Ferdowsi University of Mashhad

Iranian Journal of
Accounting, Auditing & Finance

Received: 2020-12-25
Accepted after revision: 2021-03-01
Published online: 2021-05-30

RESEARCH ARTICLE

DOI: 10.22067/ijaaf.2021.67957.1017

Prediction of Interest Rate Using Artificial Neural Network and Novel Meta-Heuristic Algorithms

Milad Shahvaroughi Farahani

Department of Finance, Khatam University, Tehran, Iran

Abstract

One of the most parameters and variables in every economics is the interest rate. Government officials and lawmakers change interest rates for various purposes: controlling liquidity, inflation, and prices, Economic growth and development, lending, etc. So, it is important to set the interest rate correctly. If you can predict the interest rate correctly, you can earn and gain profit by investing in various sectors. Moreover, the interest rate can impact other sectors through parallel markets such as the stock market, automobile, housing, etc. Interest rates are related to parallel markets. Thus, if you can forecast the interest rate, you can predict the parallel markets too. The main goal of this article, as it is clear from the title, is the prediction of interest rate using ANN and improving the network using some novel heuristic algorithms such as Moth Flame Optimization algorithm (MFO), Chimp Optimization Algorithm (CHOA), Time-varying Correlation Particle Swarm Optimization algorithm (TVAC-PSO), etc. we used 17 variables such as oil price, gold coin price, house price, etc. as input variables. We used GA and a new algorithm called Grey Wolf Optimization, Particle Swarm Optimization (GWO-PSO) algorithm as a feature selection and choosing the best variables. We have used eight loss functions such as MSE, RMSE, MAE, etc. too. Finally, we have compared different algorithms due to their estimation errors. The main contribution of this paper is that, first, this is for the first time which these novel metaheuristic algorithms have been used for the prediction of interest rate. Second, it has tried to use different graphs and tables for better understanding and totally a comprehensive research paper. The results show that Whale Optimization Algorithm (WOA) performed better than other methods along with less error.

Keywords: novel meta-heuristic algorithms, interest rate, feature selection, chimp optimization algorithm, moth flame optimization algorithm, loss function

Corresponding author: Milad Shahvaroughi Farahani
Email: *M.shahvaroughi@khatam.ac.ir*

Number of Figures: 19
Number of Tables: 15
Number of References: 39
Pages: 1-30

1. Introduction

Economics money is like blood in the body (Desan, 2008). The final goal of the firms and investors is maximizing their profit Jensen, (2002). So, banks, as one of the main pillars in economic, can distribute money between different sections and parties Peek and Rosengren, (2010). One of the main variables that can determine banks' profitability is the interest rate (Trujillo-Ponce, 2013).

This interest rate depends on different parameters and variables such as oil price, stock market, and parallel markets Friedman, (1977). If we can characterize the relationship between these parameters, we can make almost a relative confidence for future investment. Setting and adjusting a correct interest rate, the inflation rate, and liquidity can stabilize (Goodfriend, 1993). However, interest rates are determined by supply and demand (Friedman, 1966). Faced with intense competition and rising demand for loans by Borrowers, most banks are exploring ways to use their data assets to gain a competitive advantage (Subramanyam, 2016). Furthermore, with the increasing economic globalization and improvements in information technology, large amounts of financial data are being generated and stored. In the light of changing business environments, managers see the need for more flexible predicting models(Vercellis, 2009).

An artificial neural network is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase to make time-series predictions Abraham, (2005). Neural networks are non-linear statistical data modeling tools or information processing paradigm that have a remarkable ability to derive meaning from complicated or imprecise data (Qasim et al., 2013).

As it is clear from the title, the main goal of this research paper is the prediction of interest rate using ANN and improving the network using some novel heuristic algorithms such as Moth Flame Optimization algorithm (MFO), Chimp Optimization Algorithm (CHOA), Time-varying Correlation Particle Swarm Optimization algorithm (TVAC-PSO), etc. we used 17 variables such as oil price, gold coin price, house price, etc. as input variables. We used GA and a new algorithm called Grey Wolf Optimization, Particle Swarm Optimization (GWO-PSO) algorithm as a feature selection and choosing the best variables. We have used eight loss functions such as MSE, RMSE, MAE, etc. too. Finally, we have compared different algorithms due to their estimation errors. The main contribution of this paper is that, first, this is for the first time which these novel metaheuristic algorithms have been used for the prediction of interest rate with brilliant results. Second, it has tried to use different graphs and tables for better understanding and totally a comprehensive research paper. Because most articles deal with stock price forecasts, bitcoin prices, etc., while forecasting interest rates is critical due to an important economic variable. The results show that Whale Optimization Algorithm (WOA) performed better than other methods along with less error.

The structures of the article are as the following:

The first part is about the introduction. In the second part, we have talked about the literature review and background. The third part is about methodology. The next part is about findings and results, and the last part is about conclusions. You can see more results in the appendix too.

2. Literature Review and Background

In the last decades, several methods based on the theory of market equilibrium and no-arbitrage assumption have been proposed to predict interest rate Yasir et al. (2020). There are some models with stochastic processes too. These models are based on economic theories. They have not been dynamic in predicting interest rates because interest rates are dynamic variables and are always affected by other economic variables. In literature, many research papers have been used regression models such as VAR[1], ARCH[2], GARCH[3] , etc., to predict interest rate (Yasir et al. 2020). These models have some assumptions and mechanisms, for example, linearity and normality Zhang and Wu (2011). But as we said earlier, financial time series are not normal and linear, while non-linear skewness means negative skewness Taylor, (2008). So, we should apply some models which be able to identify these features. ANN can provide robust results and outperforms traditional models because it can identify linearity and non-linearity (Jain ans Kumar, 2007). When you use the ARIMA[4] model, your sample size should be more than 50, but in ANN, your sample size can be less than 50 with better results Yasir et al. (, 2020).

From the empirical viewpoint, we can divide literature into four different streams; 1) Change in the yield curve in order to predict the price level and GDP. 2) Data-driven models. Mathematical models which focus on the future behavior of interest rate. 3) Dynamic models such as equilibrium models and arbitrage-free models and 4) Data-driven models which rely on knowledge discovery techniques to predict the interest rate Yasir et al. (2020). These models are compatible with some qualifications such as non-linearity, structural breaks, and seasonality issues laying in variables. In recent years, studies are moving towards artificial intelligence for high abilities such as machine learning (ML), deep learning (DL), ANN, expert systems, fuzzy logic, etc. (Abiodun et al. (2018)

In this part, we have tried to mention the capabilities of using neural networks and metaheuristic algorithms in interest rate forecasting and prediction of the stock market and time series forecasting.

Time series such as the stock market is characterized by non-linearity, discontinuity, and volatile multifaceted elements. It is related to many factors such as political events, general economic conditions, and broker expectations (Hadavandi et al., 2010). Also, the quick process of this data by using high-tech technology and communication systems causes the financial and economic time series to fluctuate very fast. Therefore, many banks, financial institutions, big investors, and brokers have to trade the stock within the shortest possible time. Obtaining maximum profit is the ultimate goal of the investors. As a result, many researchers are looking for market forecasting capabilities in various ways Prasanna and Ezhilmaran, (2013).

ANN dedicated the best and most validated method in predicting time series (Idris et al., 2015). There are many training methods for training the ANN, and some are better than others in finding the linear and non-linear relationship. ANN uses two thresholds for exploring linear and non-linear qualifications. First, the number of the input layer and hidden layer is significant in predictability. If we use too many layers, the ANN couldn't find the fittest choice, and the structure will be complicated. In addition, too few layers mean that the ANN cannot find the global solution and non-linear relationships (Sheela and Deepa, 2013). The researchers have tried to discover some high-speed methods with high accuracy and lower the error. For this reason, metaheuristic algorithms are used. These methods are used to optimize the network and find the best number of input and hidden layers. The ANN models in forecasting stock price, stock return, exchange rate, inflation, and imports work better than traditional statistical models.

Gocken et al. (2016) used technical indicators and hybrid ANN Based on GA and HS to predict the price index in the Turkish stock market. The results showed that the error of hybrid meta-heuristic algorithms is less than ANN. Furthermore, they compared the hybrid ANN-HS and ANN-GA model and found that ANN-HS error is less than ANN-GA.

Hassanin et al. (2016) used the GWO to provide good initial solutions to the ANN. The results

1. Vector Auto-Regressive (VAR)
2. Auto-Regressive Conditionally Heteroskedastic (ARCH)
3. Generalized Auto-Regressive Conditionally Heteroskedastic(GARCH)
4. Auto Regressive Integrated Moving Average (ARIMA)

DOI: 10.22067/ijaaf.2021.67957.1017

showed that GWO based ANN outperforms both GA-based ANN and PSO-based ANN. Faris et al. (2016) presented that MVO shows very competitive optimization results 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. Finally, Rather et al. (2017) observed that the field of hybrid forecasting had received lots of attention from researchers to form a robust model. At this point, ANN provided successful results in stock market forecasting for different stocks.

Mortezapour and Afzali (2013) assessed Customer Credit through Combined Clustering of Artificial Neural Networks, Genetic Algorithm, and Bayesian, Probabilities. In their work, customer credit was assessed using the combination of Genetics Algorithm, Bayesian, Probabilities, and Neural network. Furthermore, it was compared with the methods such as Clustering-Launched Classification (CLC), Support Vector Machine (SVM), and GA+SVM, where the genetics algorithm has been used to improve them.

Dagmar and Jiri (2011) used Neural Networks China-USA business review in their work financial forecasting. From the result, it was seen that an artificial neural network is a tool for financial forecasting.

3. Rsearch Methodology

3.1. Hybrid Metaheuristic Artificial Neural Networks for interest rate Prediction

3.1.1. Input variables

This section describes the input variables selection methodology. For each case, 17 economic variables, such as oil price, gold coin price, etc., are considered input variables. Many economic variables and selecting useful variables are the key issue (Huang et al., 2012). For this reason, we use GA and a novel metaheuristic algorithm called GWO-PSO[1] for selecting the fittest input variables. GA and GWO-PSO are used for determining the variables that have the most significant effect on the forecasting performance. Using GA and GWO-PSO, we can evaluate the usefulness of variables or eliminate irrelevant ones to simplify the proposed model. In table 1, there are all considered economic variables.

3.2. Artificial Neural Network (ANN)

First, we use normal ANN without using any algorithms. Then, to continue, we get right into hybrid ANN for selecting input variables and determining the number of input and hidden layers. In this study, Gocken et al. (2016) used a multi-layer perceptron (MLP) with three layers (two layers for input and output variables and one layer for hidden layer). An input layer includes 17 input variables, i.e., there are 17 neurons in the input variable. Because the output layer has one variable, it has one neuron.

In this study, the hidden layer neurons of the normal neural network model are obtained through trial and error. So, we examine 1-32 neurons in hidden layers and choose the fittest number of neurons with the most accuracy as the ANN model. For training ANN, we use error-back propagation. The minimization algorithm in learning the model is the Marquardt-Levenberg algorithm used to find the minimum error point (Monfared et al., 2012). The number of training epochs is 1000, and for the first time training rate is 0.01, and we decrease this rate to 0.001 in order to obtain more accurate results. ANN has two threshold functions. One of them is for recognizing the linear qualification, and the other is for recognizing the non-linear qualification of the model. The output function of the hidden layer is the sigmoid function, and the threshold function of the output layer is the pure line function.

Figure 1 represents the proposed neural network architecture (fig.1. is adapted from Gocken et al., 2016).

