Iran Journal of Computer Science (2023) 6:29–61
https://doi.org/10.1007/s42044-022-00120-x

REVIEW ARTICLE

Improved market prediction using meta-heuristic algorithms and time series model and testing market efficiency

Milad Shahvaroughi Farahani[1] · Hamed Farrokhi-Asl[2]

Received: 7 October 2021 / Accepted: 24 September 2022 / Published online: 21 October 2022

Abstract

This study aims to evaluate the efficiency of using technical indicators such as closing price, lowest price, highest price, and exponential moving average in the prediction of stock prices. We use a genetic algorithm (GA) and a hybrid of grey wolf optimization and particle swarm optimization binary algorithm (GWO-PSO) as feature selection methods. In addition, we train our neural network by using some metaheuristic algorithms such as harmony search algorithm (HS), particle swarm optimization algorithm (PSO), modified particle swarm optimization algorithm (MPSO), modified particle swarm optimization algorithm with time-varying acceleration coefficients (MPSO-TVAC), moth flame optimization (MFO), wolf optimization algorithm (WOA) and chimp optimization algorithm (ChOA). The experimental results show that using metaheuristic algorithms to fortify neural networks may increase their ability in finding optimal solutions. We also compare the results of our proposed algorithms with the results of the autoregressive integrated moving average (ARIMA) model. To compare the performance of the proposed algorithms and select the best one, we introduce eight estimation criteria for error assessment. Moreover, market efficiency is another important factor that is checked in this paper to avoid abnormal returns. Briefly speaking, it is the first time that ChOA and MFO algorithms have been used for the prediction of stock prices and to improve ANN. In addition, we use two algorithms (i.e., GA and GWO-PSO) for improving the feature selection process. Finally, experimental results show that WOA has the best performance among applied algorithms.

Keywords Artificial neural network · Meta-heuristic algorithms · Feature selection · Chimp optimization algorithm · Moth flame optimization algorithm · Technical indicators

1 Introduction

Technical indicators and other methods such as fundamental analysis and statistical methods are used for stock price prediction. The main factor that should be satisfied to gain more profit by applying the stock market is the "efficient market hypothesis (EMH)". In other words, if the market is efficient, the prediction will be effective. In EMH, information has a significant impact on stock prices and prices may modify themselves according to the information [1]. The efficient market ensures investors have access to similar information. The efficient market is based on the assumption that no system can beat the market because if this system becomes public, everybody will use it. Thus, the market loses its potential profitability [2].

Neural networks are used for the prediction of stock prices because they are able to recognize the linear relationships between inputs and outputs [3]. Many researchers such as economists and financial experts have acknowledged the chaos in the stock market and other complex systems [4]. With the capability of neural networks to learn nonlinear relationships, we can overcome traditional analysis and the other computational methods' drawbacks [5]. In addition to stock market prediction, neural networks are used for other financial tasks. There are a lot of neural networks implemented systems to track demand of products in the market. Additionally, they are able to forecast futures markets, Forex trading, financial planning, and corporate stability and bankruptcy [6]. While banks use neural networks to investigate loan applicants and estimate the probability of bankruptcy, financial managers use neural networks for planning and making

✉ Milad Shahvaroughi Farahani
M.shahvaroughi@khatam.ac.ir

1 Department of Finance, Faculty of Management, Khatam University, Tehran, Iran

2 Sheldon B. Lubar College of Business, University of Wisconsin-Milwaukee, Milwaukee, WI, USA

profitable portfolios at the right time. As investment and transaction levels are growing, people are looking for tools and methods like neural network systems to maximize their profitability and minimize their risk.

Artificial neural networks are one of the main tools in machine learning. Machine learning and deep learning have become almost trending and effective methods commonly used by finance organizations to maximize their profits [7]. However, financial time series are highly nonlinear and their data seems to be completely random [8]. Traditional time series methods such as ARIMA and GARCH models are effective only when the time series are stationary [9]. This assumption is restricting and requires the series to be preprocessed. Moreover, the main problem arises during the implementation of these models in a live trading system, when there is no guarantee of stationarity as new data is added. Using neural networks can help solve this problem.

It is crystal clear that ANN also has some limitations and weaknesses. For example, in ANN, the training phase is very crucial. You may face overtraining, convergence/divergence, the risk of trapping in local minima or maxima, and so forth. One of the main solutions to overcome these drawbacks is using hybrid models. You can use meta-heuristic algorithms along with artificial neural networks as a robust method for prediction. This method has some advantages such as powerful exploration and exploitation, acceptable computational time, and being user-friendly.

In this paper, a genetic algorithm (GA) and a hybrid grey wolf optimization and particle swarm optimization binary algorithm (GWO-PSO) are used to choose the most appropriate input variables (i.e., feature selection). Applying GA as a feature selection (FS) method is so common in the literature, but we apply GWO-PSO as well to elaborate exploitation and exploration features. Some meta-heuristic algorithms such as harmony search (HS), particle swarm optimization (PSO), moth flame optimization (MFO), modified particle swarm optimization (MPSO), modified particle swarm optimization-time-varying coefficient (MPSO-TVAC), whale optimization (WOA), and chimp optimization algorithms (ChOA) are also used to improve the prediction power of the artificial neural network (ANN) and to minimize the network error by obtaining optimized weights and the best number of hidden layers in ANN. These metaheuristic algorithms are different in their mechanisms such as generation of the initial population, discovering search space, finding an optimal solution, the risk of trapping in local minima or maxima, etc. To compare the proposed algorithms' performance and to choose the best one, we introduce eight estimation criteria for error assessment. In this regard, we collect the data of a Khodro company stock price in 5 years starting from 2013 through 2018. Khodro is a big company in the automobile industry in Iran. We access these data from TseClient software. In addition, we apply four types of computer software to process data: (1) Microsoft Excel for getting data; (2) Alyudada NeuroIntelligence for data normalization and processing; (3) MATLAB for training the network; and (4) Neural Designer for getting more details and complementary finding or analysis.

The experimental results show that a hybrid WOA has the best performance. Additionally, applying hybrid models can robust the prediction, and have different advantages such as speeding up calculations, compatibility with complex data structures, and being more user-friendly compared to time series models.

To sum up, the main contributions of this paper are summarized as follows:

- First, we apply almost different metaheuristic algorithms from different categories such as evolutionary algorithms and swarm-based algorithms, and we compare their performance with a time series model called ARIMA. Applying different algorithms leads to different and interesting results. As such, we can compare and explain their pros and cons practically.
- Second, we attempt to use different hybrid metaheuristic algorithms (i.e., GWO-PSO) as feature selection methods.
- Third, we analyze EMH by running some experiments.

In this paper, we want to answer the following questions:

- Do hybrid neural networks have better results (i.e., high predictability with less error)?
- Does the use of genetic algorithms to determine technical indicators effect network error rate and computational speed?
- Which hybrid algorithms have better predictive performance?

The structure of the paper is as follows: The second section belongs to the literature review and reviews different papers about the prediction of the stock price using different techniques especially, machine learning. Section 3 is about methodology and related formulas or equations along with introducing usable techniques. Section 4 is dedicated to finding and results. In this part, we shall try to compare the results and present the best methods based on the error and predictability. Finally, the last section is the conclusion and recommendations for future research.

2 Literature review

A stock market is a public market where the stocks of companies are traded [10]. This market provides opportunities for brokers and companies to invest and is one of the main indicators of the economic situation in each country. The stock market is characterized by some features such as non-linearity, discontinuity, and volatile multifaceted elements because it is related to many factors such as political events, general economic conditions, and broker's expectations [11]. Nowadays, data are processed quickly by applying high-tech tools as well as the advent of communication systems leads to the stock prices fluctuating very fast. As such, many banks, financial institutions, big investors, and brokers have to trade the stock within the shortest possible time [12]. Gaining more profit is the main goal of the investors. So, many researchers are looking for ways to make them able to forecast the market behavior [13]. Based on the literature, there are two main viewpoints about market efficiency. The first one is that markets are efficient and as a result, returns cannot be predicted completely [14]. The second one is that markets are inefficient and abnormal return is possible.

The ANN is considered the best and most verified method in the prediction of stock price [15]. There are many methods for training the ANN and some of them are better than the others in finding the linear and non-linear relationships. The researchers have tied to introduce some methods which have more accuracy and less error in acceptable computation run time. That is why the metaheuristic algorithms are utilized in this context frequently. These algorithms are used to optimize the network and to find the best number of input and hidden layers. It is shown that ANN models outperform traditional statistical models in forecasting the stock price, stock return, exchange rate, and inflation [16, 17].

Göçken et al. [18] used technical indicators and hybrid ANN with GA and HS to predict the price index in the Turkish stock market. The results showed that hybrid meta-heuristic algorithms error is less than simple ANN. They compared the hybrid ANN-HS with the ANN-GA model and found that ANN-HS error is less than ANN-GA. Qiu et al. [19] implemented the fuzzy surfaces to select the optimal input variables. In their study, the optimal set of initial weights and biases are determined by means of GA or SA to increase the accuracy of ANN. Hassanin et al. [20] used GWO to provide the ANN with good initial solutions. The results showed that GWO-based ANN outperforms both GA-based ANN and PSO-based ANN. Faris et al. [21] presented that their approach shows very competitive results based on the set of weights and biases for multi-layer perceptron networks. In addition, GA, PSO, DE, FFLY and cuckoo search are used to compare the performance of the proposed method. Rather et al. [22] observed the field of hybrid forecasting techniques has received lots of attention from researchers to form a robust model. Chong et al. [23] predicted the future market trend of South Korea by examining the effect of three unsupervised feature extraction methods (PCA, autoencoder, and restricted Boltzmann machine (RBM)) on the deep learning network with three loss functions such as NMSE, RMSE, and MSE. Sezer et al. [24] proposed a stock trading system based on a deep neural network for buy–sell–hold predictions. The GA is used to optimize the technical analysis parameters and create the buy–sell point in the system.

Di Persio and Honchar [25] applied three different recurrent neural network (RNN) approaches including a basic RNN, the LSTM, and the gated recurrent unit (GRU) on Google stock price to evaluate which variant of RNN performs better. It is obvious from the results that the LSTM outperformed other variants with a 72% accuracy rate on a 5-day horizon. The authors also explained the hidden dynamics of RNN. Ahmed et al. [26] used ant colony optimization (ACO) in forecasting the stock price of the Nigerian stock exchange. They compared ACO with three other algorithms such as a price momentum oscillator, a stochastic method, and a moving average method. They concluded that ACO is more accurate with lower error than other methods. Ghanbari and Arian [27] used support vector regression (SVR) and butterfly optimization algorithm (BOA) to predict the stock market. They presented a novel BOA-SVR model based on BOA and compared it with eleven other meta-heuristic algorithms on a number of stocks from NASDAQ. The result indicated that the presented model is capable to optimize the SVR parameters very well. Indeed, it is one of the best models with regard to prediction performance accuracy and time consumption.

Kumar et al. [28] reviewed and organized the published papers about stock market prediction using computational intelligence. The related papers were organized according to related datasets, input variables, pre-processing methods, techniques used to feature selection, forecasting methods, and performance metrics to evaluate the methods.

Farahani and Hajiagha [29] used ANN to predict five economic indicators such as S&P500, DAX, FTSE100, Nasdaq, and DJI. They trained the network with some new meta-heuristic algorithms such as social spider optimization (SSO) and bat algorithm (BA). They used some technical indicators as input variables. Then, they used genetic algorithms (GA) as a heuristic algorithm for feature selection and choosing the best indicators. They used some loss functions such as mean absolute error (MAE) as error evaluation criteria. On the other hand, they used some time series models forecasting like ARMA and ARIMA for the prediction of stock price. Finally, they compared the results with each other means ANN-Metaheuristic algorithms and time series models.

You can observe recent papers about forecasting stock prices using neural networks and their methods in Table 1.

As it is clear, researchers have tried to use hybrid models to obtain better results and they have been successful. According to the results, we can figure out that the main merits of using the hybridization technique are as follows:

- Decreasing computation time

Table 1 Recent researches about forecasting the stock price using ANN and other methods

Authors	Subject	Methodology	Results
Pierdzioch and Risse [30]	A ML analysis of the rationality of aggregate stock market prediction	They used a machine learning algorithm known as boosted regression trees (BRT) to perform an orthogonality test of the rationality of aggregate stock market forecasts	The main result is according to the set of predictor variables used in this study, the rational expectations hypothesis (REH) cannot be refused for short-term forecasts and there is evidence against the REH for longer term forecasts
Zhong and Enke [31]	Forecasting the daily return direction of the stock market using hybrid machine learning algorithms	They used DNN and ANN with to forecast the daily direction of future stock market index returns	Simulation results indicate that the DNNs using two PCA-represented datasets obtain higher classification accuracy than those applying the entire untransformed dataset or other hybrid machine learning algorithms
Altan et al. [32]	Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques	A hybrid digital currency prediction model based on long short-term memory (LSTM) neural network and empirical wavelet transform (EWT) that is combined with cuckoo search (CS) is introduced for cryptocurrency time series	The experimental results indicate that the model can successfully handle nonlinear attribute of the cryptocurrency time series
Jiang et al. [33]	The two-stage ML combines models for stock price prediction by gathering mode decomposition, extreme learning machine and improved harmony search	They introduce new two-stage ensemble models by combining empirical mode decomposition (EMD) [or variational mode decomposition (VMD)], extreme learning machine (ELM) and improved harmony search (IHS) algorithm for stock price forecasting	The results prove that the models have better performance in terms of its accuracy and stability compared to the other approaches
Behravan and Razavi [34]	Stock Price Prediction using Machine Learning and Swarm Intelligence	In the first phase of the method, an automatic clustering algorithm clusters the data points into different clusters, and in the second phase a hybrid regression model, which is a combination of particle swarm optimization and support vector regression, is trained for each cluster. In this hybrid method, particle swarm optimization algorithm is used for parameter tuning and feature selection	The accuracy of the proposed method has been measured by 5 companies' datasets, which are active in the Tehran Stock Exchange market, through 5 different metrics. On average, the proposed method has shown 82.6% accuracy in predicting stock price in 1-day ahead
Chandar [35]	Grey wolf optimization-Elman neural network model for stock price prediction	They used ENN and GWO to optimize the parameters of ENN	Results demonstrated that the GWO-ENN model provides accurate prediction for 1 day ahead prediction and outperforms the benchmark models taken for comparison

Table 1 (continued)

Authors	Subject	Methodology	Results
Chen et al. [36]	A graph convolutional feature based convolutional neural network for stock trend forecasting	They introduce a novel method for stock trend prediction applying graph convolutional feature based convolutional neural network model, where both stock market data and individual stock data are investigated	The experimental results show that the proposed GC–CNN based method outweighs several stock trends forecasting methods and stock trading approaches
Kumar [37]	Hybrid models for intraday stock price forecasting based on artificial neural networks and metaheuristic algorithms	This paper introduces nine novel integrated models for prediction of intraday stock price based on the potential of three ANNs	Results proved that the PSO-BPNN model yielded the highest prediction accuracy in estimating intraday stock price
Farahani and Hajiagha [29]	Predicting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models	Training ANN using SSO and BA algorithms	Better performance of Hybrid model

- Decreasing model complexity
- Avoiding local minima or maxima trap
- Avoiding fast convergence, etc.

To clarify further, limitations of the previous methods and advantages and disadvantages of methods used in these articles are provided in Tables 2 and 3.

According to Table 2, we cannot say which method is better because each one has its own pros and cons. Some methods have more capabilities such as compatibility with non-linear data structure and speeding up calculations. On the other hand, they may have some limitations such as being hard to train and sensitive to noise and outliers. So, the usable method depends on the type of problem. We tried to present the strengths and weaknesses of applicable algorithms in Table 3. In this table, we want to show that all methods are not perfect and without any limitations.

