International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:13, No:3, 2019

An IM-COH Algorithm Neural Network Optimization with Cuckoo Search Algorithm for Time Series Samples

Wullapa Wongsinlatam

***Abstract*—**Back propagation algorithm (BP) is a widely used technique in artificial neural network and has been used as a tool for solving the time series problems, such as decreasing training time, maximizing the ability to fall into local minima, and optimizing sensitivity of the initial weights and bias. This paper proposes an improvement of a BP technique which is called IM-COH algorithm (IM-COH). By combining IM-COH algorithm with cuckoo search algorithm (CS), the result is cuckoo search improved control output hidden layer algorithm (CS-IM-COH). This new algorithm has a better ability in optimizing sensitivity of the initial weights and bias than the original BP algorithm. In this research, the algorithm of CS-IM-COH is compared with the original BP, the IM-COH, and the original BP with CS (CS-BP). Furthermore, the selected benchmarks, four time series samples, are shown in this research for illustration. The research shows that the CS-IM-COH algorithm give the best forecasting results compared with the selected samples.

***Keywords*—**Artificial neural networks, back propagation algorithm, time series, local minima problem, metaheuristic optimization.

I. Introduction

THE concept of artificial neural network (ANN) is to create mathematical model reflecting the system of brain functions that is a collection of neurons with connecting synapses. The collection is organized into the input layer, the hidden layer, and the output layer [1]-[4]. ANN is used in various kinds of problems such as forecasting and classification. In time series applications, ANN has been used in predicting carbon dioxide emissions, power load forecasting, as well as Dow-Jones industrial average. Still, there are three main limits to ANN, that are, slow training, easy to fall into local minima, and very sensitive for the choice of the initial weights and bias.

Recently, BP is adopted as another technique in ANN. The BP is more understandable and easily in programming. The BP is structured into two main parts: forward pass and backward pass. However, the BP contains big algorithm and requires big data for training process which takes a long time [5]-[7]. The main problems of the BP are local minima problem and its slow speed in convergence problem, which lessen the ability to fall into global minimum [8], [9]. As a result, there are many works of research focusing in finding a better performed algorithm that has global optimal solution, good generalization performance, as well as better training in order to a accelerate training speed.

W. Wongsinlatam is with Faculty of Applied Science and Engineering, Khon Kaen University, Nong Khai Campus, Nong Khai, 43000, Thailand (phone: +66-956-463-692; e-mail: wullwon@kku.ac.th).

Many years ago, the researchers were striving for the process that can solve these problems of local minima problem and slow convergence problem. One of the best performed algorithms in improving the BP is Gradient Descent Algorithm (GD) and Levenberg Marquardt Algorithm (LM) [10-13]. These algorithms used the choice of optimization methods as the activation function to apply to network. Still, it can only solve some part of the problems. Therefore, a new algorithm that has been proposed to solve optimization problems is metaheuristic optimizations. Metaheuristic optimizations are the most popular algorithms that are used to solve local minima problem and slow of convergence problem for the best performance of network by combining the BP with metaheuristic optimization, e.g., particle swarm optimization (PSO) [14], [15], artificial bee colony optimization (ABC) [16]-[18], and genetic algorithm (GA) [19], [20]. Furthermore, a cuckoo search algorithm (CS) is developed in recent years by Yang and Deb [21] as a new metaheuristic optimization that is solved optimization problem [22]-[25]. In the testing of the CS-BP algorithm performance, the result shown that the CS-BP algorithm gave a better result than the PSO-BP and GA-BP algorithms [26]. Meanwhile, the research of Nazri et al. has improved the LM algorithm in the training of the BP algorithm to solve local minima and slow of convergence in the CS algorithm. The new algorithm is called CS-LM. In the experiment where the CS-LM algorithm was compared with the ABC-LM, ABC-BP and BP algorithms, the optimal results of the CS-LM algorithm shown good performance in the case studies in term for convergence speed and rate. However, the problem of slow speed of convergence in training was improved very little in some part for the network [24].

In 2016, Wongsinlatam has improved the BP and proposed a new algorithm called Improve Control Output Hidden Layer Algorithm (IM-COH) [27]. This algorithm was developed by controlling the result in the output hidden layer together with using the BP to reduce noise by adding penalty terms to the objective function. The research tested the performance of the IM-COH algorithm with two-spiral classification problem and Mackey-Glass time series prediction. The result shows that the IM-COH algorithm gave a better performance in solving classification problem than time series prediction. Furthermore, the IM-COH algorithm performed better than the BP algorithm. However, IM-COH algorithm has limit in solving slow speed in convergence problem.