1. Grey Wolf-Particle Swarm Optimization algorithm (GW-PSO)

Table 1. Economic variables as input variables

No	Variables	The variable belongs to the desired subset
1	Liquidity in terms of its components (Billion Rial)	Monetary and credit variables (liquidity)
2	Banks and non-bank credit institutions (Billion Rial)	Monetary and credit variables (Note grant facilities of banks to governmental and non-governmental sectors)
3	Commercial Banks (Billion Rial)	Monetary and credit variables (Note grant facilities of banks to governmental and non-governmental sectors)
4	Specialized banks (Billion Rial)	Monetary and credit variables (Note grant facilities of banks to governmental and non-governmental sectors)
5	Banks and non-bank credit institutions (Billion Rial)	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)
6	Commercial Banks (Billion Rial)	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)
7	Specialized banks (Billion Rial)	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)
8	Non-governmental banks and non-bank credit institutions	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)
9	Crude oil production (one thousand barrels per day)	Energy Section (Oil)
10	Crude oil exports (thousand barrels per day)	Energy Section (Oil)
11	All urban areas (without units)	Construction and housing sector (land price index)
12	Official rate (Rials)	Price of financial assets (exchange rate and coin price), exchange rate, US dollars
13	new design (thousand Rials)	Price of financial assets (exchange rate and coin price)
14	Total index (no units)	Price indices (price index of consumer goods and services) (100 = 1383)
15	Total index (no units)	Stock Exchange (Indicators)
16	Inflation rate	Price indices
17	Short term	Facility interest rate
18	Long term	Facility interest rate

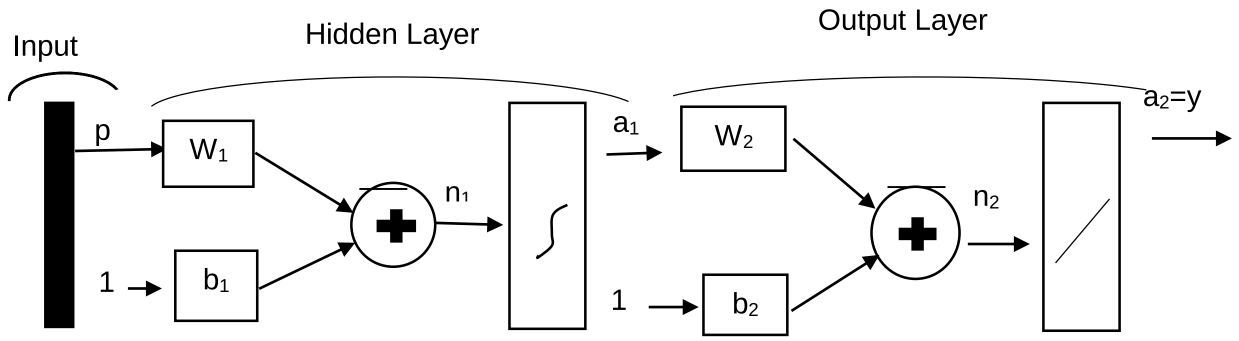

Fig 1. The architecture of the proposed Neural Network

In figure 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 the 0^{th} (i.e., input) and 1st (i.e., hidden) layers. 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. Thus, W_2 is the synaptic weight matrix between the 1^{st} (i.e., hidden) layer and the 2^{nd} (i.e., output) layer, and b_2 is the column-vector containing the

DOI: 10.22067/ijaaf.2021.67957.1017

input neurons' bias inputs. Each row of the W_2 matrix contains the synaptic weights for the corresponding output neuron [Ahmad, Jafri, Ahmad, and Khan, (2005)]. At first, the neuron receives information from the environment. Then this information multiplied by the corresponding weights is added together and used as a parameter within an activation (transfer) function. Haider and Hanif, (2009).

We forecast interest rates by using different hybrid ANN models and comparing their prediction errors. So, we divide interest rate data from 2004-2014 (because of available data) into two parts: training and testing. Then, it is analyzed with artificial intelligence algorithms and forecasting the next season's interest rate. Like Gocken et al. (2016), 70 percent of observation is used for training, and 30 percent of observation is used for testing and validation. We compare models with 8 criteria for prediction error. For training ANN, different algorithms are used. These algorithms exist in the MATLAB toolbox.

We normalized numbers between 0,1. In equation 1, numerator i is the number of data.

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

Figure 2 represents the research methodology (Fig.2. is adapted from Ghasemiyeh et al., 2017).

3.3. GA-ANN forecasting model

In this model, GA was used for input variable selection and used ANN as the fitness function. In this study, the considered coding is binary coding. The usable chromosome contains 18 bits which 13 bits present the existence or nonexistence of input (economic indicator) variables. If the bit is "0," it means don't exist variable, and if be "1" means that exist variable and forming neuron in the input layer. 5 other bits are equal to 1-32 (2^5=32), showing the number of neurons in hidden layers. The population size of GA is 20 (Davallou and Azizi, 2017). The first population selects randomly. The fitness function is ANN, and its input variable is economic variables, and the number of hidden layers and its output is the amount of MSE. The smallest MSE in this series is the fittest choice for the next forecasting period. For increasing the training speed algorithms, the epochs are 100. We use 70 percent of the data for training ANN. Other 30 percent used for testing and validation. At first, the training (learning) rate is 0.01, which will decrease with repeating training to obtain more exact results. If we want to obtain a more accurate result, we can increase the epochs to 1000. The considered parameters in the genetic algorithm are as table 2:

Like Gocken et al. (2016), we used a roulette wheel for selecting parents, and the crossover percentage is 80. We use one-point crossover in crossover and doing fitness functions for all of them. We use binary mutation, and the mutation percentage is 20. Among 20 parents and 20 children, we select the 20 best individuals as new generations. The new generation continues with repeating the above method until reaching the termination condition. One of the termination conditions is repeating the best individual to 100 generations. If this condition doesn't hold, we check the maximum generation condition. The maximum number of producing generation is equal to 2000.

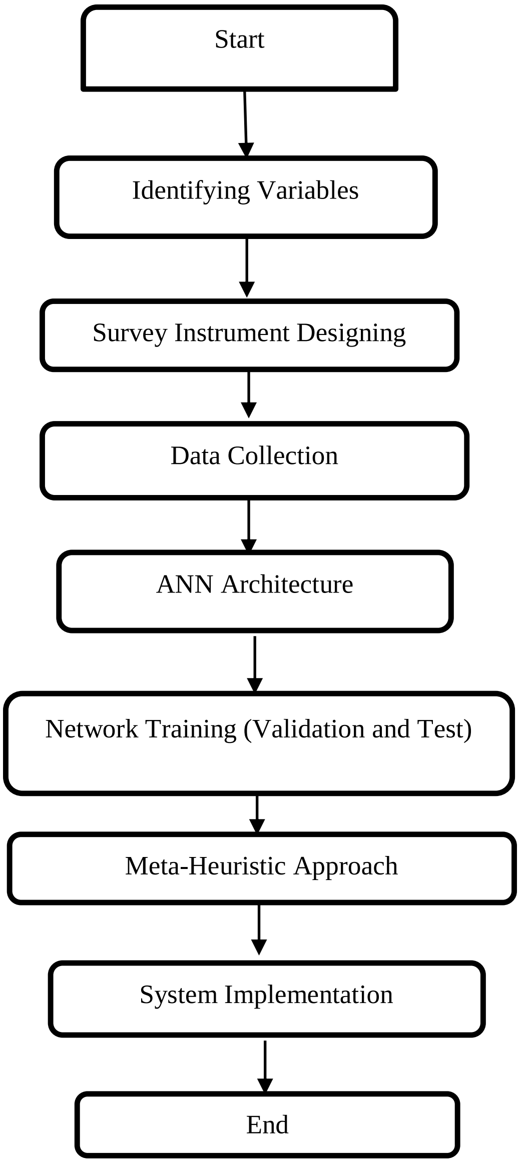

Fig 2. research methodology

Table 2. GA parameters

Output Error	**Output Activation Function**	**Input Activation Function**	**Mutation Rate**	**Crossover Rate**	**Number of Generation**	**Population Size**
SSE	Logistic	Logistic	0.1	0.9	50	50

Figure 3 represents the related flow chart of GA-ANN (Fig.3. is adapted from Gocken et al., 2016).

3.4. GWO-PSO algorithm

This part is adapted from Al-Tashi et al. (2019). It is used PSO and GWO algorithms for optimizing exploitation and exploration, respectively. The inertia weight in GWO modeled as follows:

$$\overrightarrow{D_a} = \left|\overrightarrow{C_1} \cdot \overrightarrow{X_a} - w * \vec{X}\right|. \quad (2)$$

$$\overrightarrow{D_\beta} = \left|\overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - w * \vec{X}\right|. \tag{3}$$

$$\overrightarrow{D_\delta} = \left|\overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - w * \vec{X}\right| \tag{4}$$

In the hybrid model, velocity and position can be updated as follows:

$$V_i^{k+1} = w * \left(V_i^k + c_1 r_1 (X_1 - X_i^k) + c_2 r_2 (X_2 - X_i^k) + c_3 r_3 (X_3 - X_i^k)\right) \tag{5}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{6}$$

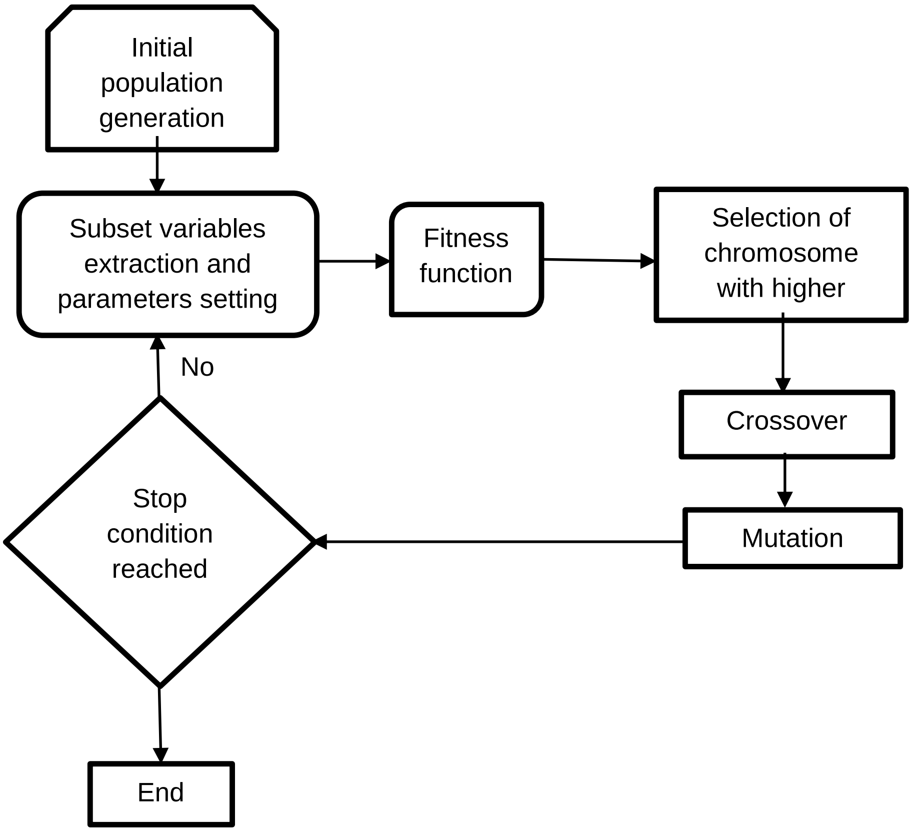

Fig 3. Considered GA flow chart for training ANN

We should turn the problem into a binary position. Agents can continuously move around the search space in the original PSO-GWO since they have position vectors with a real continuous domain. GWO algorithm contains three position vectors: x_1, x_2, x_3, which promotes every wolf to the first three best solutions. We can update the position and modifying it into the following equations:

$$X_d^{i+1} = \begin{cases} 1 & if\ sigmoid(\frac{x_1 . x_2 . x_3}{3}) \geq rand \\ 0 & otherwise \end{cases} \tag{7}$$

where X_d^{i+1} is the binary updated position at iteration t in dimension d, a rand is a random number with uniform distribution ϵ [1,0], and sigmoid(a) is denoted as follows:

$$sigmoid\ (a) = \frac{1}{1+e^{-10(x-0.5)}} \tag{8}$$

x_1, x_2, x_3 are updated and calculated using the following equation:

$$X_1^d = \begin{cases} 1 & if\ (X_a^d + bstep_a^d) \geq 1 \\ 0 & otherwise \end{cases} \tag{9}$$

$$X_2^d = \begin{cases} 1 & if\ (X_\beta^d + bstep_\beta^d) \geq 1 \\ 0 & otherwise \end{cases} \tag{10}$$

$$X_3^d = \begin{cases} 1 & if\ (X_\delta^d + bstep_\delta^d) \geq 1 \\ 0 & otherwise \end{cases} \tag{11}$$