3 Methodology

3.1 Input variables selection

This section describes the input variables selection methodology. Initially, for each case, 42 technical indicators are investigated as input variables. This number of input variables increases the complexity of the model and at some point, they do not provide extra information. For this reason, we use GA to select the most informative input variables. As such, using GA we can evaluate the usefulness of indicators or eliminate irrelevant ones to simplify the proposed model. Table 4 demonstrates all considered technical indicators as input variables [18, 38].

In Table 4, stochastic indicators ($\%K$ and $\%D$) have two types: fast and slow. High and low are maximum and minimum price of the n period ago, respectively. About RSI indicator, average gain and average loss is defined as follows:

$$\text{Average gain} = \left[(\text{previous average gain}) \times 13 + \text{current gain}\right]/14$$

$$\text{Average loss} = \left[(\text{previous average loss}) \times 13 + \text{current loss}\right]/14.$$

According to the Bollinger band indicator, MA stands for moving average, TP means typical price, n is equals to the number of periods which is usually 20 and finally, m refers to standard deviation and it is often 20. The last notation, that is $\sigma[\text{TP}.n]$, equals standard deviation during n period of TP.

3.2 Artificial neural network (ANN) model

At first, ANN is applied without adding any algorithm and then hybrid ANN is used for selecting input variables and determining the number of input and hidden layers. In this article, we consider multi-layer perceptron (MLP) including three layers (two layers for input and output variables and one layer for hidden layer). The input layer includes 42 input variables which means there are 42 neurons in the input variable. Because the output layer has one variable, it has one neuron. In this paper, the number of neurons in the hidden layer is obtained through trial and error. So, we examine 1–32 neurons in hidden layer and choose the fittest number of neurons that have the most accurate. For training ANN, we use error-back propagation. It should be mentioned that the minimization algorithm in learning the model is Levenberg–Marquardt (LM) algorithm which is used to find the

Table 2 Limitations of the previous methods

No	Methods	Purpose	Limitations
1	ARIMA	Prediction and clustering	It does not work well for non-linear time series It needs more data It Takes a long processing time for a large dataset
2	BPNN	Prediction	Sensitive to noise Performance depends on initial values Slow convergence speed Converging to a local minimum
3	CART (Classification and Regression Trees)	Classification and forecasting	Unstable even if the training data are small changed
4	GP (Gaussian Process)	Classification and forecasting	It generates" black box" models that are difficult to interpret It could be computationally expensive
5	GRNN (Generalized Regression neural network)	Classification and forecasting	It requires more memory space to store the model It could be computationally expensive because of its huge size
6	Hierarchical clustering	Clustering	The length of each time series is the same because of the Euclidean distance Useful only for small datasets because of its quadratic computational complexity
7	HMM (Hidden Markov Model)	Clustering, classification and clustering	It requires parameters to be set and is based on user assumptions that may be false with the result that clusters would be inaccurate It takes a long-time processing for a large dataset
8	K-Mean	Clustering	The number of clusters must be specified in advance Sensitive to noise Only spherical shapes can be determined as clusters Unable to handle long time series effectively because of poor scalability
9	KNN (K Nearest Neighbor	Classification and forecasting	The number of nearest neighbor's must first be determined It can be computationally expensive Memory limitation Sensitive to the local structure of the data
10	LR (Logistic Regression)	Classification and forecasting	Sensitive to outliers Strong assumptions
11	LSTM (Long Short-Term Memory)	Classification and forecasting	Lacks a mechanism to index the memory while writing and reading the data the number of memory cells is linked to the size of the recurrent weight matrices
12	MLP (Multi-Layer Perceptron)	Classification and forecasting	Convergence is quite slow Local minima can affect the training process Hard to scale
13	PSO (Particle Swarm Optimization)	Forecasting	Lacks a solid mathematical foundation for analyzing future development of relevant theories

Table 2 (continued)

No	Methods	Purpose	Limitations
14	RBF (Radial Basis Function Neural Network)	Classification and forecasting	Classification process is slow
15	RF (Random Forest)	Classification and forecasting	It requires more computational power and resources because it creates a lot of trees Requires more time to train than decision trees
16	RNN (Recurrent Neural Network)	Classification and forecasting	- Difficult to train
17	SOM (Self-Optimizing Maps)	Clustering and classification	Does not work well for time series of unequal length because of the difficulty involved in determining the scale of weight vectors Sensitive to outliers
18	SVM (Support Vector Machine)	Classification and forecasting	- Sensitive to outliers Sensitive to parameter selection
19	SVR (Support Vector Regression)	Forecasting	Sensitive to users defined free parameters
20	ANN (Artificial Neural Network)	Classification and forecasting	Over-fitting Sensitive to parameter selection—ANNs just give predicted target values for some unknown data without any variance information to assess the prediction

minimum error point [39]. The number of training epochs is 1000 and for the first-time training rate is 0.01. We decrease this rate to 0.001 in order to obtain more accurate results. The output function of the hidden layers is the sigmoid function and the threshold function of the output layer is linear function. Figure 1 represents the architecture of the proposed neural network [18].

In Fig. 1, P is the input pattern, b_1 is the vector of bias weights on the hidden neurons, and w_1 is the weight matrix between 0th (i.e., input) layer and 1th (i.e., hidden) layer. a_1 is the vector containing the outputs from the hidden neurons, and n_1 is the vector containing net-inputs going into the hidden neurons, a_2 is the column-vector coming from the second output layer, and n_2 is the column-vector containing the net inputs going into the output layer. w_2 is the synaptic weight matrix between the 1st (i.e., hidden) layer and the 2nd (i.e., output) layer and b_2 is the column-vector containing the bias inputs of the output neurons. Each row of w_2 matrix contains the synaptic weights for the corresponding output neuron [18].

This study includes two main parts. The first one includes calculating technical indicators and selecting the most informative indicator by using GA. The second part is prediction of closing price by using different hybrid ANN models and comparing their prediction errors. Figure 2 represents the research methodology [40] and the role of metaheuristic algorithms in the article. In this regard, we divide stock price data from 2013 to 2018 into two parts: training and testing. Then, it is analyzed with artificial intelligence algorithms and we predict the next day closing stock price. We use 70% and 30% of data for training, validation and testing, respectively. Afterward, we compare models with 8 criteria for prediction error. In this research, we used 42 technical indicators as input variables. To make these variables usable as input variables, they should be scaled and normalized between − 1 and 1. So, the largest number will be "1" and the smallest number will be "− 1". We can do this with Alyuda Neuro Intelligence software. In Eq. 1, numerator i is the amount of data.

$$\widetilde{S_i} = \frac{(S_i - S_{\min})}{S_{\max} - S_{\min}}. \quad i = 1.2 \ldots N. \tag{1}$$

3.2.1 Hybrid GA-ANN model

In this method, GA is used as a feature selection approach. The applied encoding approach is binary solution representation. Each chromosome contains 47 bits whereas the first 42 bits represent the existence or nonexistence of input (technical indicator) variables. "1" represents the existence and “0” shows the non-existence of the corresponding variable. Five other bits are equal to 1–32 (2^5=32) which shows the number of neurons in the hidden layer. The population size of GA is assumed to be 20 [41]. The first population is generated randomly. The fitness function is mean square error (MSE). The smallest MSE in these series is the better choice for the next forecasting period. For increasing the training phase speed, the epochs are considered 100. At first, the training (learning)

Table 3 Strengths and weaknesses of considered methods

Algorithms	Strengths	Weaknesses
ANN	Storing information on the entire network	Hardware dependence
	Ability to make machine learning	Determination of proper network structure
	Parallel processing capability	The duration of the network is unknown
	Having fault tolerance	Unexplained behavior of the network
	Gradual corruption	–
GA	Easy implementation	Time consuming
	Easy to understand	Difficult to show branching and looping
		Big tasks are difficult to put in algorithm
PSO	Calculation in PSO is simple	All solutions converge prematurely and consequently lose the population diversity
MFO	Balance between exploration and exploitation	Relaxed convergence
ChOA	Convergence speed	Unaffordable sampling rate
	Reduced complexity	Difficult because of randomness
	Reduced processing time	
WOA	Less computational cost and function evaluation	Using the current randomization technique in WOA would increase computational time especially for the highly complex problem
	High ability to avoid local optima since a set of solutions are involved during optimization	WOA has poor convergence speed in both exploration and exploitation phases
	Information can be exchanged between the candidate Solutions and assist them to overcome different difficulties of search spaces	WOA cannot work in the fields of classification and dimensionality reduction as it is not suitable for the binary space

Table 4 Important and most common technical indicators as input variables

Technical indicators and the method of calculation	
$\text{Diff} = \text{Close}_{\text{Today}} - \text{Close}_{\text{Yesterday}}$	$\text{M}_{\text{Open}} = \text{Open}_{\text{Today}} - \text{Open}_{\text{Yesterday}}$
Close	$\text{M}_{\text{High}} = \text{High}_{\text{Today}} - \text{High}_{\text{Yesterday}}$
High	$\text{M}_{\text{Low}} = \text{Low}_{\text{Today}} - \text{Low}_{\text{Yesterday}}$
Low	$\text{M}_{\text{Close}} = \text{Close}_{\text{Today}} - \text{Close}_{\text{Yesterday}}$
Open	$\text{Acc}_{\text{Open}} = \text{M Open}_{\text{Today}} - \text{M Open}_{\text{Yesterday}}$
$\text{SMA (5)} = \frac{(\text{Close1}+\text{Close2}+\cdots+\text{Close5})}{5}$	$\text{Acc}_{\text{Close}} = \text{M Close}_{\text{Today}} - \text{M Close}_{\text{Yesterday}}$
$\text{SMA (6)} = \frac{(\text{Close1}+\text{Close2}+\cdots+\text{Close6})}{6}$	$\text{Acc}_{\text{High}} = \text{M High}_{\text{Today}} - \text{M High}_{\text{Yesterday}}$
$\text{SMA (10)} = \frac{(\text{Close1}+\text{Close2}+\cdots+\text{Close10})}{10}$	$\text{Acc}_{\text{Low}} = \text{M Low}_{\text{Today}} - \text{M Low}_{\text{Yesterday}}$
$\text{SMA (20)} = \frac{(Close1+Close2+\cdots+Close20)}{20}$	$\%K = \left[\frac{(\text{Close}-\text{Low})}{(\text{High}-\text{Low})}\right] * 100$
$\text{EMA (5)}_{\text{ Today}} = \frac{\text{CloseToday}*\text{k}+\text{EMA(5)Yestarday}*(1-k)}{5}$ $\text{K} = \frac{2}{5+1}.EMA(5)0 = SMA(5)$	%D = 3-day SMA of %K
$\text{EMA (6)}_{\text{ Today}} = \frac{\text{CloseToday}*k+\text{EMA(6)Yestarday}*(1-k)}{6}$ $\text{K} = \frac{2}{6+1}.\text{EMA(5)0} = \text{SMA(6)}$	Slow%K = = Fast%D
$\text{EMA (10)}_{\text{ Today}} = \frac{\text{CloseToday}*\text{k}+\text{EMA(10)}Yestarday*(1-k)}{10}$ $\text{K} = \frac{2}{10+1}.EMA(10)0 = SMA(10)$	Slow%D = 3-day SMA of %D
$\text{EMA (20)}_{\text{ Today}} = \frac{\text{CloseToday}*k+\text{EMA(20)Yestarday}*(1-k)}{20}$ $\text{K} = \frac{2}{20+1}.EMA(20)0 = SMA(20)$	$\text{William's}\%R = \frac{(\text{Highesthigh}-\text{Close})}{(\text{Highesthigh}-\text{Lowsetlow})}$
$\text{TMA (5)} = \frac{(\text{SMA(1)}+\text{SMA(2)}+\cdots+\text{SMA(5)})}{5}$	$\text{RSI} = 100 - \frac{100}{1+RS}.\text{RS} = \frac{\text{AverageGain}}{\text{AverageLoss}}$
$\text{TMA (6)} = \frac{(\text{SMA(1)}+SMA(2)+\cdots+SMA(6))}{6}$	Middle Band = SMA (20)
$\text{TMA (10)} = \frac{(SMA(1)+\text{SMA(2)}+\cdots+\text{SMA(10)})}{10}$	Upper Band = MA(TP,n) + m ∗ σ[TP,n]
$\text{TMA (20)} = \frac{(\text{SMA(1)}+\text{SMA(2)}+\cdots+\text{SMA(20)})}{20}$	Lower Band = MA(TP,n) − m ∗ σ[TP,n]
$\text{AccDist} = \text{AccDist}_{\text{ Yesterday}} + \text{Volume*CLV}$ $\text{CLV} = \frac{[(\text{Close}-\text{Low})-(\text{High}-\text{Close})]}{\text{High}-\text{Low}}$	$\text{MP} = \frac{(\text{High}+\text{Low})}{2}$
MACD = EMA (12) − EMA (26)	$\text{ROC} = \frac{(\text{Closetoday}-\text{Close}N\text{previousday})}{\text{Close}N\text{previousday}}$
$\text{Signal}_{\text{MACD}} = \text{EMA(MACD,9)} = \text{MACD}_{\text{ Today}}*0.2 + (\text{Signal}_{\text{MACD Yesterday}}*(0.8))$	$\text{Typical Price} = \frac{(\text{High}+\text{Low}+\text{Close}+\text{Open})}{4}$

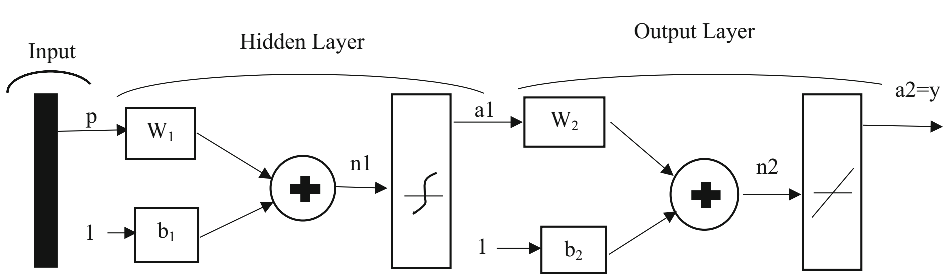


Fig. 1 Architecture of the proposed neural network

rate is 0.01 which will decrease during the iterations based on the considered results. By increasing the epochs to 1000, it is possible to get better results. The considered parameters in the genetic algorithm are summarized in Table 5:

More details about GA mechanism as feature selection can be seen in Table 6.

Figure 3 represents the related flowchart of GA-ANN [19].

Among 20 parents and 20 generated children, we select the 20 best individuals as new generations. The new generations keep repeating the mentioned method until reaching the termination condition. One of the termination conditions is repeating the best individual to 100 generations. If this condition does not hold, we address the maximum number of

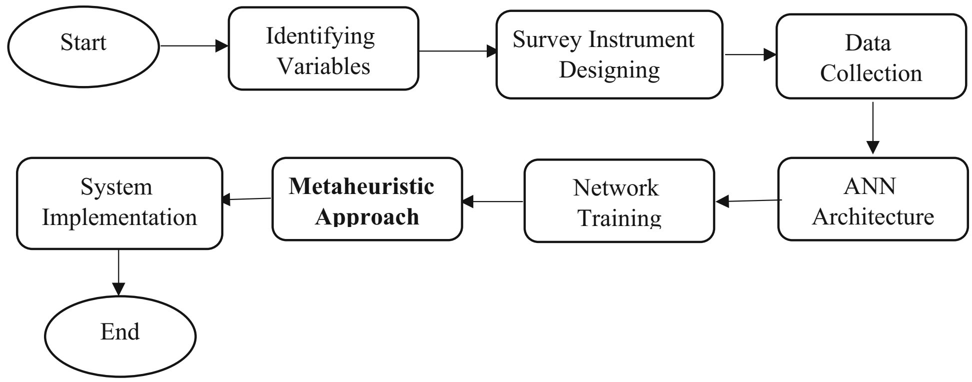


Fig. 2 Research methodology

Table 5 GA parameters

Output Error	Output Activation Function	Input Activation Function	Mutation Rate	Crossover Rate	Number of Generation	Population size
MSE	Logistic	Logistic	0.1	0.9	50	20
Selection parents			Mutation		Crossover	
Roulete wheel method			Binary Method		One-point method	

Table 6 GA-ANN algorithm

```
function GENETIC-ALGPRITHM (population, FITNESS-FN) returns an individual
   inputs: population, a set of individuals
           FITNESS-FN, a function that measure the fitness of an individual

   repeat
        new_population ← empty set
        for i = 1 to SIZE (population) does
             x ← RANDOM-SELECTION (population, FITNESS-FN)
             y ← RANDOM-SELECTION (population, FITNESS-FN)
             Child ← REPRODUCE (x, y)
             if (small random probability) then child ← MUTATE (child)
             add child to new_population
        Population ← new_population
   until some individual is fit enough, or enough time has elapsed
   return the best individual in population, according to FITNESS-FN
```

iterations. The maximum number of iterations equals 2000. You can also see the mutation and crossover operator in Fig. 4.

