is an indicator to measure the fit of the regression model, can also be interpreted as the interpretation degree of the model, and the calculation can refer to Formula 4.2.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = \frac{\Sigma_{i=1}(y_i - f_i)^2}{\Sigma_{i=1}(y_i - \bar{y})^2} \quad \text{Formula(4.2)}$$

In Table 16, it can be found that except for Nasdaq 100 index, FNN has better fit for the four output variables in most cases.

TABLE 16 COMPARISON OF DECISION COEFFICIENTS BETWEEN FULLY CONNECTED NEURAL NETWORKS AND LONG SHORT-TERM MEMORY MODELS

Types of neural networks	FNN	LSTM
S&P 500		
Mesh upper boundMesh G_{ul}	97.410%	90.490%
lower bound G_{ll}	99.586%	97.240%
Upper grid number n_l	94.972%	99.469%
Lower grid number n_u	99.392%	94.892%
Nasdaq 100		
Mesh upper boundMesh G_{ul}	96.415%	98.075%
lower bound G_{ll}	97.088%	99.915%
Upper grid number n_l	94.942%	99.635%
Lower grid number n_u	94.737%	98.019%
Dow Jones Industrial Average		
Mesh upper boundMesh G_{ul}	97.704%	95.970%
lower bound G_{ll}	99.771%	93.429%
Upper grid number n_l	99.855%	93.685%
Lower grid number n_u	99.559%	91.285%
Euro Stoxx 50		
Mesh upper boundMesh G_{ul}	96.183%	94.392%

lower bound G_{ll}	99.324%	98.063%
Upper grid number n_l	99.297%	96.858%
Lower grid number n_u	98.405%	99.760%
Shanghai Composite		
Mesh upper boundMesh G_{ul}	97.157%	96.577%
lower bound G_{ll}	96.749%	95.545%
Upper grid number n_l	93.165%	93.881%
Lower grid number n_u	99.661%	99.003%

The above results may be due to the fact that LSTM is better at dealing with continuous time series problems. However, in terms of changes in the stock market, prices are highly volatile, and the reasons for the fluctuations are complex, including changes in economic policies, international situations, wars, epidemics, etc. Overfitting problems often occur when recurrent neural networks are used to deal with finance-related problems. The loss value is low in the training set, but high in the test set, resulting in the phenomenon of low model generalization.

For FNN with relatively simple structure, we randomly sort the data in the training set and hide the time order of the data in order to improve the generalization of the model and solve the overfitting problem. In terms of the result theory, this approach has indeed achieved better performance in optimizing the grid parameters. In the following, we will use the four values of return of investment (ROI), maximum drawdown (MDD), volatility (volatility) and Sharpe ratio to examine the model performance of this study. In the method, there will be initial Buy and Sell (B&S), initial Sell and Buy (S&B), grid trading system robot (GTSbot) [10], Ichimoku Kinkohyo, and Ichimoku Kinkohyo. IK) [12], Flexible Grid with Fully Connected Neural Network (FG-FNN) trained by FNN, and elastic grid trained by LSTM

There are 9 methods: Flexible Grid with Long short term memory (FG-LSTM), isometric grid, isometric grid and elastic grid.

First of all, the formula of return on investment can be referred to Formula 4.3. Return on investment is the most common index to quantify investment performance in investment science, and the ratio between investment income and cost is calculated, which is usually presented in the form of annual return and total return. Transaction costs, including transaction fees, have been included in the calculation. Refer to Table 17 for the results.