Where $X_{a.\beta.\delta}^{d}$ the position's vector of the alpha, beta, delta wolves in d dimension, and $bstep_{a.\beta.\delta}^{d}$ is a binary step in d dimension, which can be formulated as follow:

$$bstep_{a.\beta.\delta}^{d} = \begin{cases} 1 & if\ cstep_{a.\beta.\delta}^{d}) \geq rand \\ 0 & otherwise \end{cases} \tag{12}$$

where $rand$ a random value derived from uniform distribution ϵ [1,0], d indicates dimension and $cstep\ _{a.\beta.\delta}^{d}$ is d's continuous value. This component is calculated using the following equation:

$$bstep_{a.\beta.\delta}^{d} = \frac{1}{1+e^{-10(A_1^d D_{\alpha.\beta.\delta}^d - 0.5)}} \tag{13}$$

We used 70% for training and 30% for validation and testing. In addition, we used some evaluation measures means statistical measures in each run.

The flowchart of the hybrid GWO-PSO binary algorithm is the following: (Fig.4. is adapted from Shaheen et al., 2020).

3.5. MPSO algorithm

This part is adapted from He and Guo (2013). The simple PSO doesn't have an inertia weight parameter. In the PSO algorithm, inertia factor ω according to literature, decreases linearly.

$$\omega = \omega_{\ max} - \frac{\omega_{\ max} - \omega_{\ min}}{iter_{max}} \times iter \tag{14}$$

Here $iter_{max}$ is the biggest evolution of algebra, $iter$ is the algebra for this evolution. Improved particle swarm optimization algorithm BP operations driven by the amount of correction of the weights of the way, that the amount of correction of the weights between the neural network node m and node n from below equation:

$$\Delta\omega_{nm}(t+1) = \alpha\Delta\omega_{nm}(t) + \eta S_n(t) y_m(t) \tag{15}$$

Here, the amount of correlation of the conventional BP, α is an algorithm, and $y_m(t)$ is the momentum term inertia coefficient for the output node m. The above gradient particle swarm method for three-layer neural network training, through the experiments coding network weights and threshold values, constitutes a vector. The vector is a particle in the particle swarm algorithm. If three feedforward network structure is taken as the $n_{in} - H - n_{out}$ form, you need to optimize network $n_{in}.H + H.n_{out} + H.n_{out}$ parameters.

At first, in the search space, randomly generated initial population P particles constitute, through the adaptation function defined groups each particle fitness value, the definition of appropriate learning network search for an optimal combination of parameters using the method with the smallest function value.

The Flowchart of Modified Particle Swarm Optimization Algorithm (MPSO) is as follows: (Fig.5. is adapted from Amin et al. 2020).

3.6. MPSO-TVAC

This part is adapted from Abdullah et al. (2014). There is a parameter for improving exploitation and exploration and preventing local traps. In this strategy, each particle has its own $[rbest_i{}^j = rbest_{i1}{}^j.rbest_{i2}{}^j.\ldots.rbest_{id}{}^j]$ which is randomly selected from the best position (Pbest) of other particles. A similar approach is applied to other particles in the swarm. For example, we can use the following equations for updating velocity:

$$V_{j+1}^{i} = W_j V_j^i + c_1 r_1 \left(X_j^{i.pbest} - X_j^i\right) + c_2 r_2 \left(X_j^{Gbest} - X_j^i\right) + c_3 r_3 \left(rbest_{id}{}^j - X_j^i\right) \tag{16}$$

where, c_3 is the acceleration coefficient that pulls each particle towards the rbest. Both coefficients should be changed to improving exploitation and exploration. A large value of the cognitive component and small social component in the initial iteration pushes the particles to move to the

DOI: 10.22067/ijaaf.2021.67957.1017

entire solution space. When we increase iteration, we will decrease the cognitive value, and the social components value will increase, leading the particles to the global solution. The acceleration coefficients are varied according to the following formulas:

$$c_1 = c_{1i} + c_{1f} - c_{1i}) \times \frac{j}{j_{max}} \tag{17}$$

$$c_2 = c_{2i} + c_{2f} - c_{2i}) \times \frac{j}{j_{max}} \tag{18}$$

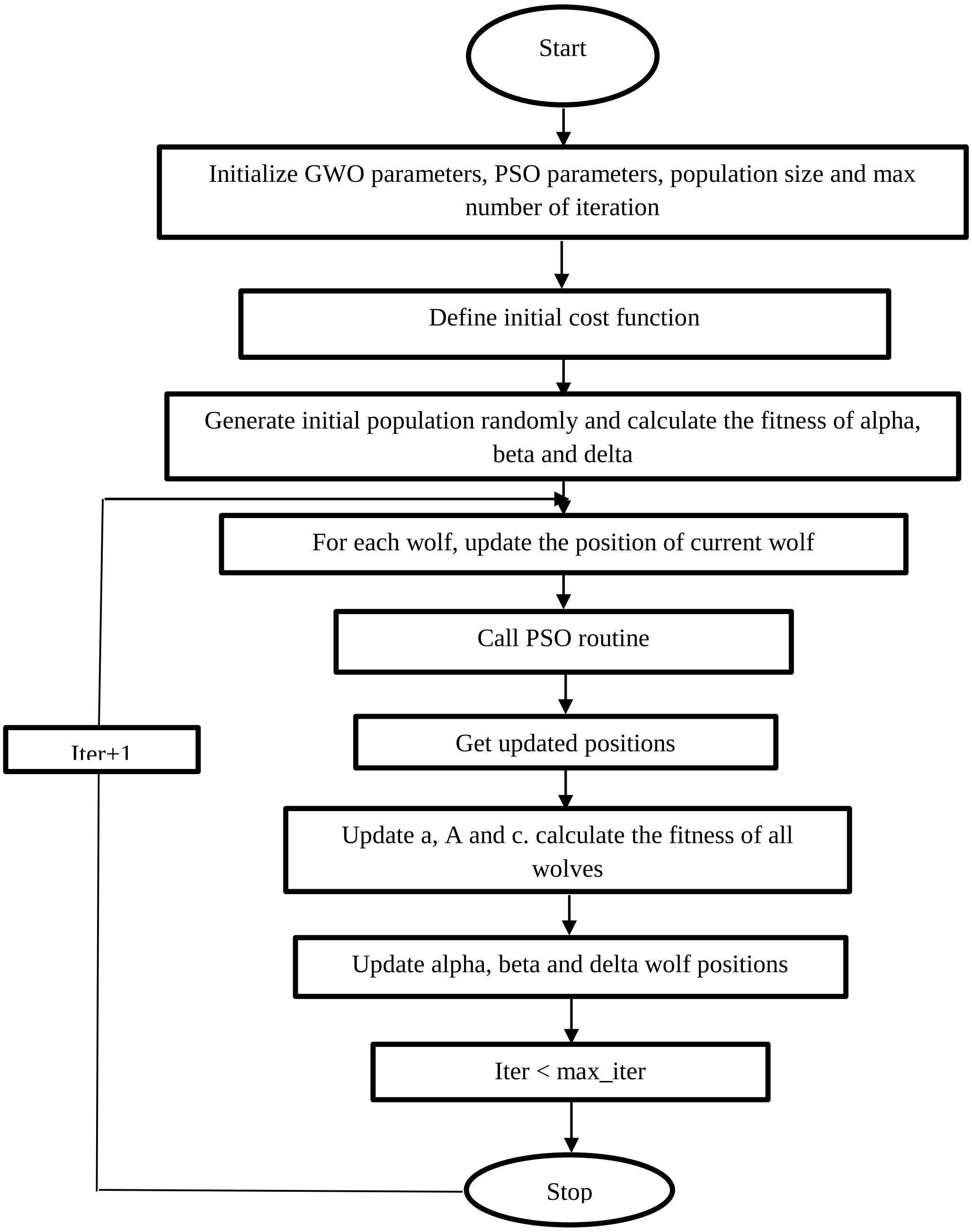

Fig 4. GWO-PSO algorithm flowchart

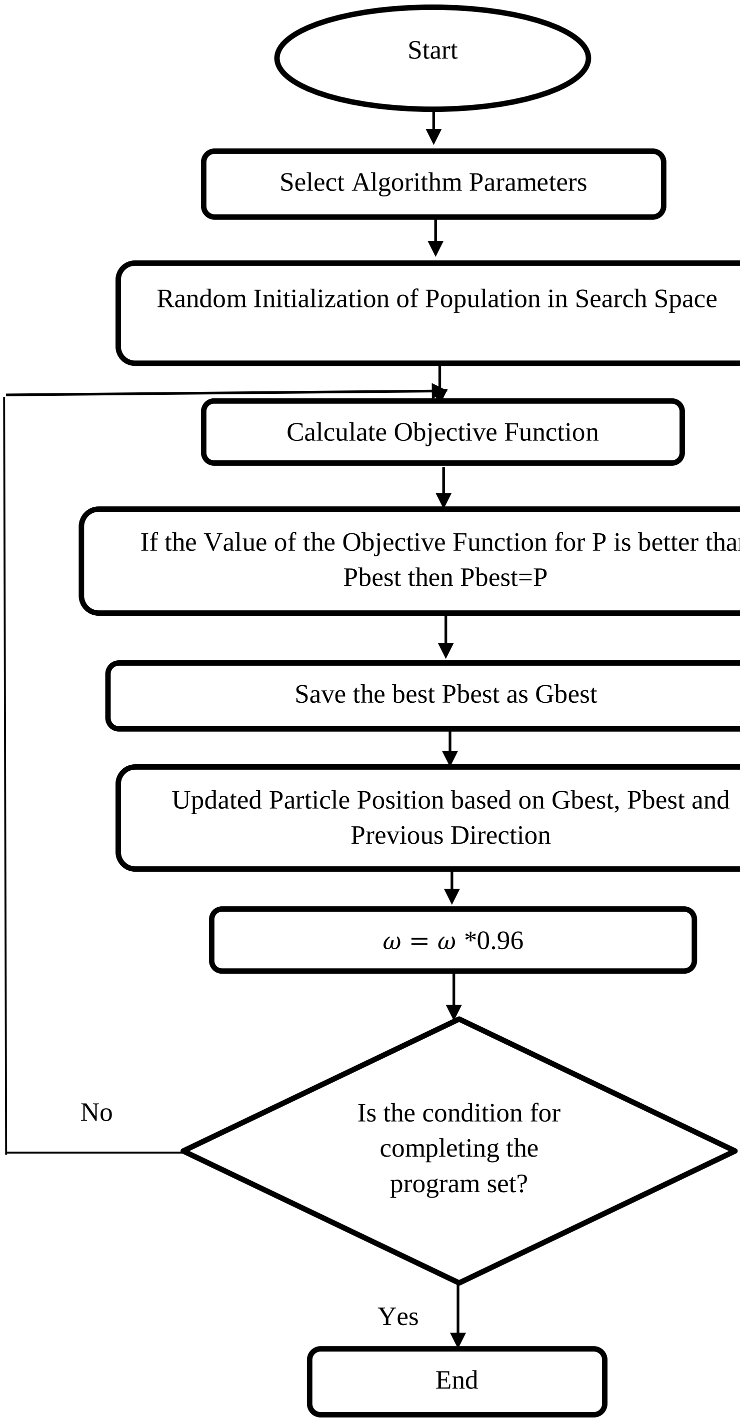

Fig 5. MPSO flowchart

where c_{1i} and c_{1f} are the initial and final values of cognitive coefficient, respectively and c_{2i} and c_{2f} are the initial and final values of the social coefficient, respectively. By adding a new parameter (rbest) in the velocity equation, the convergence rate will be improved because of provided extra information by the rbest value in the current iteration. We can calculate the time-varying acceleration coefficient for rbest component (c_3) by using the following Eq.