The crossover of two parent strings produces offspring (new solutions) by swapping parts or genes of the chromosomes. Crossover has a higher probability, typically in the range of 0.8–0.95.

3.2.2 Hybrid PSO-ANN model

PSO begins with the initial population and in sequential iterations moves toward an optimal solution [42]. In each iteration, two solutions are specified (X_j^{Gbest} and $X_j^{i.pbest}$) which represent the best-acquired location for all particles and the best location for the current solution, respectively. The structure of PSO is that in each iteration, each particle set its location in search space with regard to global and its own best location [42] (see Table 7).

In this study, we perform seven steps for training neural network by PSO that are summarized as follows:

1. Collecting data.
2. Creating network.
3. Estimating network.
4. Initializing weights and biases.
5. Training network by PSO.
6. Validating network.
7. Using network.

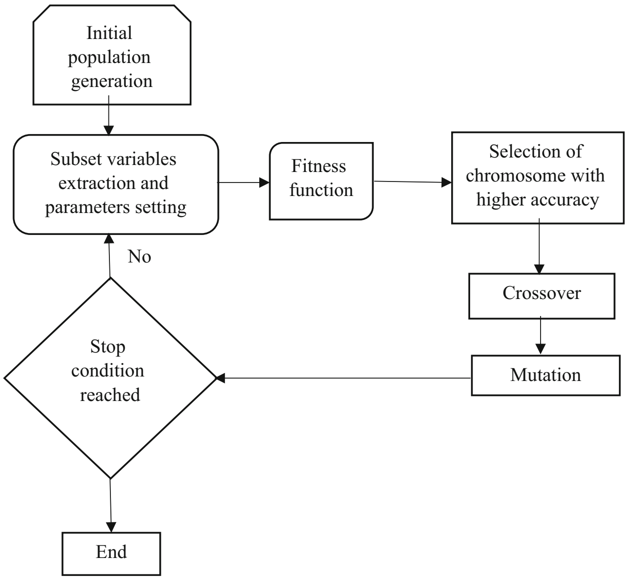


Fig. 3 Considered GA flowchart for training ANN

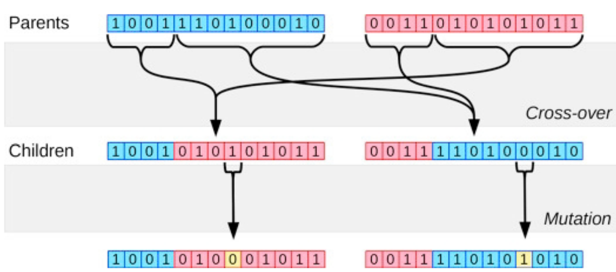


Fig. 4 Cross-over and mutation operator

Table 7 PSO parameters

Parameters	Size
Upper Bound	1.5
Lower Bound	− 1.5
C_1	1.5
C_2	2.5
Max Iteration	2000

3.2.3 Hybrid HS-ANN model

In this study, we use the HS algorithm to train ANN and find the fittest number of input and hidden layers. The HS consists of three basic phases: initialization, improvisation of a harmony vector, and updating the HM [43]. In addition, other parameters of HS should be determined. These parameters are harmony memory size (HMS) which is equals 100, harmony memory considering rate (HMCR) which is equals 0.95, pitch adjusting rate (PAR) which is 0.3, and bandwidth (bw) which is 0.2. We can show the HM with HM$S*(N+1)$ where N is 42. The HS parameters are listed in Table 8.

Table 8 HS parameters

Parameters	Size
Lower Bound	− 11
Upper Bound	11
HMS	11
NHMS	100
Max Iteration	1000
HMCR	0.75
PAR	0.05
Fret Width (FW)	0.1
FW-Damp	0.95

3.2.4 Hybrid GWO-PSO algorithm

It is a kind of hybrid algorithm including both attributes of GWO and PSO optimization algorithms in order to increase the algorithm's capability to exploit PSO with the ability to explore GWO to achieve both optimizer strength [40].[1]

3.2.5 MPSO algorithm

From equations in the PSO algorithm, it is clear that it has three parts: the first part is the previous velocity of the particles; the second and third parts are the ones contributing to the change of the velocity of a particle [44]. A model which adds a second part to PSO model is MPSO. It has a parameter called inertia weight.[2]

3.2.6 Hybrid MPSO-TVAC algorithm

To improve the quality of PSO in the optimization process and find the best solution, a novel modified PSO with time-varying acceleration coefficients (MPSO-TVAC) is proposed [46]. This method has a new parameter increasing the exploration capability thus it decreases the chance of trapping in local optimum.[3]

3.2.7 MFO algorithm

MFO is an optimization algorithm that proposed in 2016 by Mirjalili [47]. In the MFO algorithm, moths are candidate

[1] To get more information, please see Ref. [40].

[2] To get more information, please see Ref. [45]

[3] To get more information, please see Ref. [46].

solutions and the position of moths in the space are the problem's variables.[4]

3.2.8 WOA

WOA is designed based on the hunting technique used by humpback whales [48]. They have a hunting mechanism called the bubble-net feeding method. Humpback whales try to create bubbles and then encircle and attack the prey. They update their positions based on the current best candidate and near-optimal solution. After considering the best candidate, they update their positions based on the best search agent. The following steps are needed for the operation of WOA.

Step 1. The standard whale optimization algorithm starts by setting the initial values of the population size n, the parameter a, coefficients A and C, and the maximum number of iterations max_itr.
Step 2. Initialize the iteration counter t.
Step 3. The initial population n is generated randomly and each search agent x_i in the population is evaluated by calculating its fitness function $f(x_i)$.
Step 4. Assign the best search agent X.
Step 5. The following steps are repeated until the termination criterion is satisfied.
Step 5.1. Update the iteration counter $t = t + 1$.
Step 5.2. All the parameters $a.A.C.l$ and P are updated.
Step 5.3. The exploration and exploitations are applied according to the values of p and $| A |$
Step 6. The best search agent X is updated.
Step 7. The overall process is repeated until termination criteria is satisfied.
Step 8. Determine the best search agent (solution) found so far (X).

3.2.9 ChO algorithm

Generally, the hunting process of chimps is divided into two main phases: Exploration which consists of driving, blocking and chasing the prey and exploitation which consists of attacking the prey [49].

The chimps hunting model means driving, blocking, chasing and attacking has been modeled.[5]

3.3 ARIMA forecasting model

Auto-regressive integrated moving average (ARIMA) is used for modeling time series which are stationary and you cannot find or see any special pattern. When we use the ARIMA, we would like to check if there is a linear relationship between past data and future data. The ARMA model includes different steps [50]. For example, first, you should check the stationarity. If the series is non-station, you should turn it into station data. There are a lot of methods for doing so. One of them is the Kolmogorov–Smirnov test. Figure 5 shows the flowchart of the ARIMA method.

[4] To get more information, please see Ref. [47].

[5] To get more information, please see Ref. [49].

Table 9 Most common loss functions

Error criterion formula	Error criterion
$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\lvert ei\rvert$	Mean Absolute Error
$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} ei^2$	Mean Squared Error
$\mathrm{RMSE} = \sqrt{\frac{1}{n}}\sum_{i=1}^{n} ei^2$	Root Mean Squared Error
$\mathrm{MARE} = \frac{1}{n}\sum_{i=1}^{n}\left\lvert \frac{ei}{ai}\right\rvert$	Mean Absolute Relative Error
$\mathrm{MSRE} = \frac{1}{n}\sum_{i=1}^{n}\left\lvert \frac{ei}{ai}\right\rvert^2$	Mean Squared Root Error
$\mathrm{RMSRE} = \sqrt{\frac{1}{n}}\sum_{i=1}^{n}\left\lvert \frac{ei}{ai}\right\rvert^2$	Root Mean Squared Relative Error
$\mathrm{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left\lvert \frac{ei}{ai}\right\rvert$	Mean Absolute Percentage Error
$\mathrm{MSPE} = \frac{100}{n}\sum_{i=1}^{n}\left\lvert \frac{ei}{ai}\right\rvert^2$	Mean Squared Prediction Error

3.4 Testing efficient market hypothesis (EMH)

One of the main assumptions in market analysis is that the market is efficient. When you figure out if the market is efficient or not, the result affects your decision. When a market is efficient it means that abnormal returns cannot be earned by searching for mispriced stocks. So, the weak form of the EMH declines the value of technical analysis. As we mentioned, financial time series are not normal and they are skewed. So, we should perform the non-parametric test. Since the main focus of the article is on ANN, a brief explanation is provided about EMH. To decide if a sample comes from a population with a specific distribution, the Kolmogorov–Smirnov goodness of fit test is used [51]. The randomness of data is also evaluated using a run test [52].

3.5 Loss functions

For the loss function calculation, we utilize some loss functions in MATLAB to determine the best performance model which has the highest (maximum) accuracy and the lowest (minimum) error. Table 9 summarizes the available loss functions in MATLAB. Finally, we compare their accuracy with respect to calculated loss functions.

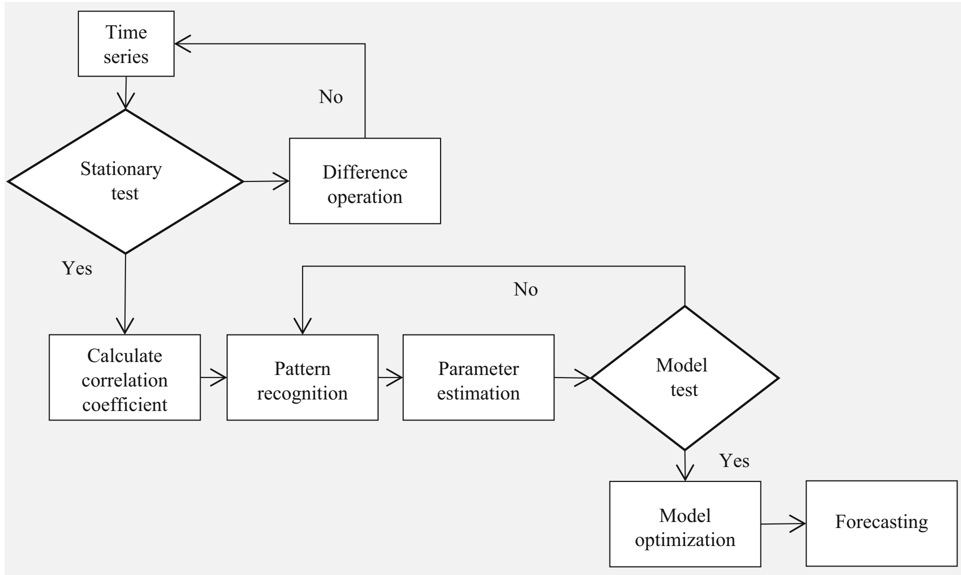


Fig. 5 ARIMA flowchart [33]

4 Findings and results

In this section, we shall discuss the test data and numerical results obtained by using the presented algorithms.

4.1 Data statistics

First of all, as we mentioned earlier, we need to normalize data and scale them between [− 1, + 1]. Table 10 shows the normalized data.

In this study, 42 technical indicators are used to predict stock prices. Among these indicators, 41 variables are used as input variables and one variable is the output or target variable, that is closing price for the next day. To run the experiments, the data is collected from the beginning of 2013 to the end of 2018 which is the daily stock price of Khodro company which is a big company in the automobile industry in Iran. The reasons for selecting this company are:

(1) Data availability and easy access to data.
(2) It is the biggest and most famous company in this industry in Iran.

To access the data, we accept Laboratory risk. The data was obtained through two different websites which are called TSETMC and CODAL (http://tsetmc.ir/ and https://www.codal.ir/). In addition, there is a financial data software called TSECLIENT 2.0 and you can download data easily according to the symbol name in the stock market.

The following pie chart shows the segmentation of data in the experiments. The total number of instances is 1082. The number of training instances is 650 (60.1%), the number of selection instances is 216 (20%), the number of testing instances is 216 (20%), and the number of unused instances is 0 (0%) (see Fig. 6).

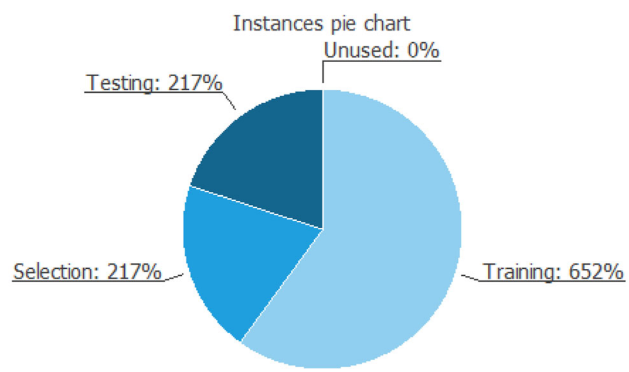


Fig. 6 Instances pie chart

In the appendix, Table 33 shows the value of the correlations between all input and target variables. The maximum correlation (0.994050) is between the input variable "Typical Price" and the target variable.

4.2 ANN model

First, we predict stock prices by ANN without using any additional algorithm. We perform it in three steps: (1) finding the best architecture (designing); (2) training the network; (3) validation and testing. We use 70% of the data for training and the remaining is used for validation and testing. Table 11 presents the best architecture of the network.