In this paper, the process of the IM-COH algorithm is combined with the CS algorithm to address the IM-COH

algorithm limit. The experiment is divided into three parts: determining the IM-COH structure, obtaining the best weights and bias through the CS algorithm, and predicting through neural network. This paper is presented as follows. The first part explains the proposed algorithms which are the BP, IM-COH, and CS algorithms. Then the paper presents the new algorithm CS-IM-COH in Section II. In Section III, the simulation results and discussions are shown to compare the performances on the four time series samples. Finally, the final section shows conclusions and presents the points of the future work orientation.

II. Proposed Algorithms

A. Back-Propagation (BP)

The one of the most widely used neural network algorithm is a BP Algorithm. The BP algorithm is a common method for training neural network. The topology of the BP algorithm has input layer, hidden layer and output layer. The two parts have the forward and the back transmission and the back transmission of network for the learning and training process. The notations used in the BP are described below:

x_i^l	output of neuron i in the l^{th} layer,
d_j	desired output of neuron j in the output layer,
$y_j^l = f(b_j^l)$	actual output of neuron j in the l^{th} layer,
b_j^l	activation of neuron j in the l^{th} layer
	weight connecting neuron i in $l-1^{th}$ layer to neuron j in the l^{th} layer,
Δw_{ji}^l	weight update value,
α	learning rate,
E	global error,
P	number of layers.

The purpose of the BP algorithm is to optimize the weights and the global error can be defined as:

$$E = [\frac{1}{2}\sum_j (d_j - y_j^P)]^2 \tag{1}$$

The basic equation of the BP algorithm for the output layer is:

$$\begin{aligned} w_{ji}^P(t+1) &= w_{ji}^P(t) + \alpha \frac{\partial(f(b_j^P))}{\partial(b_j^P)} \cdot (d_j - y_j^P) \cdot x_i^{(P-1)}, \\ &= w_{ji}^P(t) + \alpha \delta_j^P x_i^{(P-1)}, \end{aligned} \tag{2}$$

where δ_j^P is clarified to be

$$\delta_j^P = \frac{\partial(f(b_j^P))}{\partial(b_j^P)}, \tag{3}$$

the weight equation is used to update for hidden layer,

$$w_{ji}^l(t+1) = w_{ji}^l(t) + \alpha \delta_j^l x_i^{(l-1)}, \tag{4}$$

where

$$\delta_j^l = \frac{\partial(f(b_j^l))}{\partial(b_j^l)} \cdot \sum_k \delta_k^{l+1} w_{kj}^{l-1}. \tag{5}$$

Then the weight for the hidden nodes are carried out. The BP equations provide a way of computing the gradient for the error function.

The concrete steps of the BP algorithm are the forward part of the input signal of network and the back part of the error signal. The steps start with presenting the inputs at the input layer with a set of the corresponding activation function for the input layer, then feedforward, follows with output error and compute the vector, backpropagate the error, and output for the gradient of the global error function, respectively [3], [28].

B. IM-COH Algorithms (IM-COH)

The IM-COH algorithm (IM-COH) developed in 2016 by Wongsinlatam [27], by adding new term of the criterion function or the global error at (1) that is used in the basic of the BP algorithm. In the new term, the global error of the IM-COH algorithm can be defined as:

$$E = E + \varphi \cdot \Delta T, \tag{6}$$

where φ is a small value of parameter, ΔT is a new term, the training of the parameters of the IM-COH algorithm is required in order to minimize the output of the hidden layers according to (6).

The new term can be defined as:

$$\Delta T = \begin{cases} \sum_{l=1}^{P-1} \sum_j -f(b_j^l) & ;\forall j, l; f(b_j^l) \geq 0, \\ \sum_{l=1}^{P-1} \sum_j f(b_j^l) & ;\forall j, l; f(b_j^l) < 0. \end{cases} \tag{7}$$

Next, the weight is updated as follows:

$$w_{ji}^l(t+1) = \begin{cases} w_{ji}^l(t) + \alpha \delta_j^l x_i^{l-1} - \varphi \alpha u_j^l x_i^{l-1}, \\ w_{ji}^l(t) + \alpha \delta_j^l x_i^{l-1} + \varphi \alpha u_j^l x_i^{l-1}, \end{cases} \tag{8}$$

where

$$u_j^P = 0, \tag{9}$$

and,

$$u_j^l = \begin{cases} \frac{\partial(f(b_j^l))}{\partial(b_j^l)} \cdot \sum_k u_k^{l+1} w_{kj}^{l+1} - \frac{\partial(f(b_j^l))}{\partial(b_j^l)}, \\ \frac{\partial(f(b_j^l))}{\partial(b_j^l)} \cdot \sum_k u_k^{l+1} w_{kj}^{l+1} + \frac{\partial(f(b_j^l))}{\partial(b_j^l)}. \end{cases} \tag{10}$$