TABLE 17 COMPARISON OF RETURN ON INVESTMENT

$ROI =$ (Net profit - cost of investment) $\times 100\%$ Formula(4.2)

Nasdaq	DJI	Euro Stoxx	Shanghai
1.519%	**9.880%**	28.340%	-15.581%
-1.519%	-9.880%	-28.340%	15.581%
4.680%	6.594%	1.191%	**-0.466%**
23.171%	7.143%	**32.087%**	-18.758%
11.733%	8.849%	12.977%	-3.125%
1.823%	2.940%	7.734%	-9.283%
-2.972%	3.480%	10.748%	-4.495%
-3.625%	3.276%	10.764%	-4.892%
-1.895%	4.270%	11.477%	-4.309%

In terms of return rate, FNN elastic grid can reach the top three highest returns in each index. Especially, it can be seen that FNN elastic grid has the best ability to control the overall loss under the trend of price decline.

The second thing to compare is the maximum pullback, that is, the amount of return that fell during the period of the most drastic decline in return in all the periods, which is one of the indicators to evaluate investment risk. The results can be seen in Table 18.

TABLE 18 COMPARISON OF MAXIMUM DECLINES

	S&P	Nasdaq	DJI	Euro Stoxx	Shanghai
B&S	7.960 %	12.570 %	7.290 %	4.029 %	8.529%
S&B	7.491 %	10.370 %	6.580 %	7.313 %	3.103%
GTSbot	**0.524 %**	**5.335 %**	**1.000 %**	**0.510 %**	**1.126%**
IWOC	10.345 %	12.421 %	8.606 %	5.467 %	7.640%
FG-FNN	6.455 %	5.596 %	7.491 %	2.034 %	4.105%
FG-LSTM	1.580 %	49.031 %	4.809 %	2.966 %	3.551%
Equal difference	5.252 %	8.594 %	4.078 %	1.513 %	4.074%
Equal ratio	5.243 %	10.096 %	4.057 %	1.529 %	4.059%
elasticity	4.737 %	8.780 %	4.163 %	4.144 %	4.208%

In terms of the maximum fall, it can be seen that the greater the rate of return on investment, the maximum fall is relatively high, verifying the theory of high risk and high return in investment science. There are two reasons for the performance of FNN elastic grid on the maximum decline: in the case of relatively high investment return rate, the risk will be relatively high, plus the parameters of each period are calculated by the neural network, if the parameters predicted in one period are particularly poor, it is likely to cause a large decline in a single period. Whether this maximum retreat can be regarded as a single condition of the outlier needs to be verified by volatility. In quantitative trading, the risk and stability of the model will be assessed through its volatility. The calculation method can be referred to formula 4.3, and the results can be referred to Table 19.

$$Vol = \sigma\,[return] \qquad \text{Formula(4.3)}$$

It can be seen from the table that the volatility performance of FNN elastic grid is among the top three in each index, that is to say, compared with other methods, the

performance of FNN elastic grid is very stable, and it can also be inferred that the performance of FNN elastic grid is poor in the maximum decline, which may only be a single event. In most cases, it is an investment model with robust performance. LSTM elastic grid is the best among all models in S&P and NASDAQ data sets, and it is a relatively stable investment model with less risk.

It can be seen from the table that the volatility performance of FNN elastic grid is among the top three in each index, that is to say, compared with other methods, the performance of FNN elastic grid is very stable, and it can also be inferred that the performance of FNN elastic grid is poor in the maximum decline, which may only be a single event. In most cases, it is an investment model with robust performance. LSTM elastic grid is the best among all models in S&P and NASDAQ data sets, and it is a relatively stable investment model with less risk.

TABLE 19 COMPARISON OF VOLATILITY

	S&P	**Nasdaq**	**DJI**	**Euro Stoxx**	**Shanghai**
B&S	0.06126	0.07671	0.03568	0.06221	0.01840
S&B	0.06126	0.07671	0.03568	0.06221	0.01840
GTSbot	0.01710	0.03589	**0.01795**	**0.00267**	**0.00190**
IWOC	0.07689	0.08398	0.04564	0.07337	0.03568
FG-FNN	0.01687	0.02076	0.02149	0.01730	0.01414
FG-LSTM	**0.01672**	**0.01637**	0.02203	0.01679	0.02030
Equal difference	0.02450	0.03579	0.02129	0.01786	0.01590
Equal ratio	0.02473	0.03629	0.02155	0.01828	0.01629
elasticity	0.02503	0.03665	0.02162	0.01862	0.01624

Finally, we can see the Sharpe rate, which is a model performance indicator to calculate the ratio between investment return and risk, and can also be interpreted as the return that can be exchanged for each unit of risk. The calculation method can be referred to Formula 4.4, and the results can be referred to Table 20.