An architecture highlighted with blue color shows the best architecture including 41 neurons as the input layer, 50 neurons as the hidden layer, and one layer for output with the highest R-Squared. The best network error during each

Table 10 Data preview table

No.	Open	High	Low	Close	ADL	RS	RSI	True Range	…	WC
1	– 0.571152	– 0.56772	– 0.571152	– 0.602663	0.079454	– 0.834473	0.153878	– 0.104088	…	0.214768
2	– 0.57099	– 0.58494	– 0.600638	– 0.559685	0.01842	– 0.770441	0.324248	– 0.661285	…	0.201117
…	…	…	…	…	…	…	…	…	…	…
1086	0.192409	0.183533	0.164947	0	– 0.007859	– 0.919552	– 0.224432	– 0.454655	…	0.582769

Table 11 Best network architecture

ID	Architecture	# of Weights	Fitness	Train Error	Validation Error	Test Error	AIC	Correlation	R-Squared	Stop Reason
1	[41-6-1]	259	0.006067	137.504364	164.781677	164.826019	-716.694821	0.998228	0.996383	All iterations done
2	[41-103-1]	4430	0.011359	68.947495	83.625526	88.037529	7117.236782	0.999472	0.998882	All iterations done
3	[41-65-1]	2796	0.008412	93.307922	107.228661	118.882477	4071.920732	0.999273	0.99845	All iterations done
4	[41-42-1]	1807	0.011852	76.771774	88.365868	84.371315	1950.350733	0.999359	0.998668	All iterations done
5	[41-28-1]	1205	0.010616	76.715004	97.9263	94.194412	745.80626	0.99933	0.998623	All iterations done
6	[41-56-1]	2409	0.01278	60.321045	74.297363	78.248352	2976.859906	0.99956	0.999113	All iterations done
7	**[41-50-1]**	**2151**	**0.014729**	**55.884319**	**61.415993**	**67.892784**	**2404.63151**	**0.999641**	**0.999265**	**All iterations done**
8	[41-46-1]	1979	0.011669	67.853676	79.488838	85.697395	2203.466887	0.999504	0.998976	All iterations done
9	[41-53-1]	2280	0.01112	69.914757	83.504906	89.932007	2827.490286	0.999401	0.998768	All iterations done
10	[41-48-1]	2065	0.01024	76.923798	91.085052	97.653877	2467.806658	0.999361	0.998659	All iterations done
11	[41-51-1]	2194	0.007658	98.194122	115.139694	130.577759	2905.487135	0.999134	0.998195	All iterations done
12	[41-49-1]	2108	0.010882	68.700508	83.839325	91.891502	2470.595471	0.99952	0.999025	All iterations done

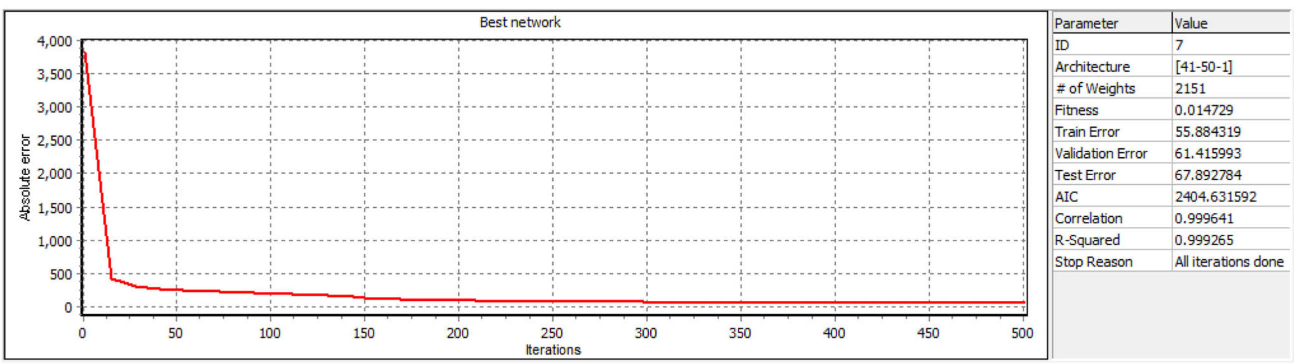


Fig. 7 Best network error

Table 12 Network properties

Parameter	Value
Input activation FX	Logistic
Output name	Close price
Output error FX	Sum-of squares
Output activation FX	Logistic

iteration is also shown in Fig. 7. Additionally, the network properties are summarized in Table 12.

Figure 8 depicts the best performance in three parts of the method (training, validation, and testing). Regression and related plots are displayed in Fig. 8 with related statistics. More details about different loss estimations are summarized in Table 13.

In addition, we apply the quasi-Newton method as an optimization algorithm in the training phase. It is designed based on Newton's method, but it does not need to calculate the second derivatives. Instead, the quasi-Newton method computes an approximation of the inverse Hessian at each iteration of the algorithm, by only using gradient information. Table 34 shows the results of this training strategy. Figure 9 also shows the training and selection errors in each iteration. The blue line represents the training error and the orange line represents the selection error. The initial value of the training error is 15.9893, and the final value after 468 epochs is 0.000213652. The initial value of the selection error is 20.6414, and the final value after 468 epochs is 0.000372232.

Table 14 shows the training results by the quasi-Newton method. It includes some final values in the neural network, the loss function, and the optimization algorithm.

Fig. 8 ANN regression

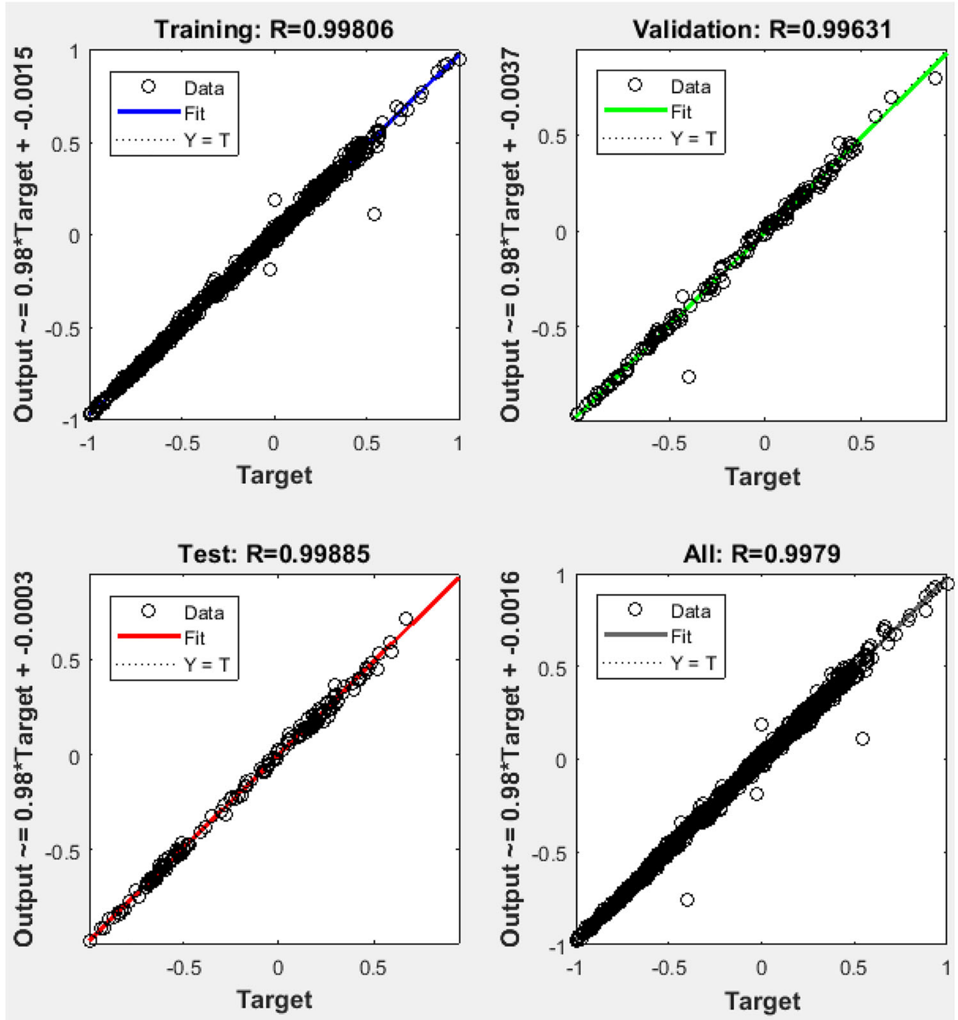


Table 13 Training, validation and testing error (before using GA)

Symbol	Training Error	Validation Error	Testing Error	MSE	MAE	SSE	SAE	R^2
Khodro	0.076	0.14	0.046	0.080	0.0174	0.8780	18.86	0.9979

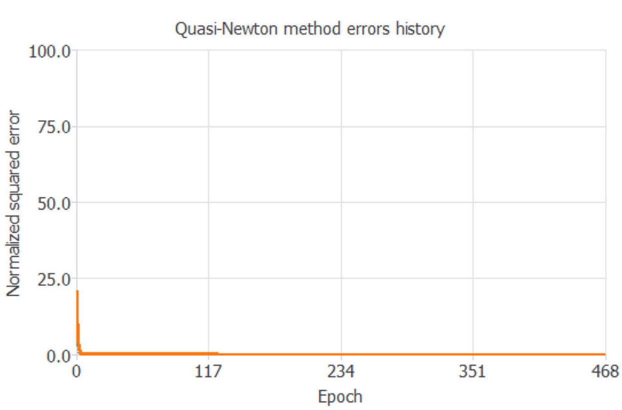


Fig. 9 Training using Quasi-Newton optimization algorithm

Table 14 Training error (using Quasi-Newton)

Criteria	Value
Final parameters norm	1.12
Final trading error	0.000214
Final selection error	0.000372
Final gradient norm	0.000867
Epochs number	468
Elapsed time	00:01
Stopping criterion	Gradient norm goal

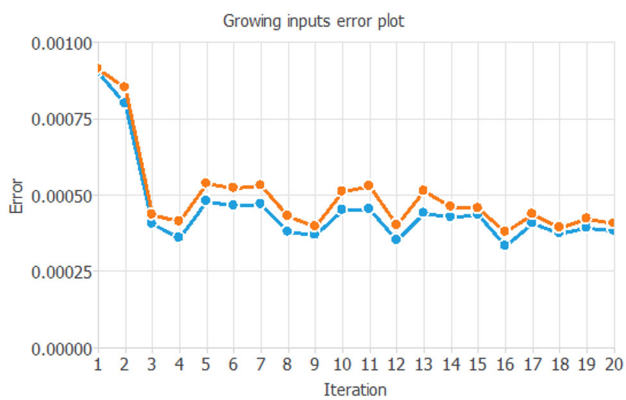


Fig. 10 Growing input error plot

Table 15 GA results

Parameters	Value
Optimal number of inputs	20
Optimum training error	0.000382034
Optimum selection error	0.000405822
Generations number	20
Elapsed time	00:02

4.2.1 Hybrid GA-ANN model

The input variables selection is the way to find the optimal subset of inputs that has the minimum error. A growing input method is used here as an inputs selection algorithm Fig. 10 shows the error history for the different subsets during the growing input selection process. The blue line represents the training error and the orange line symbolizes the selection error.

Table 15 shows the inputs selection results by the growing inputs algorithm. It includes some final values for the parameters of the neural network, the error function and the inputs selection algorithm.

A graphical representation of the deep architecture is depicted in Fig. 11. It contains a scaling layer, a neural network, and an un-scaling layer. The yellow, blue, and red circles represent scaling neurons, perceptron neurons, and un-scaling neurons, respectively. The number of inputs is 20, and the number of outputs is 1. The complexity, represented by the number of hidden neurons, is 1. Table 16 shows different types of errors in the training, selection, and testing phases.

Figure 12 represents testing the network. The horizontal line shows the closing price and the vertical line shows the output range which has normalized between [1, – 1]. Indeed, the output results of the neural network (blue line) are so close to the target values (red line).

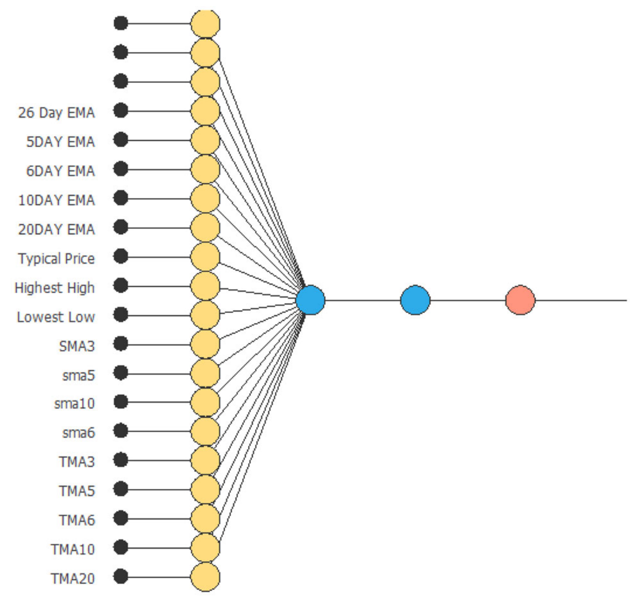


Fig. 11 Final architecture

Table 16 GA error table

Loss Functions	Training	Selection	Testing
Sum squared error	20.1567	6. 22,078	7.19076
Mean squared error	0.0310102	0.0287999	0.0332905
Root mean squared error	0.176097	0.169705	0.182457
Normalized squared error	0.165736	0.17363	0.203281
Minkowski error	45.1816	14.1957	16.0548

4.2.2 Hybrid PSO-ANN model

Since we would like to predict stock price (closing price), we should create a fitness function. We perform it in the format of *M* file in MATLAB by adjusting the considered parameters which we explained before. At first, we initialize the algorithm which includes population and speed with initial values of *pbest* and *gbest*. First, we consider values for C_1 and C_2 with given iteration which is 1000 here. We should update parameters constantly for achieving the intended goal. we should mention that the network is feedforward. We can see the regression in Fig. 13. Table 17 also shows the estimation errors and different loss functions.

4.2.3 Hybrid HA-ANN model

Like other algorithms such as GA and PSO, we perform several steps for training the network and solving the problem. The network structure is a feed-forward ANN (FFANN).

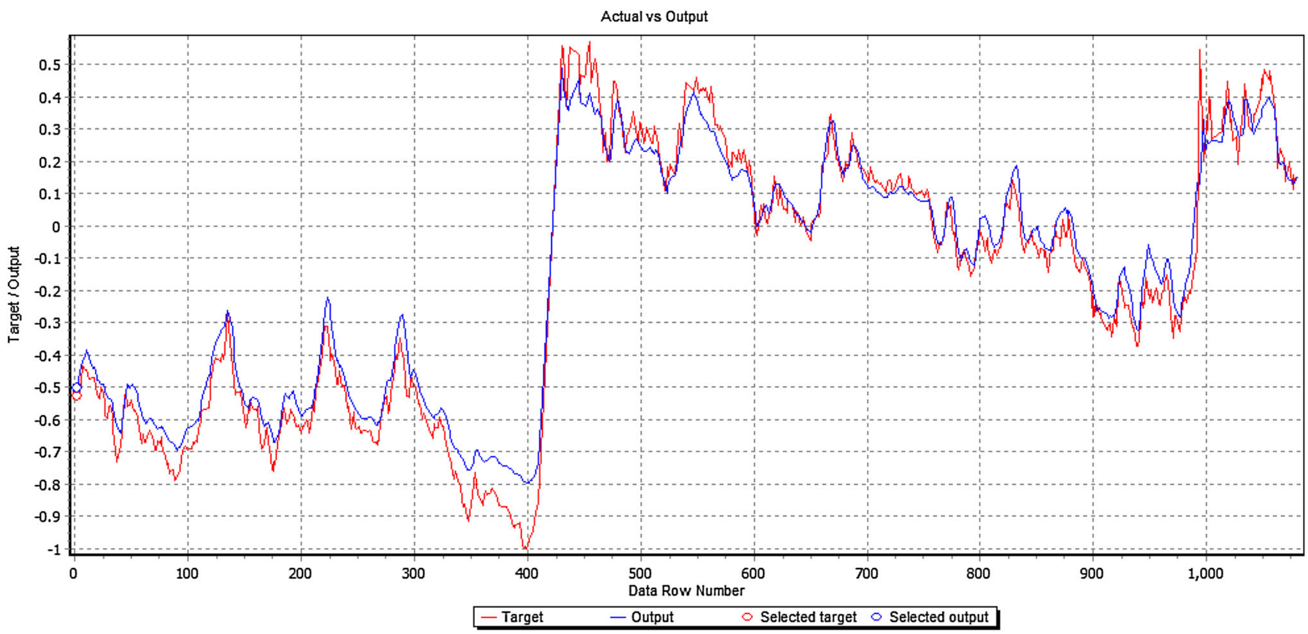


Fig. 12 Target versus output trend (Khodro)