The index of k runs all node in the $l+1^{th}$ layer and the l^{th} layer is connected by the j^{th} node. Then the weight in case 1 and 2 at (8) are used in (7) for case 1 and 2 accordingly, the derivative algorithm is defined to adjust the weights for the output nodes in (10) and according to (7) for case 1 and 2, respectively. The IM-COH algorithm is similar to BP algorithm in one epoch, however, this algorithm adds the new term to the error function to control the outputs nodes for hidden layers.

C. Cuckoo Search Algorithm (CS)

The CS algorithm is proposed by Yang and Deb for the optimization problem [21]. The CS algorithm is a global search algorithm for finding an optimum solution. The algorithm of the CS are composed of the following. First, for all of cuckoo in the population, every bird lays only one egg at the time and randomly selects a nest in order to place its egg. Second, the population cannot change the eggs with the best fit in order to make the whole population evolve forward. Third, the host bird discovers the cuckoo eggs with probability (P_a) and determine $P_a \in [0, 1]$. In this case, the cuckoo has no other choice and it has to build a fully new nest. In generating the new solution, x_i^{t+1} cuckoo L*é*vy flight can be defined as:

$$x_i^{t+1} = x_i^t + \alpha_1 \oplus \text{Lévy } (\alpha), \tag{11}$$

and

$$\text{Lévy} \sim u = t^{-\lambda}; \quad 1 < \lambda \leq 3, \tag{12}$$

where α_1 is the step size and the product $\oplus$ means entry wise multiplications. The CS algorithm initialed the population n for the nest, and randomly selected the best nest via Lvy flight. However, the most critical parameters required to obtain the optimal solution from the CS algorithm are P_a and α_1. The steps of the CS algorithm can be shown as follows in Algorithm 1.

Algorithm 1: The steps of the CS algorithm

Objective function $f(x)$, $x = (x_1, ..., x_n)^T$;
Initial a population of n host nests x_i,
for all $i = (1, 2, ..., n)$;
  while (t ¡ MaxGeneration) or (stop criterion);
    Get a cuckoo (say i) randomly by L*é*vy flights;
    Evaluate its quality/fitness F_i;
    Choose a nest among n (say j) randomly;
  if $(F_i > F_j)$
    Replace j by the new solution;
  end
    Abandon a fraction (P_a) of worse nests;
    build new ones at new locations via L*é*vy flights;
    Keep the best solutions;
    Ranking the solutions and finding the current;
end while

III. The Algorithm

A. The CS with IM-COH Algorithm (CS-IM-COH)

The new algorithm called Cuckoo Search Improve Control Output Hidden Layer Algorithm (CS- IM-COH) is the combination of the algorithm of IM-COH with the CS algorithm for the optimization of weights and bias through the CS algorithm in forecasting problem. The particularized steps of the CS-IM-COH algorithm can be presented as follows. First, determining the IM-COH structure, then the initial of weights and bias are randomed, and then they are encoded according to the algorithm of CS. The encoded weights and bias are put into the CS algorithm in order to optimize the IM-COH network. Second, the construct of the CS-IM-COH network, the optimal weights and bias obtained from the algorithm of the CS algorithm are used to construct the CS-IM-COH network and the training error is calculated. When, the training error meets the requirements, the training of the CS-IM-COH network stops. Finally, the test set is put into the trained of CS-IM-COH network to predict output.

There are six steps, initializing the CS algorithm, determining fitness function, updating position operator, selecting operator, replacing operator, and eliminating operator in order to find the cuckoo individual with the best fitness. The presented steps of the CS-IM-COH algorithm can be explained as the following:

Step 1: The cuckoo individual is encoded, that is a composed of real number string. Follow with connection weights between the hidden layer and the output layer, connection weights between the hidden layer and the input layer, connecting the bias in the output layer and the hidden layer. Each cuckoo individual contains all the weights and bias in the IM-COH algorithm.

Step 2: The initial weights and bias of the IM-COH algorithm can be determined according to the best individual. After training the IM-COH algorithm, the weights and bias are used to predict the output. The fitness value of cuckoo individual F is the sum of the absolute error as follows:

$$F = c \cdot \sum_{i=1}^{n} |d_i - y_i|, \tag{13}$$

where n is the node number of the output layer in the IM-COH algorithm and c is a coefficient. d_i is the desired output and y_i is the predicted output for the node i in the IM-COH algorithm.