TABLE 20 COMPARISON OF SHARPE RATES

$$Sharpe = \frac{return}{\sigma\,[return]}$$ Formula(4.3)

	S&P	**Nasdaq**	**DJI**	**Euro Stoxx**	**Shanghai**
B&S	2.349	0.198	2.769	4.556	-8.466
S&B	-2.34	-0.198	-2.76	-4.556	**8.466**
	9		9		
GTSbot	3.748	1.304	3.673	4.466	-2.446
IWOC	3.136	2.759	1.565	4.373	-5.258
FG-FNN	**6.828**	**5.651**	**4.119**	**7.499**	-2.210
FG-LSTM	2.774	1.114	1.335	4.606	-4.573
Equal difference	2.120	-0.830	1.634	6.017	-2.827
Equal ratio	1.834	-0.999	1.520	5.887	-3.004
elasticity	2.233	-0.517	1.975	6.165	-2.653

It can be seen from Table 20 that the performance of Sharpe rate of FNN elastic mesh is the best among all methods in the four indices; In a falling market, in addition to shorting, its sharpe ratio is also the largest. It can be seen that for each unit of risk, FNN elastic mesh gets the highest return.

5 CONCLUSION

In addition to proposing a new grid trading architecture, this study effectively improves the return on investment of the original model, improves the disadvantages of premature entry and exit, and uses simplified group algorithms and artificial neural networks to assist the parameter selection of grid trading, and gives the model the ability to adapt to the market.

In terms of model performance, we use five market indexes as validation data, which cover the United States, Europe and China. FNN elastic mesh has excellent performance in Sharpe rate, has excellent return on investment, and has model robustness, and can properly control the balance between risk and return.

The following are some directions for future research to be extended and improved:

Can analyze and study various market conditions (e.g. price continues to rise, price continues to fall, price 1. By judging the market situation, we can select the appropriate model for trading, so as to obtain better model performance.

2. In terms of maximum fall, there is room for improvement in the performance of the current model, that is, the current model will have a particularly poor performance in a few special cases, and how to avoid such a situation can be studied in the future.

3. In order to control risks, the current model controls the number of unilateral grids to at least 10 grids. If you expect to have a higher rate of return in the future, you can try to adjust the minimum number of grids.

Recently, in financial forecasting, Generative Adversarial Network (GAN) seems to be gradually

surpassing the long-term short-term memory model. GA4.N can be considered to improve the current shortcomings of the long-term short-term memory model in the future.

We demonstrate that quantitative trading, combined with artificial intelligence technology, can bring breakthroughs to the original trading model and adapt to a variety of rapidly changing market conditions. It is expected that there will be more relevant extended studies in the future, so that the trading model can be more accurately close to human decisions when making parameter adjustments, and even better than human judgments, so that the human impulse and market sentiment can be eliminated, and market trading decisions can be made rationally, so as to obtain better investment benefits.

ACKNOWLEDGMENTS

The authors thank the editor and anonymous reviewers for their helpful comments and valuable suggestions.

FUNDING

Not applicable.

INSTITUTIONAL REVIEW BOARD STATEMENT

Not applicable.

INFORMED CONSENT STATEMENT

Not applicable.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

CONFLICT OF INTEREST

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

PUBLISHER'S NOTE

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

AUTHOR CONTRIBUTIONS

Not applicable.

ABOUT THE AUTHORS

SU, Pei-chiang

Department of Electrical and Computer Engineering, Carnegie Mellon University, USA.