Fig. 13 ANN-PSO regression

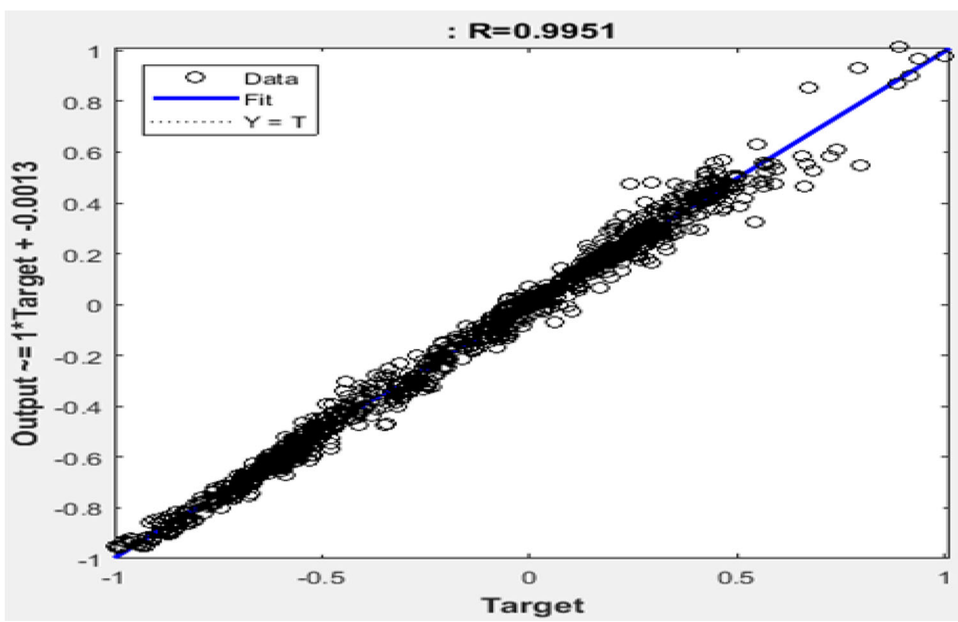


Table 17 Hybrid ANN-PSO

Symbol	MSE	RMSE	MAE	MAPE	MSRE	MARE	RMSRE	RMSPE	R^2	Best Particle
Khodro	1.0e–05	0.0004	0.0004	– 0.0001	0.0002	1.97e–06	4.0e–05	0.0042	0.995	1

Table 18 Hybrid ANN-HS

Symbol	MSE	RMSE	MAE	MAPE	MSRE	MARE	RMSRE	RMSPE
Khodro	3.09e–08	0.00001	4.70e–15	4.23e–11	6.19e–09	4.23e–13	0.000001	0.0001

Table 19 Feature selection using GWO-PSO algorithm

1: has been selected as input variable = 12
0: has not been selected as input variable = 29

Open	High	Low	WC	EMA (5)	EMA (6)	EMA (10)	EMA (20)	MACD	RS
1	0	0	0	1	1	0	1	1	0
Lowest Low	%K	SMA (5)	SMA (6)	SMA (10)	SMA (20)	TMA (5)	TMA (6)	TMA (10)	TMA (20)
0	0	1	1	0	1	1	0	0	0
ROC	MOpen	MHigh	MLow	MClose	AccOpen	AccHigh	AccLow	AccClose	AccDist
0	0	0	0	0	0	0	0	0	0
Fast %K	Fast %D	Slow %K	Slow %D	%R	RSI	Middle Band	Upper Band	Lower Band	MP
0	0	0	0	0	1	0	0	0	1
TP									
1									

Table 20 GWO-PSO feature selection results

Hybrid Acc	Hybrid Fitness	Hybrid Dimension	Hybrid Time	Number of Search Agents	Maximum Number of Iterations
1.00000	0.004615	12	13.7438	10	100

Table 21 MFO parameters

Search agents' number	30
Maximum number of iterations	1000
Upper bound	100
Lower bound	−100
Best score	**8.0081e–32**
dim	12

Bold indicates optimal solutions and obtained based on 95% significant level

Table 22 WOA parameters

Search agents' number	30
Maximum number of iterations	500
Upper bound	100
Lower bound	− 100
Best score	**1.6828e–78**
dim	12

Bold indicates optimal solutions and obtained based on 95% significant level

First, the number of iterations is assumed 1000 and in order to achieve better results, we increase it to 5000. Finally, the result after 5000 iterations is shown in Table 18.

In Table 18, the R^2 is 0.995.

4.2.4 Hybrid GWO-PSO algorithm

In this section, we would like to provide precise and general results to avoid prolonging the content. So, the following important variables are considered input variables (see Table 19).

Among these 42 indicators, 12 indicators are selected as input variables and others are not chosen. Table 20 shows the results.

4.2.5 MFO algorithm

First of all, we tune the parameters and the results are shown in Table 21. Figure 14 also shows the fitness function and convergence during iterations. You can see a clear decrease in each iteration until the best score is obtained, that is 8.0081e−32.

4.2.6 WOA

Like the MFO algorithm, first of all, we tune the parameters and the results are shown in Table 22. Figure 15 demonstrates the fitness function and convergence during iterations.

4.2.7 MPSO, MPSO-TVAC, ChO algorithms

In this part, we run three algorithms together but their results are depicted separately. As it is clear, among these three algorithms, that is, ChOA, MPSO, MPSO-TVAC, ChOA has the lowest error. Figures 16, 17, 18 and 19 show the chaotic map for types ChOA1 and ChOA2 after 500 iterations (see Table 23).

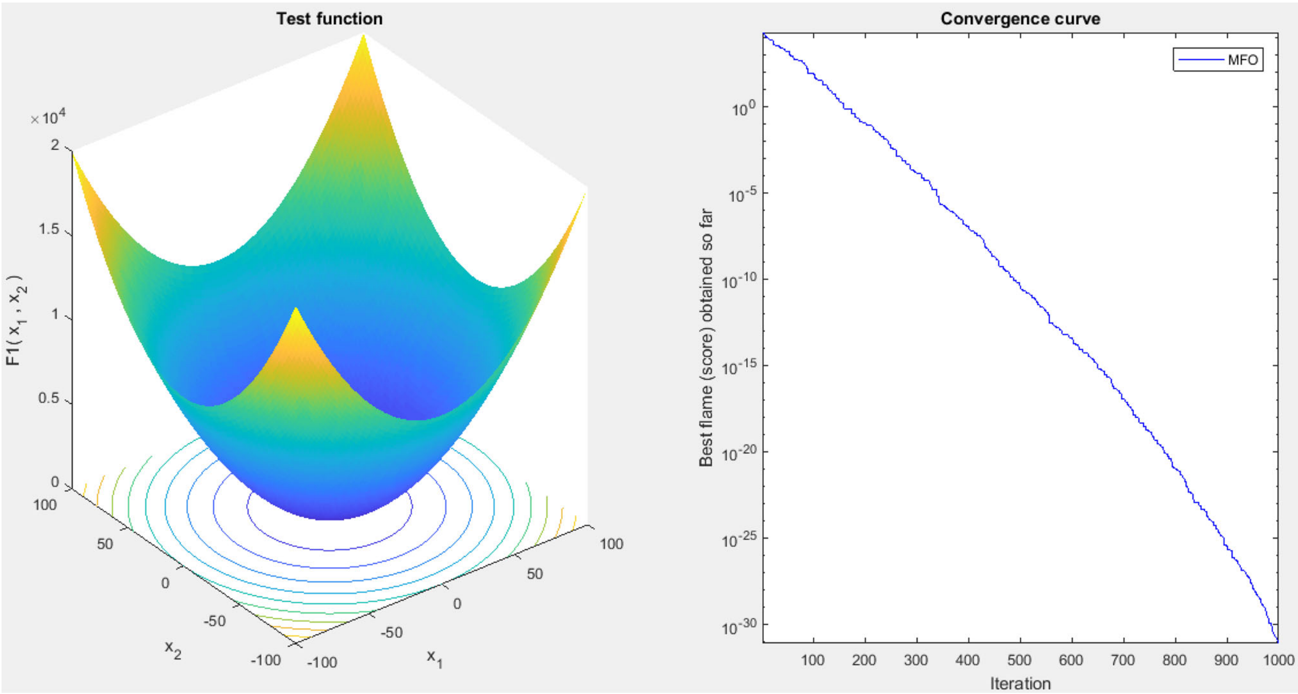


Fig. 14 Test function and convergence curve

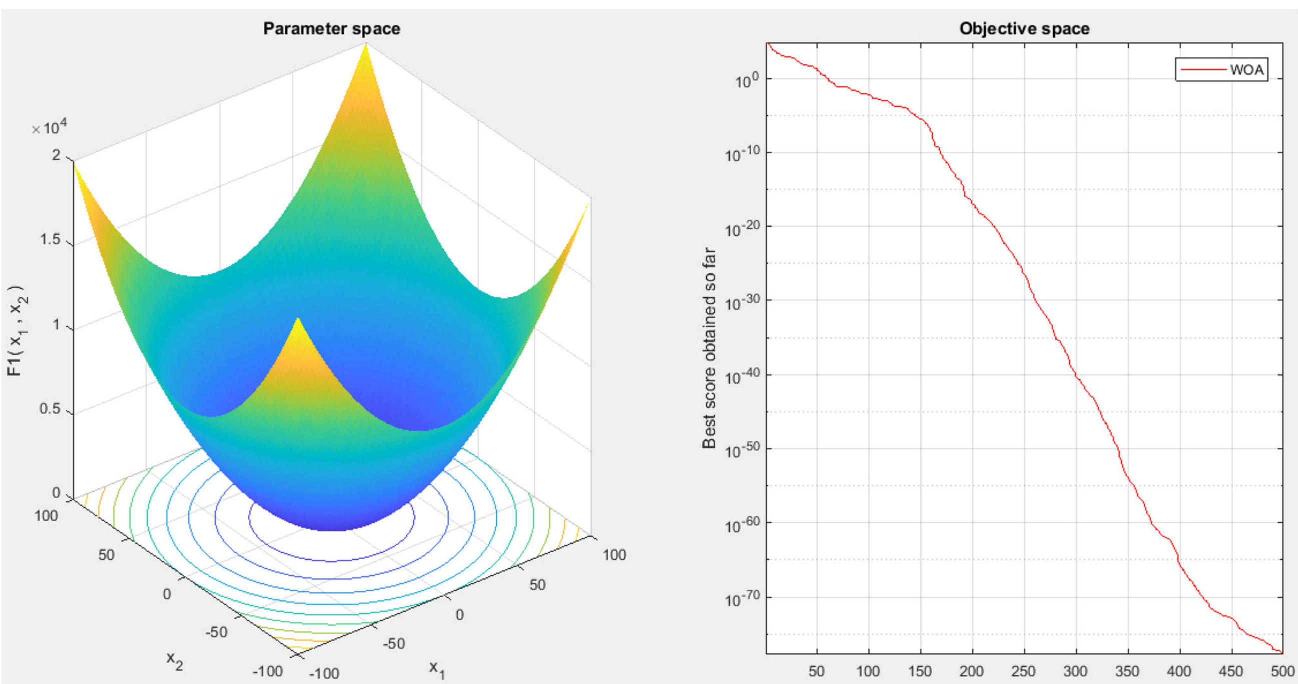


Fig. 15 Test function and convergence curve

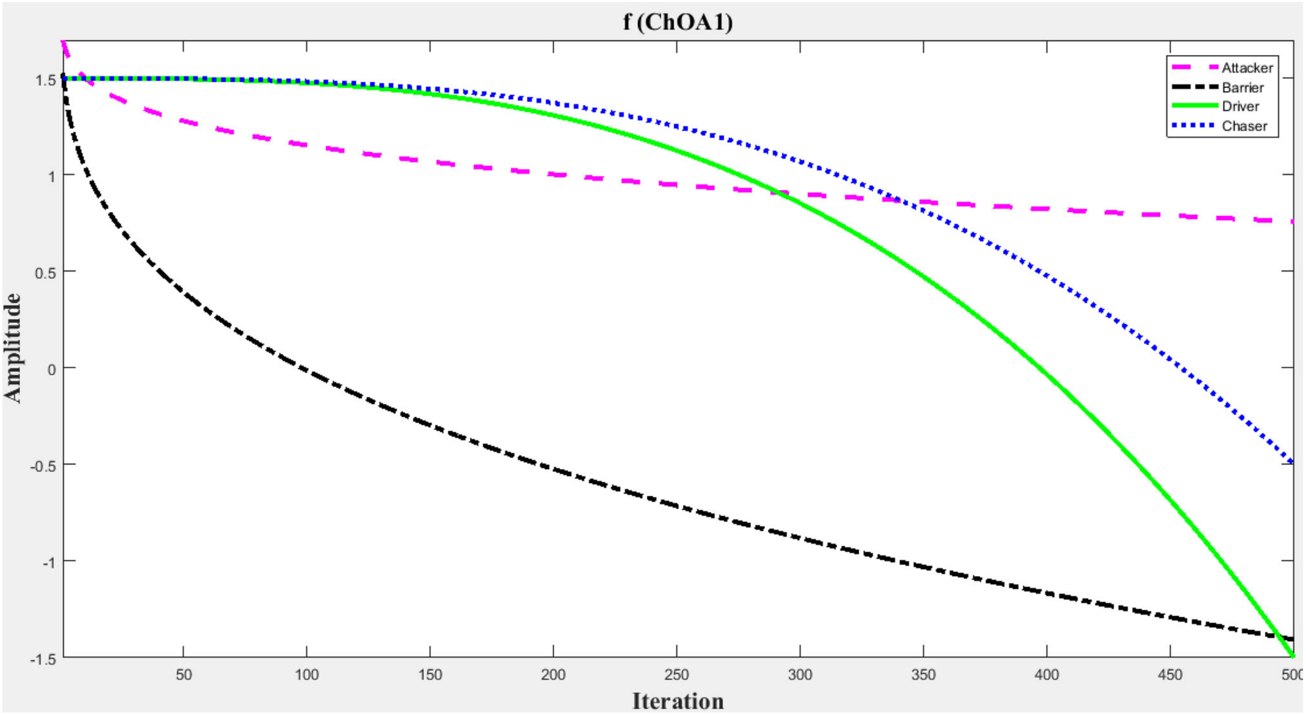


Fig. 16 Mathematical models of dynamic coefficients (f) related to independent groups for (a) ChOA1

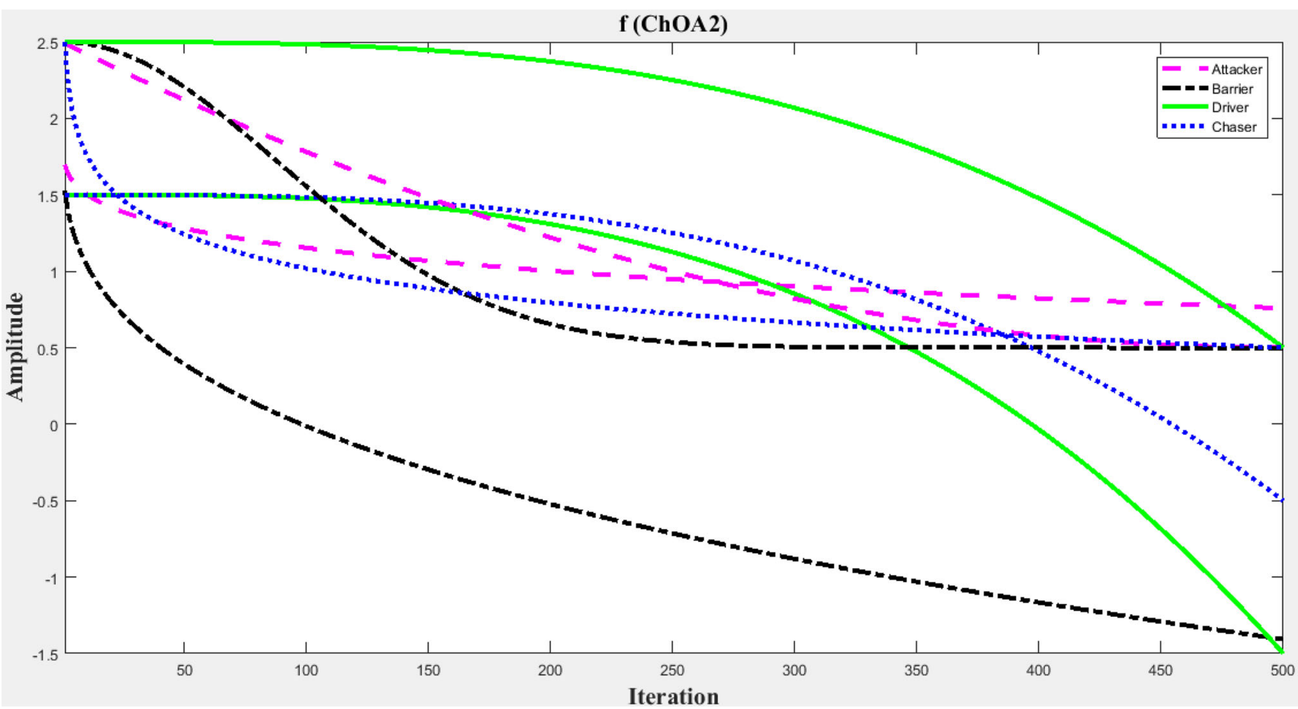


Fig. 17 Mathematical models of dynamic coefficients (f) related to independent groups for (a) ChOA1