$$c_3 = c_1 \times (1 - exp(-c_2 \times j)) \quad (19)$$

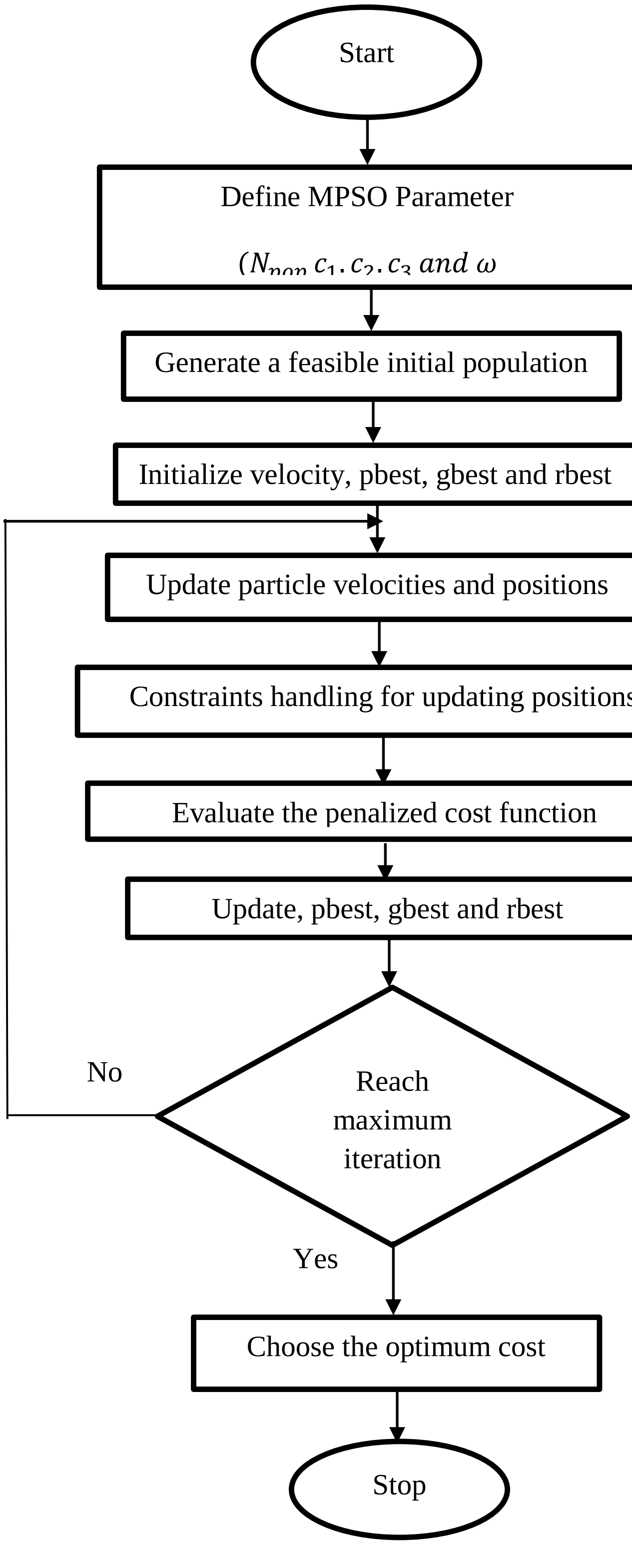

Fig 6. MPSO-TVAC flowchart

It assumed that, the $c_1$value varies from 1 to 0.2 and c_2 varies from 0.2 to 1 during 100 iterations.

At the initial iteration, the c_3 value is increased immediately, which expanding the exploration based on the best neighbor particle (rbest). This can avoid quick convergence and local traps. As iteration proceeds, the c_2 value moves toward global gbest. Therefore, it can improve the exploitation and exploration and providing good solutions near-global optimum. The MPSO-TVAC flowchart is as the following:

3.7. MFO algorithm

This part is adapted from Mirjalili (2015). In the MFO algorithm, moths and the position of moths in the space are candidate solutions and the problem's variables, respectively. So, moths fly in different dimensional spaces, and they can shift their position vector too. Here, moths and flames are both solutions. In each iteration, they update themselves. The moths are actual search agents that move around the search space, whereas flames are the best moths that obtain so far. A logarithmic spiral has been chosen for updating the mechanism of moths. However, different kinds of the spiral can be used: Spiral's initial point should start from the moth Spiral's final point should be the position of the flame Fluctuation of the range of spiral should not exceed from the search space considering these points, we can define a logarithmic spiral for the MFO algorithm as follows:

$$S(M_i.F_i) = D_i e^{bt} \cos(2\pi t) + F_j \tag{20}$$

Where D_i indicates the distance of the $i-th$ moth for the $j-th$ flame, b is a constant for defining the shape of the logarithmic spiral, and t is a random number in [-1,1]. D is calculated as follows:

$$D_i = |F_j - M_i| \tag{21}$$

Where M_i indicate the $i-th$ moth, F_j indicates the $j-th$ flame, and D_i indicates the distance of the $i-th$ moth for the $j-th$ flame.

Equation1. Simulating the spiral flying path of moths. As you can see in this equation, the next position of a moth is defined regarding a flame. The t parameter in the spiral equation defines how much the next position of the moth should be close to the flame (t = -1 is the closest position to the flame, while t = 1 shows the farthest). Therefore, a hyper ellipse can be assumed around the flame in all directions, and the next position of the moth would be within this space. Spiral movement is important because it shows the updating of moths to their positions around flames. The spiral equation allows a moth to fly "around" a flame and not necessarily in the space between them. Therefore, the exploration and exploitation of the search space can be improved. The exploitation and exploration for finding new and optimum solutions are very important. So, the matrix F in the above equation always includes n recent best solutions obtained so far. In order to further emphasize exploitation, we assume that t is a random number in [r,1] where r is linearly decreased from -1 to -2 throughout the iteration. Note that we name r as the convergence constant. With this method, moths tend to exploit their corresponding flames more accurately proportional to the number of iterations. In order to prevent a local optima trap, each moth must update its position using only one of the flames. After each iteration and updating, the list of flames is sorted based on fitness value. The first moth updates its position due to its flames, whereas the last moth updates its position concerning the worst flame in the list. We should move moths around different flames. However, it increases search space, and it prevents us from reaching the optimal solution. Thus, as a solution, we have proposed an adaptive mechanism for the number of flames.

$$FlameNumber = round(N - l \times \frac{N-1}{T}) \tag{22}$$

Where l is the current iteration number, N is the maximum number of flames and T indicates the maximum number of iterations. There is N number of flames in the initial steps of iterations.

However, in the final steps of iterations, the moths update their positions only for the best flame. By a gradual decrease in the number of flames, a kind of balance between exploitation and exploration is generated.

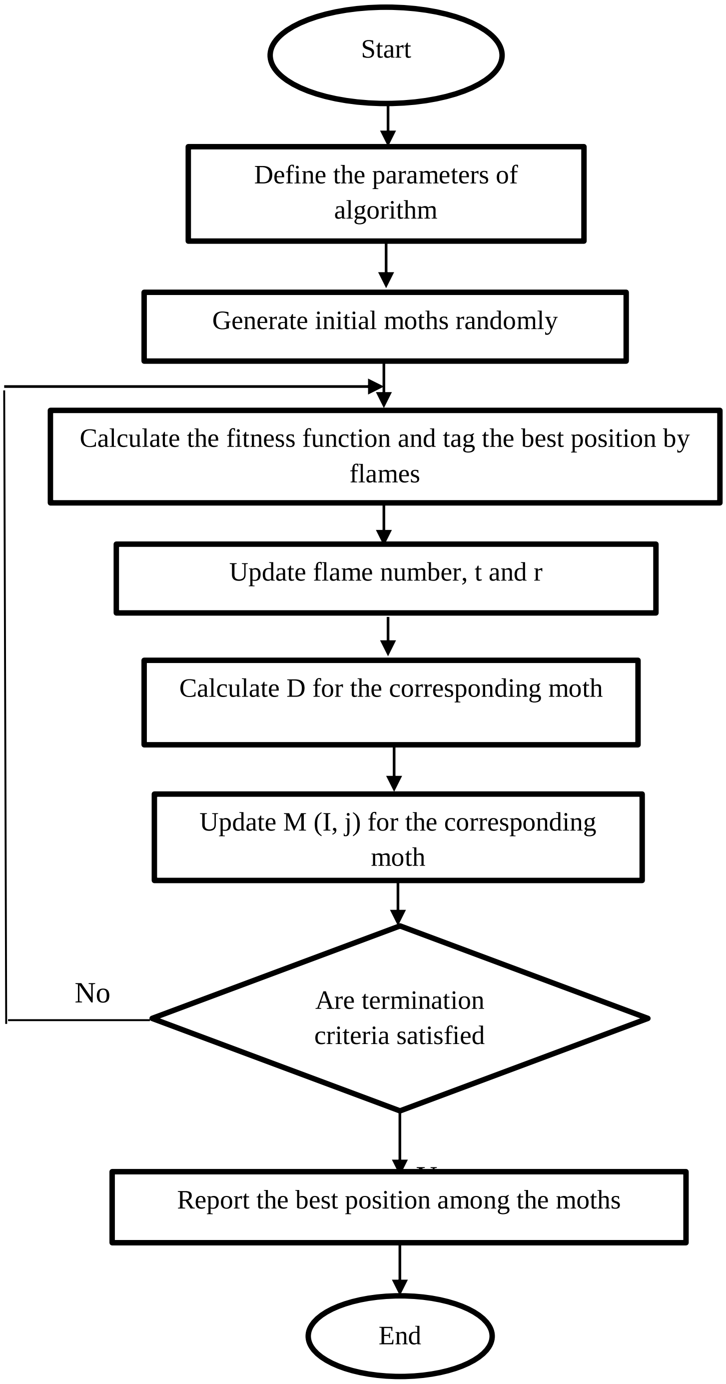

Fig 7. MFO flowchart

3.8. WOA

Humpback whales consider the current best candidate solution and best-obtained solution near the optimal solution. After achieving the best candidate solution, the other agents update their positions towards the best search agent. So, we can show it in the following equations;

$$D = |C.X^{*}(t) - X(t)| \tag{23}$$

$$X(t+1) = X^*(t) - A.D \tag{24}$$

Where (t) is the current iteration, A and C are coefficient vectors, X^* is the position vector of the best solution, and X indicates the position vector of a solution, | | is the absolute value. The vectors A and C are calculated as follows:

$$A = 2a.r.a \tag{25}$$

$$C = 2.r \tag{26}$$

Where components of a are linearly decreased from 2 to 0 throughout iterations for convergence purpose and r is a random vector in [0; 1]. The humpback whales attack the prey with the bubble-net mechanism. This mechanism is mathematically formulated as follow:

Shrinking encircling mechanism:

In this mechanism, the value of A is a random value in the interval [-a, a], and the value of a is decreased from 2 to 0 for iterations, as shown in Eq. 25.