REFERENCES

[1] IOSCO, J., Regulatory Issues Raised by the Impact of Technological Changes on Market Integrity and Efficiency. Consultation Report, 2011.

[2] Allen, F., J. McAndrews, and P. Strahan, E-finance: an introduction. Journal of financial services research, 2002. 22(1): p. 5-27.

[3] Shahrokhi, M., E-finance: status, innovations, resources and future challenges. Managerial Finance, 2008.

[4] Turing, A.M., Computing machinery and intelligence, in Parsing the turing test. 2009, Springer. p. 23-65.

[5] Hutson, M., How researchers are teaching AI to learn like a child. Science, 2018. 10.

[6] Minsky, M. and S.A. Papert, Perceptrons: An introduction to computational geometry. 2017: MIT press.

[7] Silver, D., et al., Mastering the game of Go with deep neural networks and tree search. nature, 2016. 529(7587): p. 484-489.

[8] Aldridge, I., High-frequency trading: a practical guide to algorithmic strategies and trading systems. Vol. 604. 2013: John Wiley & Sons.

[9] Lai, Z.-R., et al., Reweighted price relative tracking system for automatic portfolio optimization. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018. 50(11): p. 4349-4361.

[10] Rundo, F., et al., Grid trading system robot (gtsbot): A novel mathematical algorithm for trading fx market. Applied Sciences, 2019. 9(9): p. 1796.

[11] Wen, Q., et al., Automatic stock decision support system based on box theory and SVM algorithm. Expert systems with Applications, 2010. 37(2): p. 1015-1022.

[12] Deng, S. and A. Sakurai. Short-term foreign exchange rate trading based on the support/resistance level of Ichimoku Kinkohyo. in 2014 International Conference on Information Science, Electronics and Electrical Engineering. 2014. IEEE.

[13] Ye, A., et al. Developing sustainable trading strategies

using directional changes with high frequency data. in 2017 IEEE International Conference on Big Data (Big Data). 2017. IEEE.

[14] Krauss, C., Statistical arbitrage pairs trading strategies: Review and outlook. Journal of Economic Surveys, 2017. 31(2): p. 513-545.

[15] Poterba, J.M. and L.H. Summers, Mean reversion in stock prices: Evidence and implications. Journal of financial economics, 1988. 22(1): p. 27-59.

[16] Hurst, B., Y.H. Ooi, and L.H. Pedersen, A century of evidence on trend-following investing. The Journal of Portfolio Management, 2017. 44(1): p. 15-29.

[17] Kampouridis, M. and F.E. Otero, Evolving trading strategies using directional changes. Expert Systems with Applications, 2017. 73: p. 145-160.

[18] MacKinlay, A.C., Event studies in economics and finance. Journal of economic literature, 1997. 35(1): p. 13-39.

[19] Hautsch, N. and R. Huang, On the dark side of the market: identifying and analyzing hidden order placements. Available at SSRN 2004231, 2012.

[20] Pardo, A. and R. Pascual, On the hidden side of liquidity. The European Journal of Finance, 2012. 18(10): p. 949-967.

[21] DuPoly, A. The Expert4x, No stop, Hedged, Grid Trading System and The Hedged, Multi-Currency, Forex Trading System. 2008

[22] Jegadeesh, N., Evidence of predictable behavior of security returns. The Journal of finance, 1990. 45(3): p. 881-898.

[23] Jegadeesh, N., Seasonality in stock price mean reversion: Evidence from the US and the UK. The Journal of Finance, 1991. 46(4): p. 1427-1444.

[24] Li, B., et al., PAMR: Passive aggressive mean reversion strategy for portfolio selection. Machine learning, 2012. 87(2): p. 221-258.

[25] Kennedy, J. and R. Eberhart. Particle swarm optimization. in Proceedings of ICNN'95-international conference on neural networks. 1995. IEEE.

[26] Yeh, W.-C., A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems. Expert Systems with Applications, 2009. 36(5): p. 9192-9200.