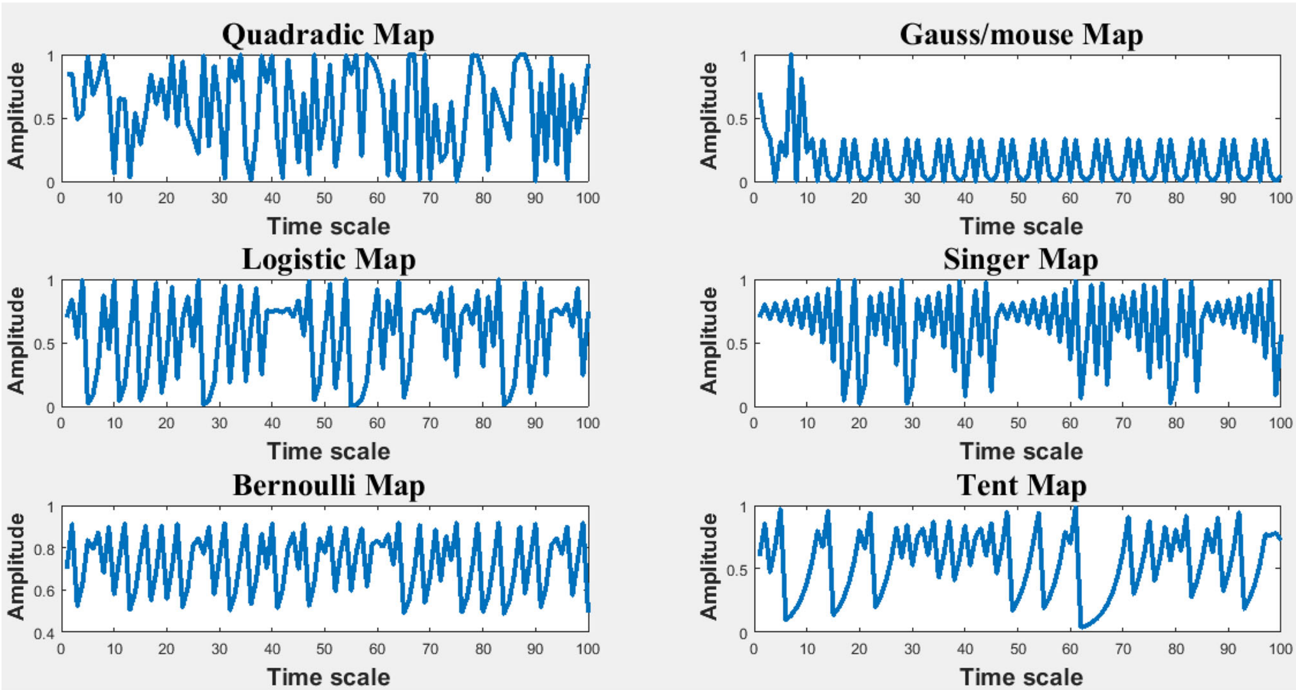


Fig. 18 Chaotic map

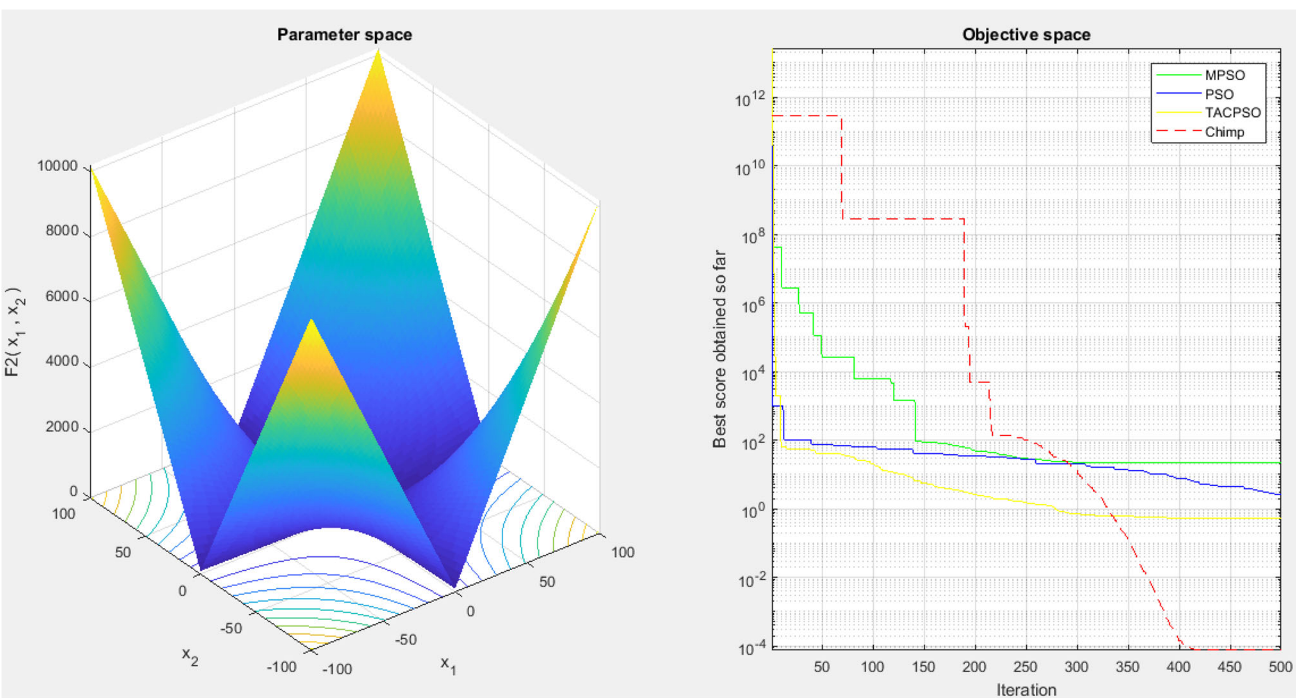
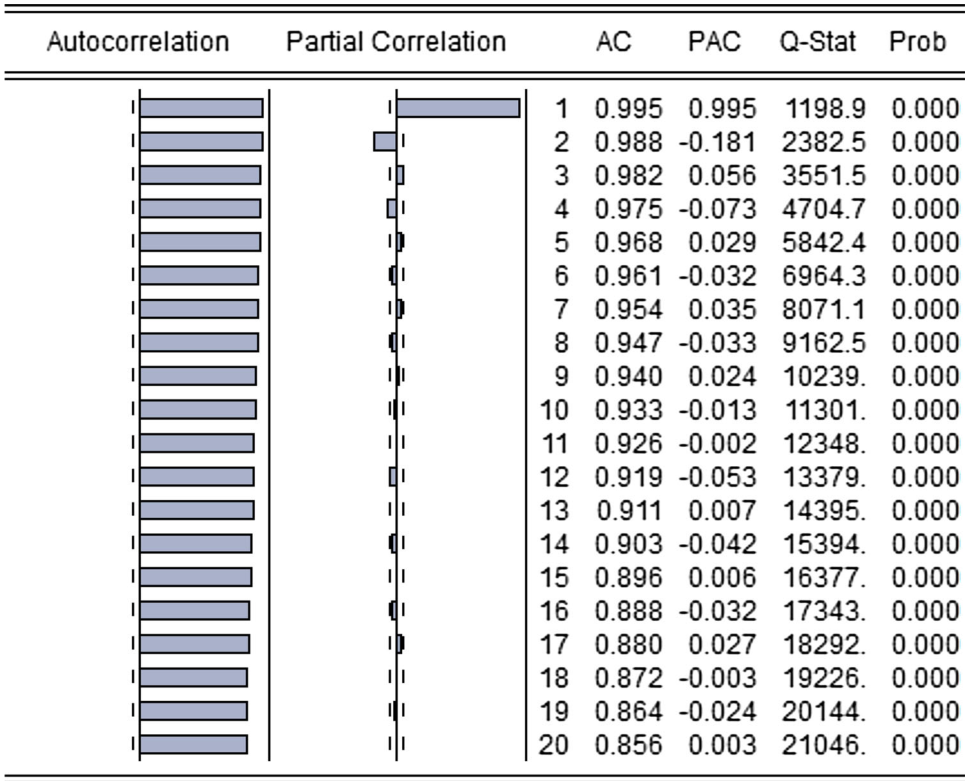
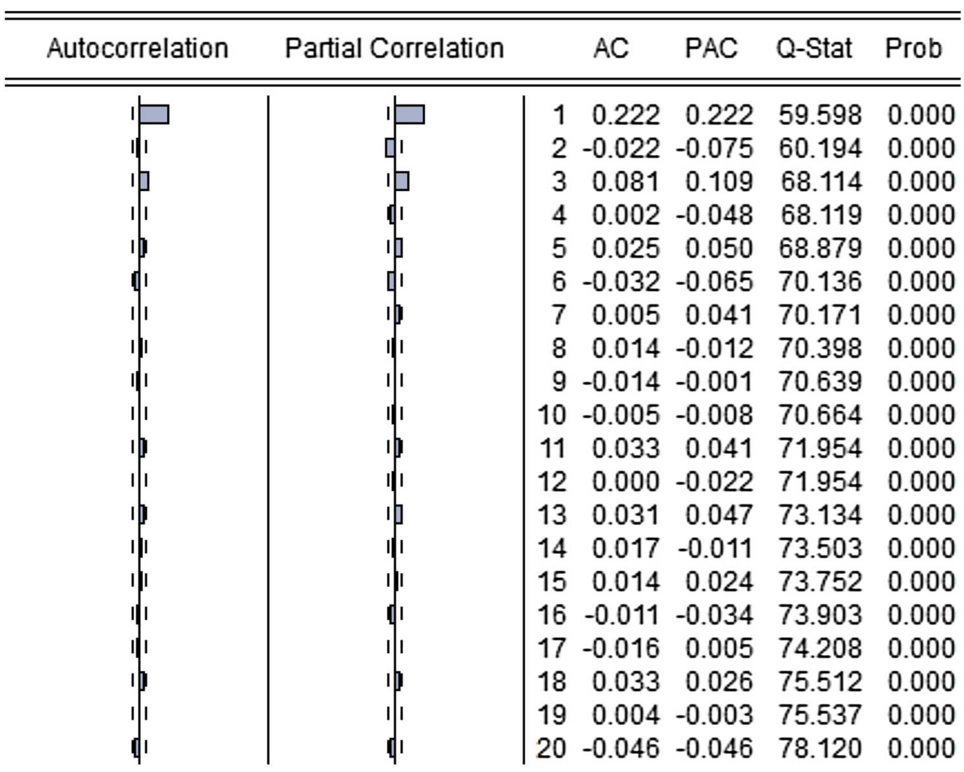


Fig. 19 Test function and convergence curve



Autocorrelation	Partial Correlation		AC	PAC	Q-Stat	Prob
		1	0.995	0.995	1198.9	0.000
		2	0.988	-0.181	2382.5	0.000
		3	0.982	0.056	3551.5	0.000
		4	0.975	-0.073	4704.7	0.000
		5	0.968	0.029	5842.4	0.000
		6	0.961	-0.032	6964.3	0.000
		7	0.954	0.035	8071.1	0.000
		8	0.947	-0.033	9162.5	0.000
		9	0.940	0.024	10239.	0.000
		10	0.933	-0.013	11301.	0.000
		11	0.926	-0.002	12348.	0.000
		12	0.919	-0.053	13379.	0.000
		13	0.911	0.007	14395.	0.000
		14	0.903	-0.042	15394.	0.000
		15	0.896	0.006	16377.	0.000
		16	0.888	-0.032	17343.	0.000
		17	0.880	0.027	18292.	0.000
		18	0.872	-0.003	19226.	0.000
		19	0.864	-0.024	20144.	0.000
		20	0.856	0.003	21046.	0.000

Fig. 20 Correlogram of closing price



Autocorrelation	Partial Correlation		AC	PAC	Q-Stat	Prob
		1	0.222	0.222	59.598	0.000
		2	-0.022	-0.075	60.194	0.000
		3	0.081	0.109	68.114	0.000
		4	0.002	-0.048	68.119	0.000
		5	0.025	0.050	68.879	0.000
		6	-0.032	-0.065	70.136	0.000
		7	0.005	0.041	70.171	0.000
		8	0.014	-0.012	70.398	0.000
		9	-0.014	-0.001	70.639	0.000
		10	-0.005	-0.008	70.664	0.000
		11	0.033	0.041	71.954	0.000
		12	0.000	-0.022	71.954	0.000
		13	0.031	0.047	73.134	0.000
		14	0.017	-0.011	73.503	0.000
		15	0.014	0.024	73.752	0.000
		16	-0.011	-0.034	73.903	0.000
		17	-0.016	0.005	74.208	0.000
		18	0.033	0.026	75.512	0.000
		19	0.004	-0.003	75.537	0.000
		20	-0.046	-0.046	78.120	0.000

Fig. 21 Correlogram after one level differencing

Table 23 Parameters and errors

Search agents' number	30
Maximum number of iterations	500
Upper bound	100
Lower bound	− 100
Best score chimp	**7.4341e–05**
Best score MPSO	**21.4027**
Best score MPSO-TVAC	**0.5373**
dim	12

Table 24 Unit root test using ADF

Null Hypothesis: _CLOSE_ has a unit root				
Exogenous: Constant				
Lag Length: 3 (Automatic—based on SIC, maxlag = 22)				
			t statistic	Prob.*
Augmented Dickey–Fuller test statistic			**− 2.108315**	0.2415
Test critical values:	1% level		− 3.435567	
	5% level		− 2.863732	
	10% level		− 2.567987	
*MacKinnon (1996) one-sided *p* values				
Augmented Dickey–Fuller test equation				
Dependent Variable: D(_CLOSE_)				
Method: least squares				
Date: 07/15/21 Time: 11:36				
Sample (adjusted): 5 1208				
Included observations: 1204 after adjustments				
Variable	Coefficient	Std. error	*t* statistic	Prob
CLOSE (-1)	− 0.005292	0.002510	− 2.108315	0.0352
D (_CLOSE_ (-1))	0.248784	0.028670	8.677486	0.0000
D (_CLOSE_ (-2))	− 0.098467	0.029402	− 3.348986	0.0008
D (_CLOSE_ (-3))	0.111819	0.028700	3.896088	0.0001
C	89.59021	41.25284	2.171735	0.0301
R-squared	0.069290	Mean dependent var		6.627243
Adjusted R-squared	0.066185	S.D. dependent var		343.3799
S.E. of regression	331.8221	Akaike info criterion		14.45122
Sum squared resid	1.32E + 08	Schwarz criterion		14.47237
Log likelihood	− 8694.634	Hannan–Quinn criter		14.45919
F statistic	22.31607	Durbin–Watson stat		1.989850
Prob(F statistic)	0.000000			

Bold indicates optimal solutions and obtained based on 95% significant level

Table 25 ADF test after one level differencing

Null Hypothesis: D(_CLOSE_) has a unit root				
Exogenous: Constant				
Lag Length: 2 (Automatic—based on SIC, maxlag = 22)				
			t statistic	Prob.*
Augmented Dickey–Fuller test statistic			**– 17.31204**	0.0000
Test critical values:	1% level		– 3.435567	
	5% level		– 2.863732	
	10% level		– 2.567987	
*MacKinnon (1996) one-sided p values				
Augmented Dickey–Fuller Test Equation				
Dependent Variable: D(_CLOSE_,2)				
Method: Least Squares				
Date: 07/15/21 Time: 11:44				
Sample (adjusted): 5 1208				
Included observations: 1204 after adjustments				
Variable	Coefficient	Std. Error	*t* Statistic	Prob
D (_CLOSE_ (– 1))	– 0.745099	0.043039	– 17.31204	0.0000
D (_CLOSE_ (– 1),2)	– 0.008025	0.036020	– 0.222789	0.8237
D (_CLOSE_ (– 2),2)	– 0.108945	0.028709	– 3.794792	0.0002
C	4.987406	9.580677	0.520569	0.6028
R-squared	0.399638	Mean dependent var		0.001578
Adjusted R-squared	0.398137	S.D. dependent var		428.3305
S.E. of regression	332.2980	Akaike info criterion		14.45326
Sum squared resid	1.33E + 08	Schwarz criterion		14.47018
Log likelihood	– 8696.861	Hannan–Quinn criter		14.45963
F statistic	266.2648	Durbin–Watson stat		1.989520
Prob(F statistic)	0.000000			

You can see that chaser with driver and attacker with barrier almost have the same behavior but as we stated previously, they follow different strategies.