Spiral updating position mechanism

In this mechanism, the distance between the whale location and the prey location is calculated then the helix-shaped movement of the humpback is created as shown in the following equation;

$$X(t+1) = D'.e^{bl}.cos(2\pi l) + X^*(t) \tag{27}$$

Where $D' = | X^*(t) - X(t) |$ is the distance between the prey (best solution) and the i^th^ whale, b is a constant, l is a random number in [-1; 1].

The humpback whales used the mentioned two mechanisms when they swim around the prey. We set the mathematical model of these two mechanisms. We assume that there is a probability of 50% to choose between these two mechanisms to update the position of whales as follow:

$$X(t+1) = \begin{cases} X^* - A.D & if\ p<0{\cdot}5 \\ D'.e^{bl}.cos(2\pi l) + X^*(t) & if\ p\geq 0{\cdot}5 \end{cases} \tag{28}$$

Where p is a random number in [0; 1].

In the exploration phase, the humpback whales (search agents) randomly search for prey (best solution) and adjust their positions based on the other whales. In order to oblige the search agent to move far away from the reference whale or best solution, we use the A with values > 1 or < 1. The mathematical model of the exploration phase is as follows:

$$D = |C.X_{rand} - X| \tag{29}$$

$$X(t+1) = X_{rand} - A.D \tag{30}$$

Where X_{rand} is a random position vector chosen from the current population

3.9. CHOA

This part is adapted from Khishe and Mosavi (2020).

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.

The chimp's hunting model means driving, blocking, chasing, and attacking is modeled in this section.

3.9.1. Driving and chasing the prey

As mentioned before, during the exploration and exploitation phases, the prey is hunted. Therefore, we can model driving and chasing mathematically like the below equations:

$$d = \left| c \cdot x_{prey}(t) - m \cdot x_{chimp}(t) \right| \tag{31}$$

$$x_{chimp}(t+1) = x_{prey}(t) - a \cdot d \tag{32}$$

DOI: 10.22067/ijaaf.2021.67957.1017

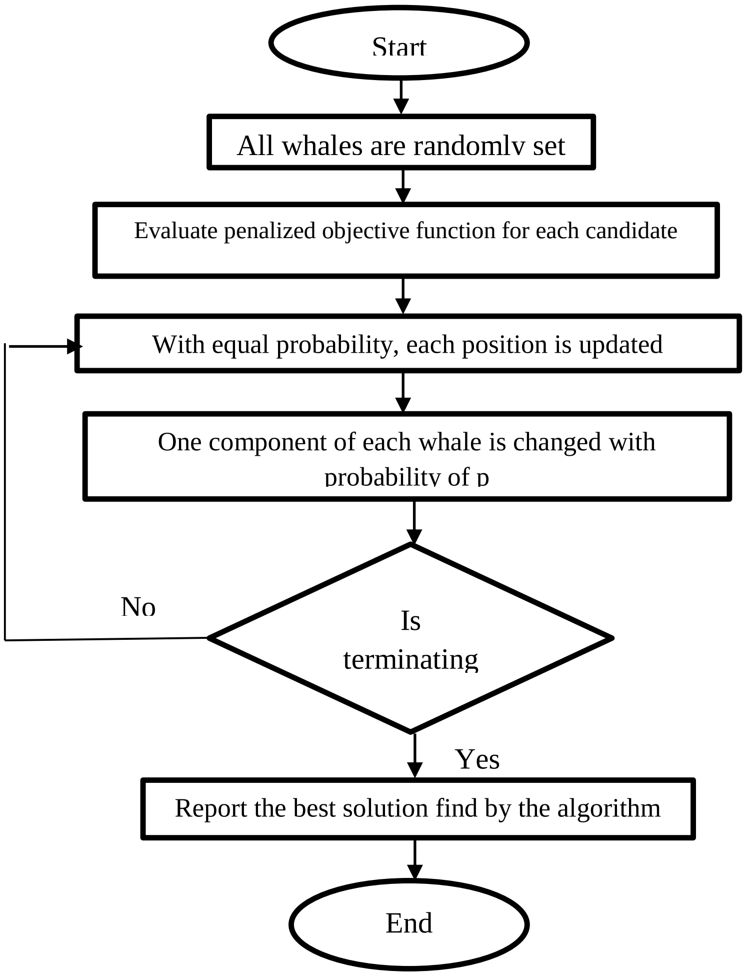

Fig 8. WOA flowchart

Where t indicates the number of the current iteration, a, m, and c is the coefficient vectors, x_{prey} is the vector of prey position and x_{chimp} is the position vector of a chimp. a, m, and c vectors are calculated by the Eq.s (33), (34), and (35), respectively.

$$a = 2 \cdot f \cdot r_1 - f \tag{33}$$

$$c = 2 \cdot r_2 \tag{34}$$

$$m = chaotic\_value \tag{35}$$

In which, f is reduced non-linearly from 2.5 to 0 through the iteration process (in both exploitation and exploration phase). Where r_1 and r_2 are the random vectors in the range of [0,1]. Finally, we can show the sexual motivation of chimps in the hunting process based on the different chaotic map which m is a chaotic vector which is calculated based on the various chaotic map. These four independent groups use their own patterns to search the problem space locally and globally. We have divided strategies into two parts called ChOA1 and ChOA2 are selected to have the best performance in the benchmark optimization problems. The dynamic coefficients of f have been proposed in Table 3. In this table, T represents the maximum number of iterations, and t indicates the current iteration. These dynamic coefficients have been chosen with various curves and slopes so that each independent group has specific searching behavior to improve the performance of CHOA.

Table 3. Two Kinds of CHOA along with four groups

Groups	ChOA1	ChOA2
Group1	1.95-22$t^{1/4}/T^{1/3}$	2.5-(2log(t)/log(T))
Group2	1.95-22$t^{1/3}/T^{1/4}$	$(-2t^3/T^3)$+2.5
Group3	$(-3t^3/T^3)$+1.5	$0.5 + 2\exp[-(4t/T)^2]$
Group4	$(-2t^3/T^3)$+1.5	2.5+2$((t/T)^2 - 2(2t/T)$

A chimp in position (x, y) can change its position for preys $(X^*.Y^*)$ location. Various locations around the most suitable agent can be taken considering its current location and changing and setting the values of a and c vectors. For instance, the location of $((X^* - X.Y^*)$ is obtained by setting a=(1,0), m=(1,1), and c=(1,1). It should be noted that the chimps are allowed to access any position between the points through the random vectors r_1 and r_2. So, any chimp can randomly change its location within the space surrounding the prey using Eq. s (31) and (32).

3.9.2. Attacking method (exploitation phase)

Chimps can find the prey's location (by driving, blocking, and chasing) and encircling it. In order to mathematically simulate the behavior of the chimps, it is assumed that the first attacker (best solution available), driver, barrier, and chaser are better informed about the location of potential prey. So, the best solutions are stored, and the other chimps update their positions based on the best solutions. The Eqs express this relationship. (36-40).

$$d_{Attacker} = |c_1 x_{Attacker} - m_1 x| . d_{Barrier} = |c_2 x_{Barrier} - m_2 x| . \quad (36)$$
$$d_{Chaser} = |c_3 x_{Chaser} - m_3 x| . d_{Driver} = |c_4 x_{Driver} - m_4 x| \quad (37)$$
$$x_1 = x_{Attacker} - a_1(d_{Attacker}) . x_2 = x_{Barrier} - a_2(d_{Barrier}) . \quad (38)$$
$$x_3 = x_{Chaser} - a_3(d_{Chaser}) . x_4 = x_{Driver} - a_4(d_{Driver}) \quad (39)$$
$$x(t+1) = \frac{x_1+x_2+x_3+x_4}{4} \quad (40)$$

The final position is located randomly in a circle defined by an attacker, barrier, chaser, and driver chimp positions. In other words, the best position to prey (solution) is obtained by four chimps, and then the remaining chimps adjust their position based on the best position.

3.9.3. Prey attacking (utilization)

The final stage is attacking the prey and kill it as soon as possible. Thus, like other algorithms, the value of f should be reduced. Note that the variation range of the a is also reduced by f. In other words, a is a random variable in the interval of [-2f,2f], whereas the value of f reduces from 2.5 to 0 in the period of iterations when the random values of a lie in the range of [-1,1], the next position of a chimp can be in any location between its current position and the position of the prey.

The risk of a local minima trap is still existing because the other chimps following the agent. So, we should overcome this problem.

3.9.4. Searching for prey

Due to mathematical modeling attacking prey, chimps should take in a domain for divergence. So, the a vector is used with a random value between [-1,1]. This is an exploration process, and it's a global search. The inequality $|a| > 1$ forces the chimps to scatter in the environment to find better prey.

We should improve the position and distance to prey to prevent the local minima trap with the value of c parameter, which is a random variable in [0,2]. The value of c can make the hunt harder or easier.

3.9.5 Social incentive

Finally, chimps show chaotic behavior due to social motivation. This is accompanied by slow convergence and preventing the local minima trap. The chaotic map is used to improve the performance of CHOA. Six chaotic maps are used. The initial value for the map is 0.7, and there is a probability of 50% to choose between either the normal updating position mechanism or the chaotic model to update—the position of chimps during optimization.

$$x_{chimp}(t+1) = \begin{cases} x_{prey}(t) - a \cdot d & if\ \mu < 0 \cdot 5 \\ Chaotic\_value & if\ \mu \geq 0 \cdot 5 \end{cases} \tag{41}$$

Where μ is a random number in [0,1].

Table 4. Chaotic map with its functions and name

No	Name	Chaotic map	Range
1	Quadratic	$x_{i+1} = x_i^2 - c . c = 1$	(0,1)
2	Gauss / mouse	$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \dfrac{1}{mod(x_{i,1})} & otherwise \end{cases}$	(0,1)
3	Logistic	$x_{i+1} = ax_i(1 - x_i). a = 4$	(0,1)
4	Singer	$x_{i+1} = \mu(7 \cdot 86x_i - 23 \cdot 31x_i^2 + 28 \cdot 75x_i^3 - 13 \cdot 3028x_i^4). \mu = 1 \cdot 07$	(0,1)
5	Bernoulli	$x_{i+1} = 2x_i(mod1)$	(0,1)
6	Tenet	$x_{i+1} = \begin{cases} \dfrac{x_i}{0 \cdot 7} & x_i < 0 \cdot 7 \\ \dfrac{10}{3}(1 - x_i) & 0 \cdot 7 \leq x_i \end{cases}$	(0,1)

You can see the CHOA flowchart in the following:

4. Findings and Results

4.1. Organize and describe data

in this article, we use 18 technical indicators to predict interest rate which 16 variables are input variables, and 1 variable is output or target variable that is the long-term interest rate for the next day. The desired time interval is from the beginning of 2004 to the end of 2014, about 10years which data is seasonal. In order to access and getting data, we used the Laboratory risk of Khatam University. We used direct write-off for missing value, or N/A data which is the common and default approaches in many software and our research. The method to choose the most appropriate indicators is the genetic algorithm and GWO-PSO algorithm, an evolutionary algorithm.

The numbers of inputs, targets, and unused variables here are 17, 1, and 0, respectively.

The following pie chart details the uses of all the instances in the data set. The total number of instances is 44. The number of training instances is 28 (63.6%), the number of selection instances is 8 (18.2%), the number of testing instances is 8 (18.2%), and the number of unused instances is 0 (0%).