[27] Yeh, W.-C., Novel swarm optimization for mining classification rules on thyroid gland data. Information Sciences, 2012. 197: p. 65-76.

[28] Yeh, W.-C., W.-W. Chang, and Y.Y. Chung, A new hybrid approach for mining breast cancer pattern using discrete particle swarm optimization and statistical method. Expert Systems with Applications, 2009. 36(4): p. 8204-8211.

[29] Huang, C.-L., A particle-based simplified swarm optimization algorithm for reliability redundancy allocation problems. Reliability Engineering & System Safety, 2015. 142: p. 221-230.

[30] Yeh, W.-C., Orthogonal simplified swarm optimization for the series–parallel redundancy allocation problem with a mix of components. Knowledge-Based Systems, 2014. 64: p. 1-12.

[31] Yeh, W.-C., A novel boundary swarm optimization method for reliability redundancy allocation problems. Reliability Engineering & System Safety, 2019. 192: p. 106060.

[32] Huang, W., et al., Neural networks in finance and economics forecasting. International Journal of Information Technology & Decision Making, 2007. 6(01): p. 113-140.

[33] Qi, S., et al., The exploration of internet finance by using neural network. Journal of Computational and Applied Mathematics, 2020. 369: p. 112630.

[34] Siami-Namini, S. and A.S. Namin, Forecasting economics and financial time series: ARIMA vs. LSTM. arXiv preprint arXiv:1803.06386, 2018.

[35] Cao, J., Z. Li, and J. Li, Financial time series forecasting model based on CEEMDAN and LSTM. Physica A: Statistical Mechanics and its Applications, 2019. 519: p. 127-139.

[36] Russell, S. and P. Norvig, Artificial intelligence: a modern approach. 2002.

[37] Deng, L. and D. Yu, Deep learning: methods and applications. Foundations and trends in signal processing, 2014. 7(3–4): p. 197-387.

[38] McCulloch, W.S. and W. Pitts, A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 1943. 5(4): p. 115-133.

[39] Rosenblatt, F., The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 1958. 65(6): p. 386.

[40] Werbos, P., Beyond regression:" new tools for prediction and analysis in the behavioral sciences. Ph. D. dissertation, Harvard University, 1974.

[41] Wang, S.-C., Artificial neural network, in Interdisciplinary computing in java programming. 2003, Springer. p. 81-100.

[42] Leshno, M., et al., Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural networks, 1993. 6(6): p. 861-867.

[43] Rumelhart, D.E., G.E. Hinton, and R.J. Williams, Learning representations by back-propagating errors. nature, 1986. 323(6088): p. 533-536.

[44] Ruder, S., An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747, 2016.

[45] Medsker, L.R. and L. Jain, Recurrent neural networks. Design and Applications, 2001. 5: p. 64-67.

[46] Graves, A., Long short-term memory. Supervised sequence labelling with recurrent neural networks, 2012: p. 37-45.

[47] Dunis, C.L. and A. Nathani, Quantitative trading of gold and silver using nonlinear models. Neural Network World, 2007. 17(2): p. 93.

[48] Bloch, D.A., Recipe for quantitative trading with machine learning. Available at SSRN 3232143, 2018.

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

[50] Jia, W., et al. Quantitative trading on stock market based on deep reinforcement learning. in 2019 International Joint Conference on Neural Networks (IJCNN). 2019. IEEE.

[51] Liu, Y., et al. Adaptive quantitative trading: An imitative deep reinforcement learning approach. in Proceedings of the AAAI conference on artificial intelligence. 2020.

[52] Sun, Y., & Ortiz, J. (2024). Data Fusion and Optimization Techniques for Enhancing Autonomous Vehicle Performance in Smart Cities. Journal of Artificial Intelligence and Information, 1, 42-50.

[53] Lin, W. (2024). The Application of Real-time Emotion Recognition in Video Conferencing. Journal of Computer Technology and Applied Mathematics, 1(4), 79-88.