In ChOA2, it is clear that in iteration 400, three groups including attacker, barrier and chaser are closed to each other.

It goes without saying that among these three algorithms, ChOA, MPSO-TVAC, and MPSO have the lowest error and optimal solutions, respectively. ChOA has a very sharp decline compared to other algorithms.

4.3 Time series forecasting (ARIMA)

Most of the time, the economic and financial time series are not normal and they have some characteristics such as skewness and kurtosis. So, we should check if the time series is stationary or not. For this purpose, we used the augmented dicky fuller (ADF) test for testing stationarity. One of the main methods which can show the existence of a unit root is a correlogram plot. The results are presented in Fig. 20 and Table 24.

As it is clear, there is at least one-unit root. Further results and details can be obtained by ADF.

From Table 25, we can see that *t* statistic (i.e., – 2.108315) is higher than critical values in 1%, 5%, and 10% significance levels. Thus, the time series is not stationary and we have to solve it with one level differencing.

Now, we can see that *t* statistic (i.e., -17.31204) is less than critical values in all three significance lev3els. So, the series is stationary. Figure 21 presents more details.

Now, we can use ARIMA as a prediction model. We used Eviews10 as a tool for computation. The best model estimation is presented in Table 26. The model selection criteria are summarized in Table 27. Also, Fig. 30 illustrates the Akaike information criteria while Table 28 illustrates the ARIMA forecasting summary (See Fig. 22).

Table 26 ARIMA forecasting

Dependent Variable: DLOG(_CLOSE_)				
Method: ARMA Maximum Likelihood (BFGS)				
Date: 07/15/21 Time: 12:28				
Sample: 2 1208				
Included observations: 1207				
Convergence achieved after 25 iterations				
Variable	Coefficient	Std. Error	*t* statistic	Prob
C	0.000460	0.000764	0.602433	0.5470
AR (1)	− 0.718797	0.022972	- 31.29068	0.0000
AR (2)	0.131919	0.028090	4.696380	0.0000
AR (3)	0.037372	0.030073	1.242728	0.2142
AR (4)	0.110730	0.023361	4.739956	0.0000
MA (1)	0.988479	0.008005	123.4849	0.0000
SIGMASQ	0.000355	1.14E-05	31.18357	0.0000
R-squared	0.087769	Mean dependent var		0.000458
Adjusted R-squared	0.083208	S.D. dependent var		0.019733
S.E. of regression	0.018894	Akaike info criterion		− 5.093621
Sum squared resid	0.428369	Schwarz criterion		− 5.064068
Log likelihood	3081.001	Hannan–Quinn criter		− 5.082492
F statistic	19.24263	Durbin–Watson stat		1.996982
Prob(*F* statistic)	0.000000			
Inverted AR Roots	0.51	− 0.14-0.46i	− 0.14 + 0.46i	− 0.95
Inverted MA Roots	− 0.99			

From Table 28, we can find that the best ARIMA selected model is (4.1.1) with AIC value -5.0936.

4.4 Testing EMH

At first, we need to check the normality. So, we used the Kolmogorov–Smirnov normality test (see Table 29).

The value of Sigma is less than 0.05 which means that the time series is not normal. So, it is possible to use the non-parametric test. It means that we should run the test for checking the EMH (see Table 30).

Sigma is less than %0.05 which means that data are not random. So, the market is not efficient.

4.5 Comparative study

In this section, we have reviewed some similar articles and compared our results with them in a table format (see Table 31). For better understanding of the results, we order the methods in accordance with their MSE (from minimum to maximum error).

Table 27 Model selection criteria

Model selection criteria table				
Dependent Variable: DLOG(_CLOSE_)				
Date: 07/15/21 Time: 12:28				
Sample: 1 1208				
Included observations: 1207				
Model	LogL	AIC*	BIC	HQ
(4,1)	3081.000503	− 5.093621	− 5.064068	− 5.082492
(2,4)	3081.726732	− 5.093168	− 5.059392	− 5.080448
(4,2)	3081.152531	− 5.092216	− 5.058441	− 5.079497
(3,4)	3081.958798	− 5.091895	− 5.053898	− 5.077586
(4,3)	3081.444196	− 5.091043	− 5.053045	− 5.076733
(4,4)	3082.051398	− 5.090392	− 5.048172	− 5.074493
(0,3)	3075.149728	− 5.087241	− 5.066131	− 5.079291
(1,2)	3074.860019	− 5.086761	− 5.065651	− 5.078811
(1,3)	3075.397373	− 5.085994	− 5.060662	− 5.076454
(2,2)	3075.341047	− 5.085901	− 5.060569	− 5.076361
(0,4)	3075.319221	− 5.085864	− 5.060533	− 5.076325
(3,1)	3074.517887	− 5.084537	− 5.059205	− 5.074997
(2,3)	3075.504936	− 5.084515	− 5.054962	− 5.073386
(1,4)	3075.414474	− 5.084365	− 5.054812	− 5.073236
(3,2)	3075.356681	− 5.084270	− 5.054716	− 5.073140
(3,3)	3076.307434	− 5.084188	− 5.050412	− 5.071469
(2,1)	3073.056663	− 5.083772	− 5.062663	− 5.075823
(1,1)	3071.514326	− 5.082874	− 5.065986	− 5.076514
(3,0)	3072.089935	− 5.082171	− 5.061061	− 5.074221
(4,0)	3072.846306	− 5.081767	− 5.056435	− 5.072227
(0,2)	3067.172625	− 5.075680	− 5.058792	− 5.069320
(0,1)	3062.956734	− 5.070351	− 5.057685	− 5.065581
(2,0)	3060.396382	− 5.064451	− 5.047564	− 5.058092
(1,0)	3056.838962	− 5.060214	− 5.047548	− 5.055444
(0,0)	3025.903915	− 5.010611	− 5.002167	− 5.007431

Table 28 ARIMA forecasting summary

Automatic ARIMA Forecasting
Selected dependent variable: DLOG(_CLOSE_)
Date: 07/15/21 Time: 12:28
Sample: 1 1208
Included observations: 1207
Forecast length: 0
Number of estimated ARMA models: 25
Number of non-converged estimations: 0
Selected ARMA model: (4,1)
AIC value: − 5.0936213805

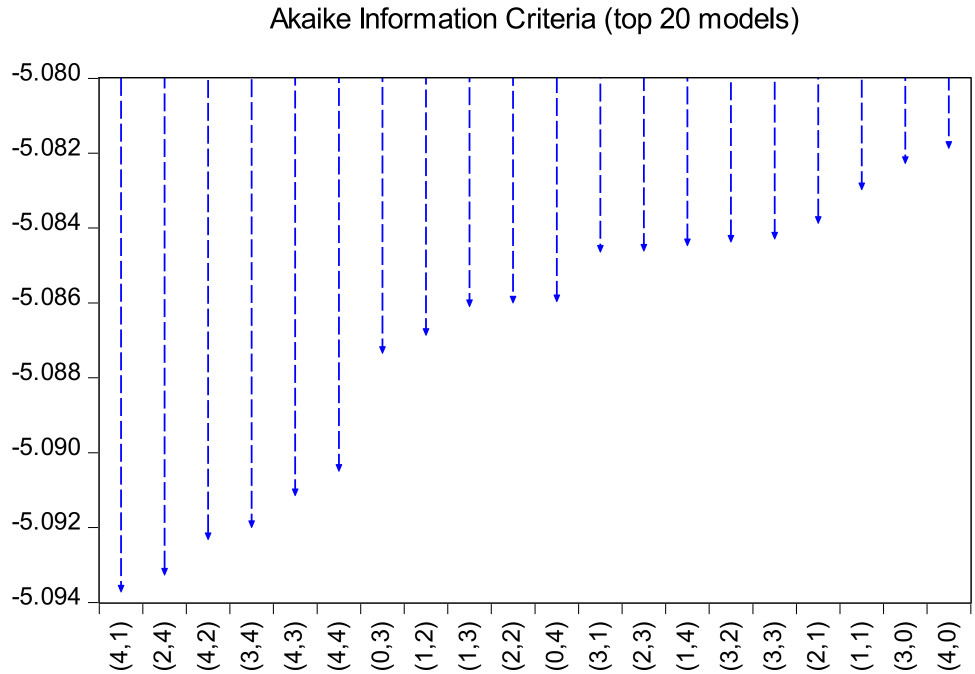


Fig. 22 Akaike information criteria (top 20 models)

Table 29 Testing normality using K-S test

Khodro						
	Kolmogorov–Smirnov[a]			Shapiro–Wilk		
	Statistic	df	**Sig**	Statistic	df	Sig
Adj.Closing Price;	0.097	1086	0.0**00**	0.963	1086	0.000

Table 30 Run test

Khodro	
	Adj.Closing Price;
Test Value[a]	– 0.16692865
Cases < Test Value	504
Cases > = Test Value	582
Total Cases	1086
Number of Runs	12
Z	– 32.299
Asymp. Sig. (two tailed)	0.000

5 Conclusions

In this paper, we used an artificial neural network as a prediction method to forecast Khodro stock prices. In this regard, we used a couple of important technical indicators such as SMA, EMA, and TMA as input variables. At this point, we selected the most important ones by using GA and GWO-PSO. Afterward, we trained the network using different meta-heuristic algorithms such as HS, PSO, MFO, MPSO, MPSO-TVAC, WOA, CHOA, and a time series model called ARIMA.

After obtaining optimum indicators and weights by GA and GWO-PSO, we computed different loss functions for each algorithm. As it can be concluded from Table 32, WOA and MPSO have the lowest and highest training and testing error, respectively. For evaluating the performance of the model, we should test it with a new set of data called testing performance data. Finally, we analyzed the EMH and the results showed that the market is inefficient. The main

Table 31 Comparative Study

Author	Proposed Approaches	Type of Data	MSE	MAE	R^2
Ghasemiyeh, et al. [38]	GA-ANN	Train	0.00442	0.0194	0.9866
		Test	0.00869	0.00902	0.9895
	PSO-ANN	Train	0.002410	0.04910	0.9972
		Test	0.00015	0.00260	0.9969
	PSO-ANN	Train	0.00076	0.00450	0.9966
		Test	0.0068	0.00694	0.9995
Sedighi et al. [53]	ARIMA-SVM	Final Outcome	0.1548	0.0142	0.9691
	SVM-RF	Final Outcome	0.0000475	0.00726	0.9875
	ANFIS-SVM	Final Outcome	0.01518	0.0268	0.9961
	FA-MSVR	Final Outcome	0.00014	0.00130	0.9982
Safa and Panahian [54]	HS-ANN	Final Outcome	0.00036	0.00517	0.9641
Emamverdi et al. [55]	ANN	Final Outcome	0.00030	0.0174	0.9791
	ARIMA	Final Outcome	0.0121	0.0561	0.9689
Farahani and Hajiagha [29]					
	ANN	Train	–	12.1827	0.9975
		Test		13.499	
	GA-ANN	Train	–	10.8316	0.9988
		Test		19.7717	
	BA	Final Outcome	–	1.0E–40	0.9993
	SSO	Final Outcome	–	1.0E–52	0.999
	ARIMA	Final Outcome	–	0.071284	0.6028
Current research	ANN	Train	0.01768	0.036408	0.9973
		Test	0.06578	0.00621	
	GA-ANN	Train	0.00070	0.0130	0.9984
		Test	0.00045	0.000532	
	PSO-ANN	Train	0.000022	0.00392	0.99
		Test	0.00431	0.000216	
	HS-ANN	Train	3.0258E–07	5.366E–15	0.99
		Test	0.000061402	0.00042	
	MPSO	Final Outcome	21.4027	–	0.9793
	MPSO-TVAC	Final Outcome	0.5373	–	0.9895
	ChOA	Final Outcome	7.4341e–05	–	0.9989
	WOA	Final Outcome	1.6828e–78	–	0.9995
	MFO	Final Outcome	8.0081e–32	–	0.9989
	GWO-PSO	Final Outcome	0.004615	–	0.9897

advantages obtained by using meta-heuristic algorithms are as follows:

- Speeding up calculations.
- Reducing the model complexity.
- Increasing the network accuracy.
- Ease of using models.

On the other hand, as we mentioned earlier, these algorithms have some limitations:

- These algorithms are sensitive to the value of their parameters. As such, these parameters should be tuned before ahead. In other words, setting parameters and assigning suitable values to each one can affect the outputs. Thus, if the tuning phase is not performed correctly, your model will face serious problems.
- Another limitation of these algorithms (especially evolutionary algorithms) is that most of them fall into local optimum. In other words, there is no guarantee for global

Table 32 Arrange algorithms based on MSE

ROW	Algorithm	MSE
1	WOA	1.6828e–78
2	MFO	8.0081e–32
3	HS-ANN	3.0258E–07
4	ChOA	7.4341e–05
5	PSO-ANN	0.000022
6	GA-ANN	0.00045
7	GWO-PSO	0.004615
8	ANN	0.01768
9	MPSO-TVAC	0.5373
10	MPSO	21.4027

optimally. As a result, most of these algorithms have different strategies for exploitation and exploration. They have different approaches for generating the initial population, finding an optimal solution, etc.

- The next limitation of these algorithms is that the obtained solutions are not repeatable. Each time you run these algorithms; you may reach different solutions.

So, in this research, we used different approaches to overcome the limitations of each algorithm and compared them to each other.

Our suggestion for future research is to concentrate on other parameters such as the number of hidden layers and activation function and to apply other models of HS such as HIS. In addition, researchers can train neural networks or select features with other new metaheuristic algorithms such as the bald eagle algorithm (BEA), sparrow search algorithm (SSA), Lichtenberg algorithm (LA), and so forth. Furthermore, we believe the prediction of crypto price by using these algorithms and other AI-based methods such as deep learning and fuzzy logic could be a good idea for future research.