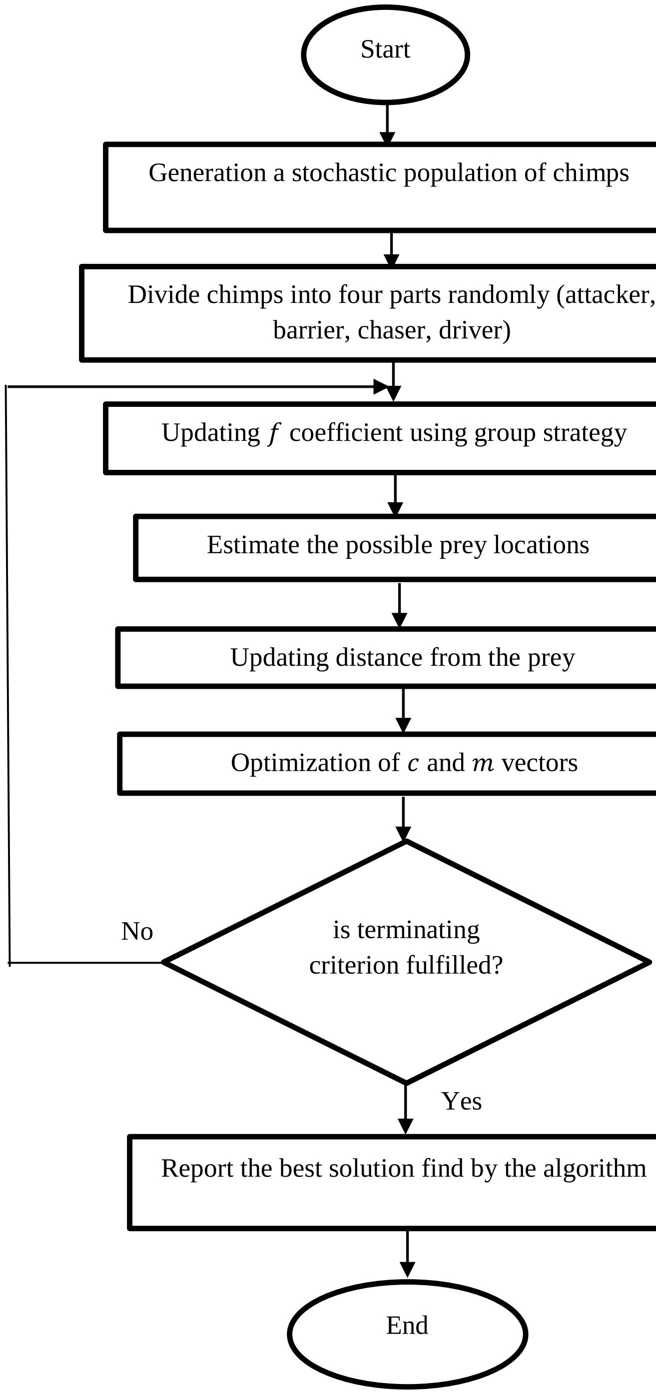

Fig 9. CHOA flowchart

The following table shows the statistics of the parameters of the neural network. The total number of parameters is 1.

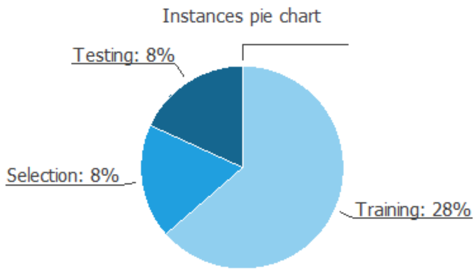

Fig 10. Instances pie chart

Table 5. Neural network parameters

	Max	Min	Mean	Std.dev
Statistics	-0.93	0.966	0.0483	0.552

A graphical representation of the network architecture is depicted next. It contains a scaling layer, a neural network, and an un-scaling layer. The yellow circles represent scaling neurons, the blue circle's perceptron neurons, and the red circle's un-scaling neurons. The number of inputs is 16, and the number of outputs is 1. The complexity, represented by the number of hidden neurons, is 3.

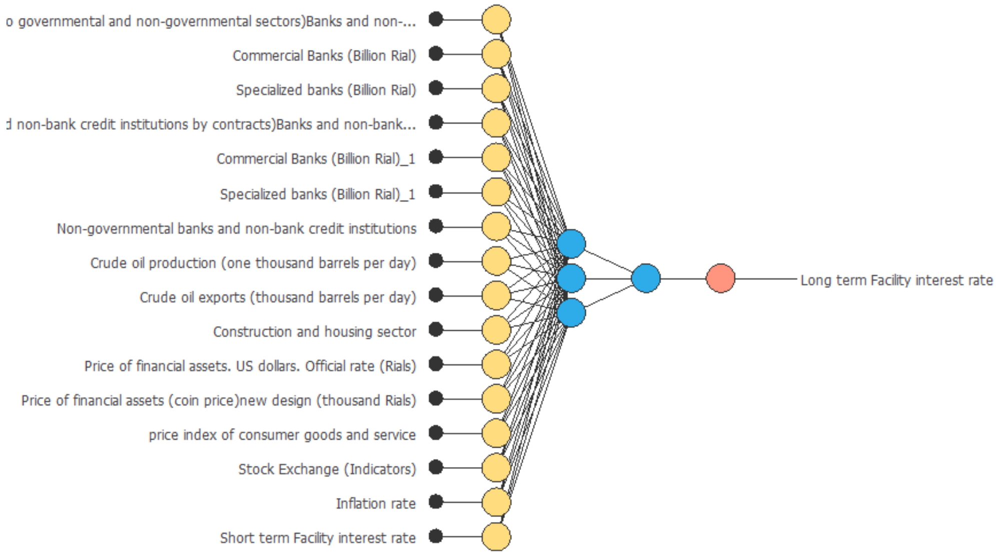

Fig 11. Neural network structure

The quasi-Newton method is used here as an optimization algorithm. It is based on Newton's method but does not require the calculation of 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.

The next table shows the training results by the quasi-Newton method. They include some final states from the neural network, the less functional, and the optimization algorithm.

Table 6. Quasi-newton method results

	Value
Final parameters norm	2.25
Final training error	0.000168
Final selection error	0.134
Final gradient norm	0.00084
Epochs number	288
Elapsed time	00:01
Stopping criterion	Gradient norm goal

The next part is about the selection of the most important variables means feature selection by using GA. Then, model selection is applied to find a neural network with a topology that optimizes the error on new data. Finally, input selection algorithms are responsible for finding the optimal subset of input variables.

Table 7. Input selection algorithm

	Value
Trial number	1
Tolerance	0.01
Population size	20
Initialization method	Random
Fitness assignment method	Rank-based
Crossover method	Uniform
Elitism size	2
Crossover first point	0
Crossover second point	0
Selective pressure	1.5
Mutation rate	0.05
Selection loss goal	0
Maximum generations number	1000
Maximum time	3600
Plot training error history	True
Plot selection error history	True
Plot generation mean history	True

A graphical representation of the resulted deep architecture is depicted next. It contains a scaling layer, a neural network, and an un-scaling layer. The yellow circles represent scaling neurons, the blue circle's perceptron neurons, and the red circle's un-scaling neurons. The number of inputs is 16, and the number of outputs is 1. The complexity, represented by the number of hidden neurons, is 7.

The next chart shows the error history for the different subsets during the genetic algorithm inputs selection process. The blue line represents the training error, its initial value is 0.000396812, and the final value after 1000 generations is 0.000418828. The orange line symbolizes the selection error, its initial value is 0.0454684, and the final value after 1000 generations is 0.0299098.

The next chart shows the history of the mean of the selection error in each generation during the genetic algorithm inputs selection process. The initial value is 0.516062, and the final value after 1000 generations is 0.0268.

The next table shows the input selection results by the genetic algorithm. They include some final states from the neural network, the error function, and the selection of the input algorithm

DOI: 10.22067/ijaaf.2021.67957.1017

Table 8. Genetic algorithm results

	Value
The optimal number of inputs	7
Optimum training error	0.000418828
Optimum selection error	0.0299098
Generations number	1000
Elapsed time	00:39

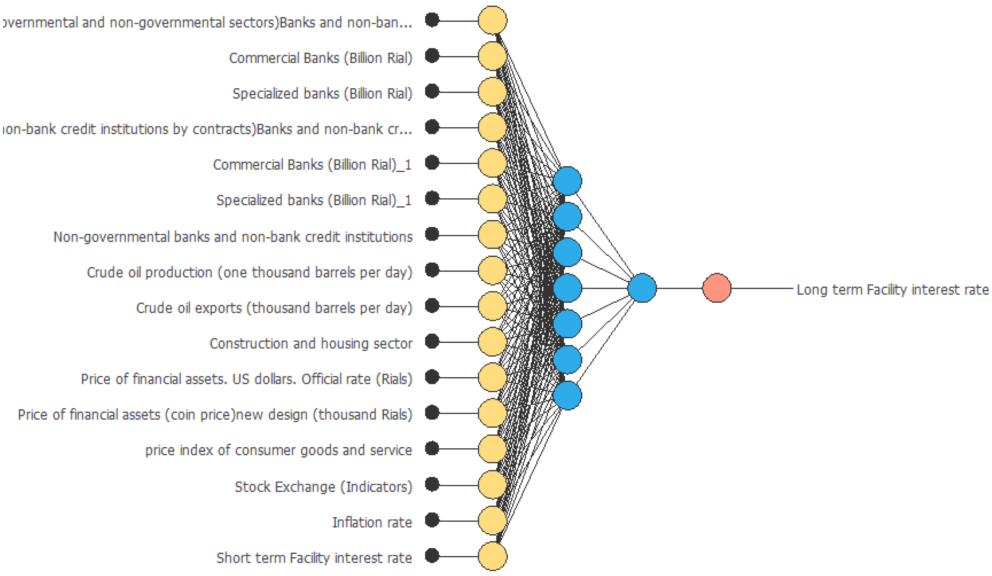

Fig 12. Final architecture

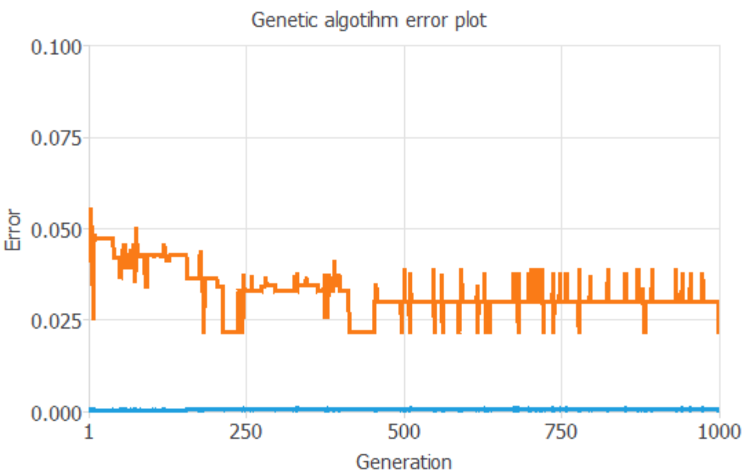

Fig 13. The genetic algorithm error plot

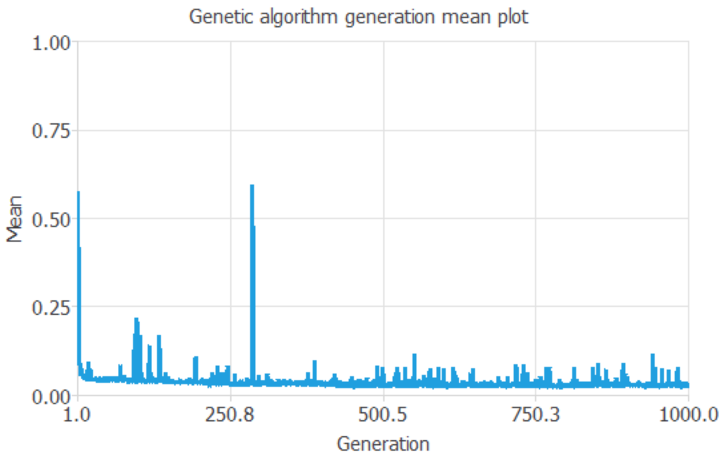

Fig 14. Genetic algorithm generation means plot

A graphical representation of the resulted deep architecture is depicted next. It contains a scaling layer, a neural network, and an un-scaling layer. The yellow circles represent scaling neurons, the blue circle's perceptron neurons, and the red circle's un-scaling neurons. The number of inputs is 7, and the number of outputs is 1. Thus, the complexity, represented by the number of hidden neurons, is 7.

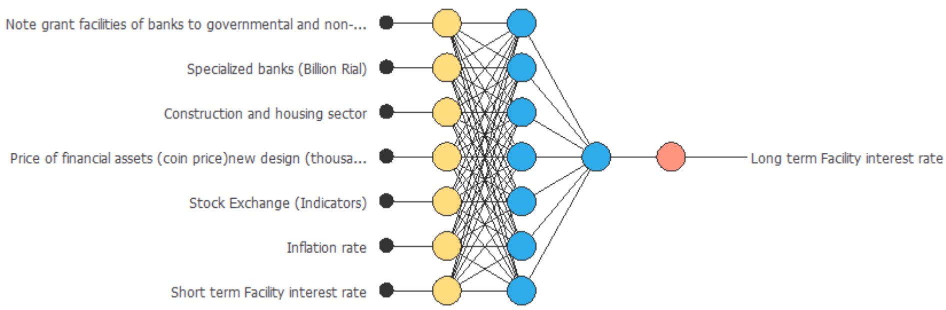

Fig 15. Final architecture

The next chart illustrates the linear regression for the scaled output Long Term Facility interest rate. Again, the predicted values are plotted versus the actual ones as circles. Again, the grey line indicates the best linear fit.

This part measures all the errors of the model. It takes into account every used instance and evaluates the model for each use. The next table shows all the errors of the data for each use of them.

Table 9. Error table

	Training	Selection	Testing
Sum squared error	0.0721675	0.0359317	0.0447423
Mean squared error	0.00257741	0.00449147	0.00559278
Root mean squared error	0.0507682	0.0670184	0.0747849
Normalized squared error	0.0368375	0.0909744	0.0503351
Minkowski error	0.0288267	0.122164	0.148644

There are some more details in the appendix, which has done with MATLAB software.

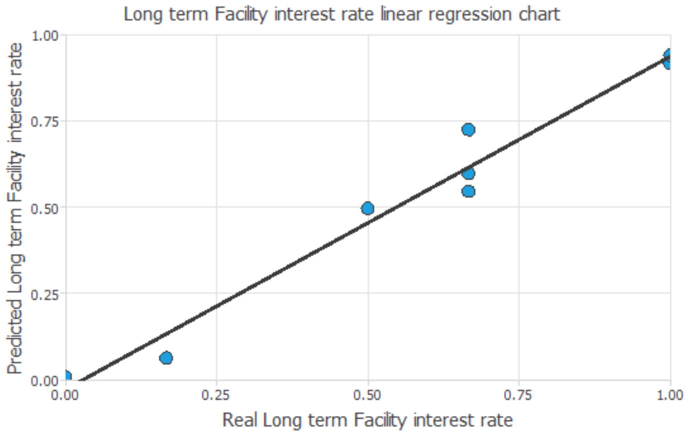

Fig 16. Target parameter linear regression chart

4.2. GWO-PSO algorithm

In this section, we have tried to provide very concise results to avoid prolonging the content. Therefore, the following important variables have considered as input variables:

Table 10. Feature selection using GWO-PSO algorithm

No	Variables	The variable belongs to the desired subset	Selected (1) or not (0)
1	Liquidity in terms of its components (Billion Rial)	Monetary and credit variables (liquidity)	0
2	Banks and non-bank credit institutions (Billion Rial)	Monetary and credit variables (Note grant facilities of banks to governmental and non-governmental sectors)	1
3	Commercial Banks (Billion Rial)	Monetary and credit variables (Note grant facilities of banks to governmental and non-governmental sectors)	1
4	Specialized banks (Billion Rial)	Monetary and credit variables (Note grant facilities of banks to governmental and non-governmental sectors)	1
5	Banks and non-bank credit institutions (Billion Rial)	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)	1
6	Commercial Banks (Billion Rial)	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)	0
7	Specialized banks (Billion Rial)	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)	0
8	Non-governmental banks and non-bank credit institutions	Monetary and credit variables (Facilities granted by banks and non-bank credit institutions by contracts)	0
9	Crude oil production (one thousand barrels per day)	Energy Section (Oil)	1
10	Crude oil exports (thousand barrels per day)	Energy Section (Oil)	0
11	All urban areas (without units)	Construction and housing sector (land price index)	0
12	Official rate (Rials)	Price of financial assets (exchange rate and coin price), exchange rate, US dollars	1
13	new design (thousand Rials)	Price of financial assets (exchange rate and coin price)	1
14	Total index (no units)	Price indices (price index of consumer goods and services) (100 = 1383)	0
15	Total index (no units)	Stock Exchange (Indicators)	1
16	Inflation rate	Price indices	1
17	Short term	Facility interest rate	1

Among these 17 indicators, 10 indicators have been selected as input variables, and others have not been chosen. It is clear that in both feature selection methods means GA and GWO-PSO algorithms, there are commonly selected variables in them Below table shows the results:

Table 11. 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

4.3. MFO algorithm

First of all, let's set the parameters such as the following table:

Table 12. MFO parameters

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

Below figures show the fitness function and convergence during iterations:

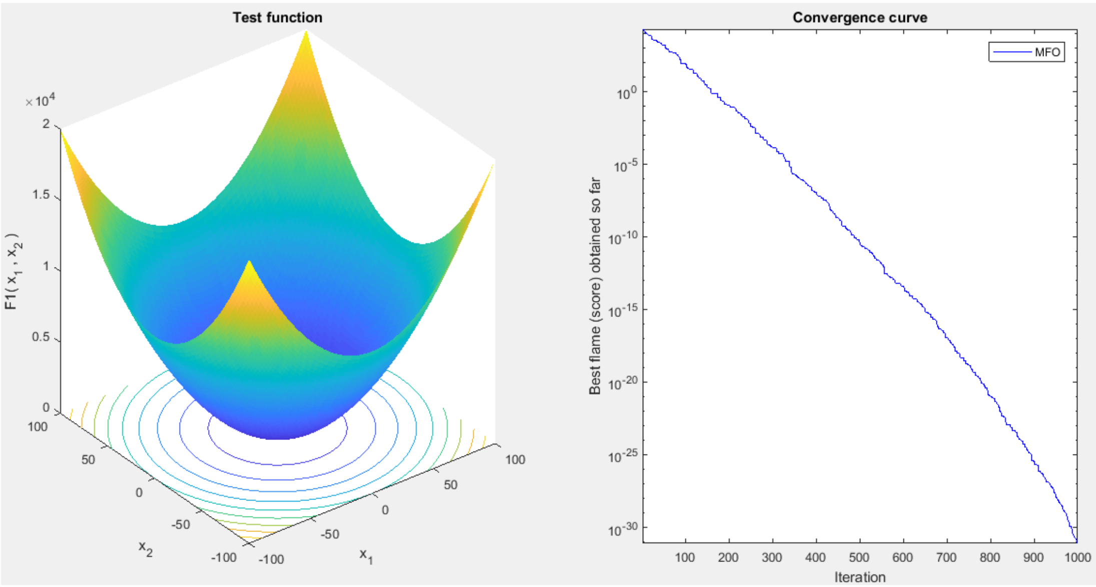

Fig 17. Test Function and Convergence Curve

You can see a clear decrease in each iteration until the best score has been obtained means 8.0081e-32.

4.4. WOA

Like the MFO algorithm, first of all, let's set the parameters such as the following table:

DOI: 10.22067/ijaaf.2021.67957.1017

Table 13. WOA parameters

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

Below figures show the fitness function and convergence during iterations:

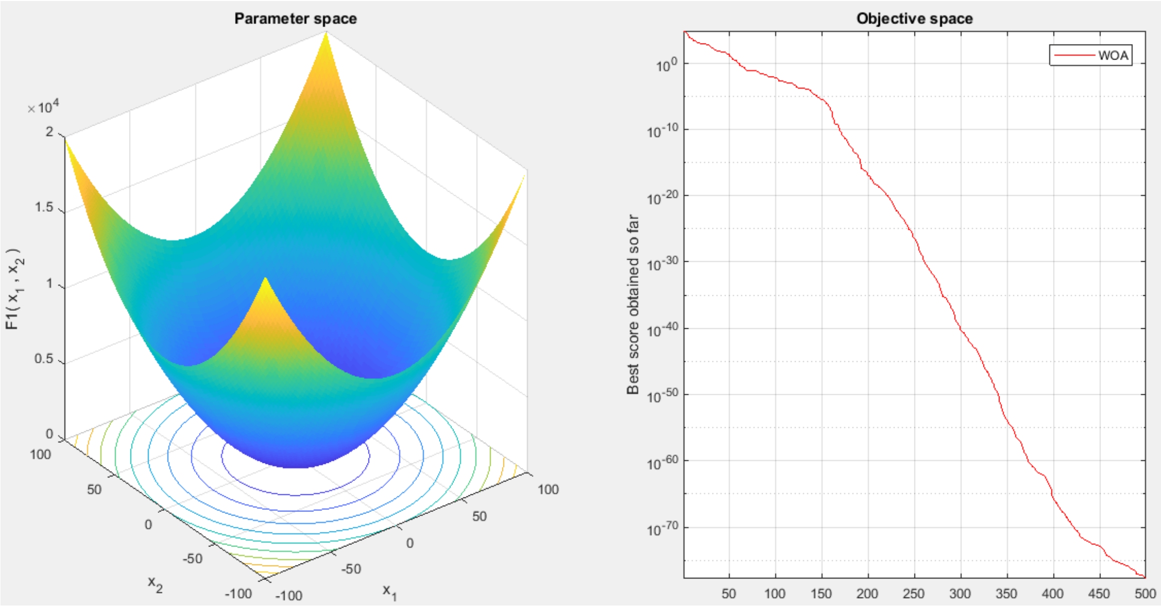

Fig 18. Test Function and Convergence Curve

4.5. MPSO, MPSO-TVAC, CHOA

In this part, we run these three algorithms together, but the result has been depicted separately. The following table shows the parameters:

Table 14. 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

CHOA, MPSO, MPSO-TVAC, CHOA has the lowest error among these three algorithms. The following figures show the chaotic map for types ChOA1 and ChOA2 after 500 iterations:

You can see that chasers with driver and attacker with barrier almost have the same behavior, but they follow different strategies as we told in previous.