Author contributions This paper is handled and done by Milad Shahvaroughi Farahani. In this paper, Hamed Farrokhi-asl who is added as an author to the paper helped us in improving the paper and doing revisions.

Funding This research paper is not supported by any organization. Any organization or employment will not gain or loss financially through publication of this manuscript. The authors declare they have no financial interests.

Declarations

Conflict of interest/Competing interests The authors have no conflicts of interest to declare that are relevant to the content of this article.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Availability of data and material The data were obtained through two sites which are called TSETMC and CODAL sites, respectively Which are at the following address: http://tsetmc.ir/, https://www.codal.ir/. On the other hand, there is a financial data software which is called TSE-CLIENT 2.0 and you can download data easily according to symbol, date, different variables and etc.

Code availability All of the necessary pseudo-codes are mentioned in the manuscript.

Appendix

See Tables 33, 34 and Fig. 23.

Table 33 Input and target correlation

Variables	Type	Closing price
Typical price	Linear	0.999785
Low	Linear	0.999391
High	Linear	0.999274
Open	Linear	0.998081
TMA3	Linear	0.997881
TMA5	Linear	0.997510
TMA6	Linear	0.997166
TMA10	Linear	0.995414
SMA3	Linear	0.993949
TMA20	Linear	0.991306
SMA5	Linear	0.990946
SMA6	Linear	0.989483
SMA10	Linear	0.984243
10Day EMA	Linear	0.980144
6Day EMA	Linear	0.980065
5Day EMA	Linear	0.980036
Lowest Low	Linear	0.979325
26Day EMA	Linear	0.979263
20Day EMA	Linear	0.979201
Highest High	Linear	0.975629
SMA20	Linear	0.973401
Money Flow	Linear	0.633487
Average True Range	Linear	0.623142
Vol	Linear	0.488431
True Range	Linear	0.331648
Chaikin Oscillator	Linear	0.282556
3Day EMA	Linear	− 0.227193
10Day EMA	Linear	− 0.222457
ADL	Linear	− 0.216619
MACD	Linear	0.165757
OBV	Linear	0.164625
RSI	Linear	0.135669
Money Ratio_26	Linear	0.091625
12Day ROC	Linear	0.072828
%D	Linear	0.072211
%K	Linear	0.062261
Money Flow Volume	Linear	− 0.052864
RS	Linear	0.046729
Money Ratio	Linear	0.026702
PX change	Linear	0.021243
Money Flow Multiplier	Linear	0.007488

Table 34 Training strategy

Criteria	Description	Value
Inverse Hessian approximation method	Method used to obtain a suitable training rate	BFGS
Training rate method	Method used to calculate the step for the quasi-Newton training direction	Brent Method
Loss tolerance	Maximum interval length for the training rate	0.001
Minimum parameters increment norm	Norm of the parameters increment vector at which training stops	1e-09
Minimum loss decrease	Minimum loss improvement between two successive epochs	1e–12
Loss goal	Goal value for the loss	1e–12
Gradient norm goal	Goal value for the norm of the objective function gradient	0.001
Maximum selection error increases	Maximum number of epochs at which the selection error increases	100
Maximum iterations number	Maximum number of epochs to perform the training	1000
Maximum time	Maximum training time	3600
Reverse parameters norm history	Plot a graph with the parameter's norm of each iteration	False
Reverse error history	Plot a graph with the loss of each iteration	True
Reverse selection error history	Plot a graph with the selection error of each iteration	True
Reverse gradient norm history	Plot a graph with the gradient norm of each iteration	False

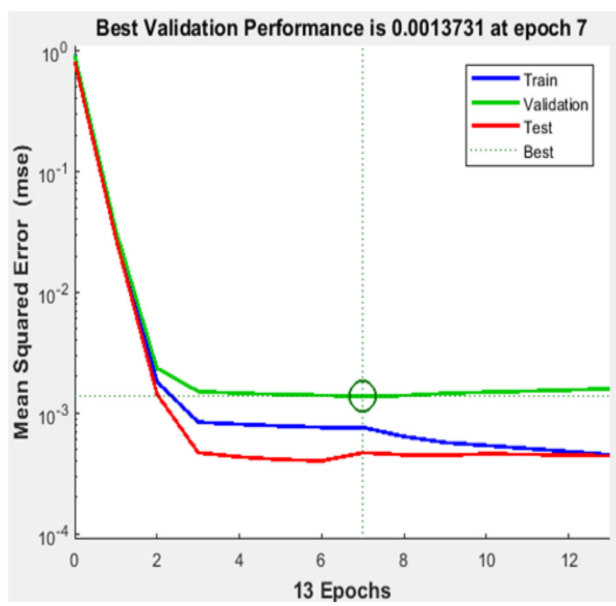


Fig. 23 The best validation performance(Khodro)

References

1. Bouattour, M., Martinez, I.: Efficient market hypothesis: an experimental study with uncertainty and asymmetric information. Finance Contrôle Stratég. **22**(4), 22–24 (2019)
2. Sánchez-Granero, M.A., Balladares, K.A., Ramos-Requena, J.P., Trinidad-Segovia, J.E.: Testing the efficient market hypothesis in Latin American stock markets. Phys. A **540**, 123082 (2020)
3. Hiransha, M., Gopalakrishnan, E.A., Menon, V.K., Soman, K.P.: NSE stock market prediction using deep-learning models. Proc. Compu.t Sci. **132**, 1351–1362 (2018)
4. Drożdż, S., Kwapień, J., Oświęcimka, P.: Complexity in economic and social systems. Entropy **23**(2), 133 (2021)
5. Sohangir, S., Wang, D., Pomeranets, A., Khoshgoftaar, T.M.: Big Data: Deep Learning for financial sentiment analysis. J. Big Data **5**(1), 1–25 (2018)
6. Jóhannsson, Ó.S.: Forecasting the Icelandic stock market using a neural network (Doctoral dissertation) (2020)
7. Krauss, C., Do, X.A., Huck, N.: Deep neural networks, gradient-boosted trees, random forests: statistical arbitrage on the S&P 500. Eur. J. Oper. Res. **259**(2), 689–702 (2017)
8. Weng, B., Martinez, W., Tsai, Y.T., Li, C., Lu, L., Barth, J.R., Megahed, F.M.: Macroeconomic indicators alone can predict the monthly closing price of major US indices: Insights from artificial intelligence, time-series analysis and hybrid models. Appl. Soft Comput. **71**, 685–697 (2018)
9. Siami-Namini, S., Namin, A.S.: Forecasting economics and financial time series: ARIMA vs. LSTM. arXiv preprint arXiv:1803.06386 (2018)
10. Bahmani-Oskooee, M., Hasanzade, M., Bahmani, S.: Stock returns and income inequality: asymmetric evidence from state level data in the US. Glob. Financ. J. **52**, 100715 (2022)
11. Farahani, M.S.: Prediction of interest rate using artificial neural network and novel meta-heuristic algorithms. Iran. J. Account. Audit. Finance (IJAAF) **5**(1), 1–30 (2021)
12. Bahmani-Oskooee, M., Hasanzade, M.: Asymmetric link between US tariff policy and income distribution: evidence from state level data. Open Econ. Rev. **31**(4), 821–857 (2020)
13. Peykani, P., Nouri, M., Eshghi, F., Khamechian, M., Farrokhi-Asl, H.: A novel mathematical approach for fuzzy multi-period multi-objective portfolio optimization problem under uncertain environment and practical constraints. J. Fuzzy Ext. Appl. **2**(3), 191–203 (2021)
14. Fama, E.F.: Market Efficiency, Long-Term Returns, and Behavioral Finance, pp. 174–200. University of Chicago Press (2021)
15. Shah, D., Isah, H., Zulkernine, F.: Stock market analysis: a review and taxonomy of prediction techniques. Int. J. Financ. Stud. **7**(2), 26 (2019)
16. Kaya, A., Kaya, G., Çebi, F.: Forecasting automobile sales in Turkey with artificial neural networks. Int. J. Bus. Anal. (IJBAN) **6**(4), 50–60 (2019)
17. Charef, F., Ayachi, F.: Non-linear causality between exchange rates, inflation, interest rate differential and terms of trade in Tunisia. Afr. J. Econ. Manag. Stud. (2018)
18. Göçken, M., Özçalıcı, M., Boru, A., Dosdoğru, A.T.: Integrating metaheuristics and artificial neural networks for improved stock price prediction. Expert Syst. Appl. **44**, 320–331 (2016)
19. Qiu, M., Song, Y., Akagi, F.: Application of artificial neural network for the prediction of stock market returns: The case of the Japanese stock market. Chaos Solitons Fractals **85**, 1–7 (2016)
20. Hassanin, M.F., Shoeb, A.M., Hassanien, A.E.: Grey wolf optimizer-based back-propagation neural network algorithm. In: Paper presented at the 2016 12th International Computer Engineering Conference (ICENCO), pp. 213–218 (2016)
21. Faris, H., Aljarah, I., Mirjalili, S.: Training feedforward neural networks using multi-verse optimizer for binary classification problems. Appl. Intell. **45**(2), 322–332 (2016)
22. Rather, A.M., Sastry, V., Agarwal, A.: Stock market prediction and Portfolio selection models: a survey. Opsearch **54**(3), 558–579 (2017)
23. Chong, E., Han, C., Park, F.C.: Deep learning networks for stock market analysis and prediction: methodology, data representations, and case studies. Expert Syst. Appl. **83**, 187–205 (2017)
24. Sezer, O.B., Ozbayoglu, M., Dogdu, E.: A Deep neural-network based stock trading system based on evolutionary optimized technical analysis parameters. Proc. Comput. Sci. **114**, 473–480 (2017)
25. Di Persio, L., Honchar, O.: Recurrent neural networks approach to the financial forecast of Google assets. Int. J. Math. Comput. Simul., **11**, 7–13 (2017)
26. Ahmed, M.K., Wajiga, G.M., Blamah, N.V., Modi, B.: Stock market forecasting using ant colony optimization based algorithm. Am. J. Math. Comput. Modell. **4**(3), 52–57 (2019)
27. Ghanbari, M., & Arian, H.: Forecasting stock market with support vector regression and butterfly optimization algorithm. arXiv preprint arXiv:1905.11462. (2019)
28. Kumar, G., Jain, S., Singh, U.P.: Stock market forecasting using computational intelligence: a survey. Arch. Comput. Methods Eng. **28**(3), 1–33 (2020)
29. Farahani, M.S., Hajiagha, S.H.R.: Forecasting stock price using integrated artificial neural network and metaheuristic algorithms compared to time series models. Soft Comput., pp. 1–31 (2021)
30. Pierdzioch, C., Risse, M.: A machine-learning analysis of the rationality of aggregate stock market forecasts. Int. J. Financ. Econ. **23**(4), 642–654 (2018)
31. Zhong, X., Enke, D.: Predicting the daily return direction of the stock market using hybrid machine learning algorithms. Financ. Innov. **5**(1), 1–20 (2019)
32. Altan, A., Karasu, S., Bekiros, S.: Digital currency forecasting with chaotic meta-heuristic bio-inspired signal processing techniques. Chaos Solitons Fractals **126**, 325–336 (2019)
33. Jiang, M., Jia, L., Chen, Z., Chen, W.: The two-stage machine learning ensemble models for stock price prediction by combining mode decomposition, extreme learning machine and improved harmony search algorithm. Ann. Oper. Res. **309**, 1–33 (2020)
34. Behravan, I., Razavi, S.M.: Stock price prediction using machine learning and swarm intelligence. J. Electric. Comput. Eng. Innov. (JECEI) **8**(1), 31–40 (2020)
35. Chandar, S.K.: Grey Wolf optimization-Elman neural network model for stock price prediction. Soft. Comput. **25**(1), 649–658 (2021)
36. Chen, W., Jiang, M., Zhang, W.G., Chen, Z.: A novel graph convolutional feature based convolutional neural network for stock trend prediction. Inf. Sci. **556**, 67–94 (2021)
37. Kumar, C.S.: Hybrid models for intraday stock price forecasting based on artificial neural networks and metaheuristic algorithms. Pattern Recognit. Lett. **147**, 124–133 (2021)
38. Ghasemiyeh, R., Moghdani, R., Sana, S.S.: A hybrid artificial neural network with metaheuristic algorithms for predicting stock price. Cybern. Syst. **48**(4), 365–392 (2017)
39. Bilski, J., Kowalczyk, B., Marchlewska, A., Zurada, J.M.: Local Levenberg-Marquardt algorithm for learning feedforwad neural networks. J. Artif. Intell. Soft Comput. Res., **10**, 299–316 (2020)
40. Al-Tashi, Q., Kadir, S.J.A., Rais, H.M., Mirjalili, S., Alhussian, H.: Binary optimization using hybrid grey wolf optimization for feature selection. IEEE Access **7**, 39496–39508 (2019)
41. Davallou, M., Azizi, N.: The Investigation of Information Risk Pricing; Evidence from Adjusted Probability of Informed Trading Measure. Financ. Res. J. **19**(3), 415–438 (2017)

42. Kaveh, A.: Particle swarm optimization. In: Advances in Metaheuristic Algorithms for Optimal Design of Structures, pp. 11–43. Springer, Cham (2017)
43. Ouyang, H.B., Gao, L.Q., Li, S., Kong, X.Y., Wang, Q., Zou, D.X.: Improved harmony search algorithm: LHS. Appl. Soft Comput. **53**, 133–167 (2017)
44. Wang, C., Song, W.: A modified particle swarm optimization algorithm based on velocity updating mechanism. Ain Shams Eng. J. **10**(4), 847–866 (2019)
45. He, J., Guo, H.: A modified particle swarm optimization algorithm. TELKOMNIKA Indones. J. Electric. Eng. **11**(10), 6209–6215 (2013). (**e-ISSN: 2087-278X**)
46. Abdullah, M.N., Bakar, A.H.A., Rahim, N.A., Mokhlis, H., Illias, H.A., Jamian, J.J.: Modified particle swarm optimization with time varying acceleration coefficients for economic load dispatch with generator constraints. J. Electric. Eng. Technol. **9**(1), 15–26 (2014)
47. Mirjalili, S.: Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl.-Based Syst. **89**, 228–249 (2015)
48. Mirjalili, S., Lewis, A.: The whale optimization algorithm. Adv. Eng. Softw. **95**, 51–67 (2016)
49. Khishe, M., & Mosavi, M. R. (2020). Chimp optimization algorithm. Expert Syst. Appl., p. 113338.
50. Wadi, S.A.L., Almasarweh, M., Alsaraireh, A.A., Aqaba, J.: Predicting closed price time series data using ARIMA Model. Mod. Appl. Sci., **12**(11), 181–185 (2018)
51. Pervez, M., Rashid, M., Ur, H., Chowdhury, M., Iqbal, A., Rahaman, M.: Predicting the Stock market efficiency in weak form: a study on Dhaka Stock Exchange (2018)
52. Hawaldar, I.T., Rohith, B., Pinto, P.: Testing of weak form of efficient market hypothesis: evidence from the Bahrain Bourse. Invest. Manag. Financ. Innov. **14**(2–2), 376–385 (2017)
53. Sedighi, M., Jahangirnia, H., Gharakhani, M., Farahani Fard, S.: A novel hybrid model for stock price forecasting based on metaheuristics and support vector machine. Data **4**(2), 75 (2019)
54. Safa, M., Panahian, H.: Ranking P/E predictor factors in Tehran stock exchange with using the harmony search meta heuristic algorithm. pp. 67–82 (2019)
55. Emamverdi, G., Karimi, M.S., Khakie, S., Karimi, M.: Forecastinh The Total Index of Tehran Stock Exchange. Financ. Stud. **20**(1) (2016)