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

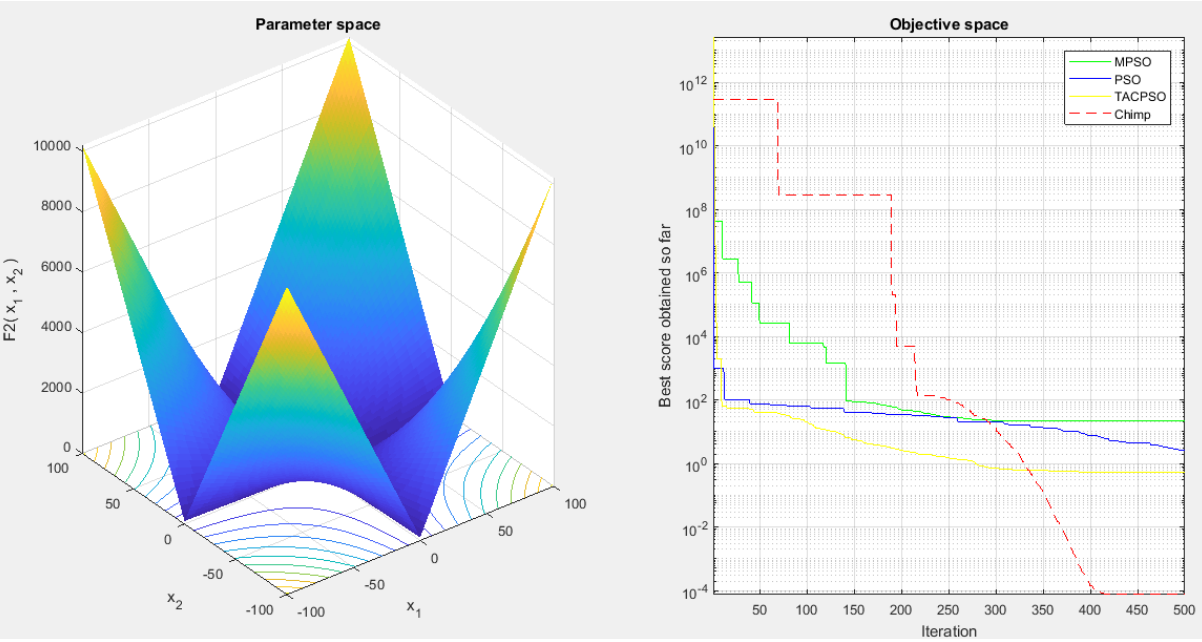

Fig 19. Test Function and Convergence Curve

As it is clear, between these three algorithms, CHOA, MPSO-TVAC, and MPSO have the lowest error and optimal solutions, respectively. CHOA has had a very sharp decline compared to other algorithms.

We set the methods according to their MSE (from minimum to maximum error) to understand results sensibly.

Table 15. Arrange algorithms based on MSE

ROW	Algorithm	MSE
1	WOA	1.6828e-78
2	MFO	8.0081e-32
3	ANN	1e-12
4	ChOA	7.4341e-05
6	GA-ANN	0.00041
6	GWO-PSO	0.004615
7	MPSO-TVAC	0.5373
8	MPSO	21.4027

5. Conclusions

In this article, we used a neural network as a prediction method for predicting long-term interest rates. Then, we used important economic variables such as oil price, inflation rate, etc., as input variables. So, we selected the most important one using GA and GWO-PSO. After that, we trained the network by using novel meta-heuristic algorithms such as MFO, MPSO, MPSO-TVAC, WOA, and CHOA.

We compute different loss functions for each algorithm after obtaining optimized input variables and weights using GA and GWO-PSO. As deduced from table (31) among algorithms, WOA has the lowest training and testing error. I should note that MPSO has the highest forecasting error. To evaluate the model's performance, we should test the model with new data called testing performance and is the right indicator for forecasting performance. We used GA and GWO-PSO as feature selection, and our focus is on other algorithms. The main advantages of using novel meta-heuristic algorithms are as follow:

- Speed up calculations
- Reduce model complexity
- Increase the network accuracy
- Ease of using models

Our suggestion for future research is to focus on different parameters such as the number of the hidden layer, activation function, and the other HS models such as HIS, etc., using different parameters of GA such as crossover and mutation rate be interesting. One of the other offers is a training network with other new metaheuristic algorithms such as the bat algorithm. We can test other algorithms as feature selection too.

References

1. Abdullah, M. N. Bakar, A. H. A. Rahim, N. A. Mokhlis, H. Illias, H. A. and Jamian, J. J. (2014). Modified particle swarm optimization with time varying acceleration coefficients for economic load dispatch with generator constraints. *Journal of Electrical Engineering & Technology*, 9(1), 15-26. http://dx.doi.org/10.5370/JEET.2013.8.5.742
2. Abiodun, O. I. Jantan, A. Omolara, A. E. Dada, K. V. Mohamed, N. A. and Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. Heliyon, 4(11), e00938. https://doi.org/10.1016/j.heliyon.2018.e00938
3. Abraham, A. (2005). Artificial neural networks. Handbook of measuring system design.
4. Al-Tashi, Q. Kadir, S. J. A. Rais, H. M., Mirjalili, S. and Alhussian, H. (2019). Binary optimization using hybrid grey wolf optimization for feature selection. *IEEE Access*, 7, 39496-39508.
5. Amin, I. A. Mahmood, D. Y. and Numan, A. H. (2020, July). Optimal location of UPFC devices for minimizing Losses in Transmission Line. In IOP Conference Series: Materials Science and Engineering. https://doi.org/10.1088/1757-899X/881/1/012129
6. Dagmar Blatná and JiříTrešl, (2011), Financial Forecasting Using Neural Networks China-USA Business Review, David Publishing ISSN 1537-1514, 10(3), 169-175.
7. Davallou, M. and Azizi, N. (2017). The Investigation of Information Risk Pricing; Evidence from Adjusted Probability of Informed Trading Measure. *Financial Research Journal*, 19(3), 415-438. *https://doi.org/*10.22059/jfr.2018.251305.1006600
8. Desan, C. (2008). From blood to profit: Making money in the practice and imagery of early America. Journal of Policy History, 20(1), 26-46. *https://doi.org/10.1353/jph.0.0010*
9. Faris, H. Aljarah, I. and Mirjalili, S. (2016). Training feedforward neural networks using multi-verse optimizer for binary classification problems. *Applied Intelligence*, 45(2), 322-332. https://link.springer.com/article/10.1007/s10489-016-0767-1
10. Friedman, B. M. (1977). Financial flow variables and the short-run determination of long-term interest rates. *Journal of Political Economy*, 85(4), 661-689.
11. Friedman, M. (1966). Interest rates and the demand for money. *The Journal of Law and Economics*, 9, 71-85.
12. Ghasemiyeh, R. Moghdani, R. and Sana, S. S. (2017). A hybrid artificial neural network with metaheuristic algorithms for predicting stock price. *Cybernetics and Systems*, 48(4), 365-392.
13. https://doi.org/10.1080/01969722.2017.1285162
14. Göçken, M. Özçalıcı, M. Boru, A. and Dosdoğru, A. T. (2016). Integrating metaheuristics and artificial neural networks for improved stock price prediction. *Expert Systems with Applications*, 44, 320-331. https://doi.org/10.1016/j.eswa.2015.09.029

15. Goodfriend, M. (1993). Interest rate policy and the inflation scare problem: 1979-1992. FRB *Richmond Economic Quarterly*, 79(1), 1-23.
16. Hadavandi, E. Ghanbari, A. and Abbasian-Naghneh, S. (2010). Developing an evolutionary neural network model for stock index forecasting. Paper presented at the International Conference on Intelligent Computing.
17. Haider, A. and Hanif, M. N. (2009). Inflation forecasting in Pakistan using artificial neural networks. *Pakistan economic and social review*, 47(1), 123-138. https://www.jstor.org/stable/25825345
18. Hassanin, M. F. Shoeb, A. M. and Hassanien, A. E. (2016). Grey wolf optimizer-based back-propagation neural network algorithm. Paper presented at the 2016 12th International Computer Engineering Conference (ICENCO). https://ieeexplore.ieee.org/abstract/document/7856471
19. He, J. and Guo, H. (2013). A Modified Particle Swarm Optimization Algorithm. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 11(10), 6209-6215. https://doi.org/10.11591/telkomnika.v11i10.2947
20. Huang, C. F. Chang, B. R. Cheng, D. W. and Chang, C. H. (2012). Feature Selection and Parameter Optimization of a Fuzzy-based Stock Selection Model Using Genetic Algorithms. International *Journal of Fuzzy Systems*, 14(1), 65-75.
21. Idris, M. A. Saiang, D. and Nordlund, E. (2015). Stochastic assessment of pillar stability at Laisvall mine using Artificial Neural Network. *Tunnelling and Underground Space Technology*, 49, 307-319. https://doi.org/10.1016/j.tust.2015.05.003
22. Jain, A., and Kumar, A. M. (2007). Hybrid neural network models for hydrologic time series forecasting. *Applied Soft Computing*, 7(2), 585-592.
23. Jensen, M. C. (2002). Value maximization, stakeholder theory, and the corporate objective function. *Business ethics quarterly*, 235-256.
24. Khishe, M. and Mosavi, M. R. (2020). Chimp optimization algorithm. Expert Systems with Applications, 113338. https://doi.org/10.1016/j.eswa.2020.113338
25. Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249. https://doi.org/10.1016/j.knosys.2015.07.006
26. Monfared, J.H., Alinejad, M.A., & Metghalchi, S. (2012). A comparative study of neural network models with box Jenkins methodologies in prediction of Tehran price index (tepix). *Quarterly of Journal Financial Engineering and Securities Management (portfolio management)*, 3(11): 1-16. (*in Persian*)
27. Mortezapour, R. and Afzali, M. (2013). Assessment of customer credit through combined clustering of artificial neural networks, genetics algorithm and Bayesian probabilities. arXiv preprint arXiv:1312.7740.
28. Peek, J. and Rosengren, E. S. (2010). The role of banks in the transmission of monetary policy (pp. 257-277). Oxford: Oxford University Press.
29. Prasanna, S. and Ezhilmaran, D. (2013). An analysis on stock market prediction using data mining techniques. *International Journal of Computer Science & Engineering Technology (IJCSET)*, 4(3), 49-51. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.480.90&rep=rep1&type=pdf
30. Qasim, N. Rind, M. Q. Sheikh, M. S. (2013). NEURAL NETWORKS A GATEWAY TO NON-LINEAR DATA MODELING TECHNIQUES. *Islamic Countries Society of Statistical Sciences*, 24, 115-126.
31. Rather, A. M. Sastry, V. and Agarwal, A. (2017). Stock market prediction and Portfolio selection models: a survey. Opsearch, 54(3), 558-579. https://link.springer.com/article/10.1007/s12597-

DOI: 10.22067/ijaaf.2021.67957.1017

016-0289-y

32. Shaheen, M. A. Hasanien, H. M. and Alkuhayli, A. (2020). A novel hybrid GWO-PSO optimization technique for optimal reactive power dispatch problem solution. Ain Shams Engineering Journal. 12(1), 621-630. https://doi.org/10.1016/j.asej.2020.07.011
33. Sheela, K. G. and Deepa, S. N. (2013). Review on methods to fix number of hidden neurons in neural networks. Mathematical Problems in Engineering, 2013.
34. Subramanyam, T. (2016). Selection of input-output variables in data envelopment Analysis-Indian commercial banks. International Journal of Computer & Mathematical Sciences, 5(6), 2347-8527.
35. Taylor, S. J. (2008). Modelling financial time series. World scientific.
36. Trujillo-Ponce, A. (2013). What determines the profitability of banks? Evidence from Spain. Accounting and Finance, 53(2), 561-586. https://doi.org/10.1111/j.1467-629X.2011.00466.
37. Vercellis, C. (2009). Business intelligence: data mining and optimization for decision making (pp. 1-420). New York: Wiley.
38. Yasir, M. Afzal, S. Latif, K. Chaudhary, G. M. Malik, N. Y.,Shahzad, F. and Song, O. Y. (2020). An Efficient Deep Learning Based Model to Predict Interest Rate Using Twitter Sentiment. Sustainability, 12(4), 1660. https://doi.org/10.3390/su12041660
39. Zhang, T. and Wu, W. B. (2011). Testing parametric assumptions of trends of a nonstationary time series. Biometrika, 98(3), 599-614. https://doi.org/10.1093/biomet/asr017