Step 3: A cuckoo is randomly chosen from the cuckoo population and its position is updated according to (11). The fitness (F_i) of the i^{th} cuckoo at generation t and position x_i^t is evaluated by (13).

Step 4: Another cuckoo is randomly chosen from the cuckoo population which is $i \neq j$, and its position fitness F_j of the i^{th} cuckoo at generation t and position x_j^t is evaluated by (13).

Step 5: Replacing operator, where the fitness value of the cuckoo i is bigger than the cuckoo j, that is, $F_i > F_j$, and x_j is replaced by the new solution.

Step 6: When the populations are in the finalized state, ceil $(n * Pa)$, the worst cuckoos are removed in each generation. In order to make the population size unchanged, ceil $(n * Pa)$ cuckoos would randomly be generated. The cuckoos with the best fitness will be passed directly to the next generation.

This optimization process is repeated until the satisfactory weights and bias are found. In the last part, the IM-COH algorithm with the optimal weights and bias is constructed and is trained to predict the output.

IV. Simulation Results and Discussions

Four time series samples from datamarket.com are used in this paper for analysis. First, Milk Production Samples, this samples have the information about the monthly milk production in the units of pounds per cow from January of 1962 to December of 1975. Second, Pigs Samples which

are information about the monthly total number of pigs slaughtered in Victoria in the units of number of pigs from Jannuary of 1980 to August of 1995. Third, Clay Bricks Samples which contains the quarterly production of clay bricks: million of units from March of 1956 to September of 1981. Finally, the sample is the Raw Steel Samples which are information about the monthly production of raw steel in Australia : thousand tonnes from January of 1956 to November of 1993. The appropriate sizes of time series samples to be used in this paper are shown as in-samples and out-of-samples; (In-samples=118 and Out-of-samples=50 for Milk Production Samples), (In-samples=132 and Out-of-samples=56 for Pigs Samples), (In-samples=109 and Out-of-samples=46 for Clay Bricks Samples), and (In-samples=319 and Out-of-samples=136 for Raw Steel Samples). The workstation used to carry out the result equipped with SCILAB. The SCILAB version 6.0.1 is a free software that is used to carry out simulation of the algorithm [29]. The global performance of forecasting was calculated by an accuracy of algorithms and MSE errors. The equation for MSE was given as follows:

$$MSE = \frac{1}{H}\sum_{j=T+1}^{T+H}(d_j - y_j), \tag{14}$$

where, T is the current time period, H is the forecasting horizon, d_j is original values which is normalized into the range $[0, 1]$, and y_j is the estimated forecast. In all results, the lower values of MSE indicate a better forecasting ability of the algorithm [30]

A. Comparing Performance of Algorithms

For the purpose of comparing performance in this section, the BP algorithm, IM-COH algorithm and the CS algorithm are tested to find optimal parameters estimation for algorithms and the parameters are set as follows. The results are recorded in Table I. For CS-IM-COH algorithm, epochs = 1000, learning rate = 0.1, sigmoid activation function and linear activation function for the output. For the CS algorithm part in CS-IM-COH algorithm, discovery rate P_a= 0.1, population size = 100, and maximum generation = 100. Table I showed that, for training dataset, the best performance and the average performance of the BP, IM-COH, CS-BP, and CS-IM-COH algorithms have little difference. Still, the CS-IM-COH algorithm performs slightly better than the CS-BP, IM-COH, and BP algorithms, respectively.

The CS-IM-COH algorithm is better than the CS-BP, and IM-COH algorithms and is significantly superior to the BP algorithm. For test dataset, the overall prediction accuracy of the CS-IM-COH algorithm is much better than all other algorithms. In addition, the Std. (standard deviation) of the CS-IM-COH algorithm is clearly less than the CS-BP, IM-COH, and BP algorithms. To conclude, the CS-IM-COH algorithm generates a more stable prediction output with little fluctuation.

B. Comparing Convergence for Time Series Samples

For the results of time series samples, the stopping criteria of algorithms is maximum generation (100 generations). The obtained results are shown in Table II. The MSE error of the CS-IM-COH algorithm is lower which indicates a better forecast when compared with the CS-BP, IM-COH, and BP algorithms, respectively.

To summarize, the best performed algorithm is the CS-IM-COH algorithm. The result is shown with MSE error: Milk Production Samples 4.385×10^{-3}, Pigs Samples 4.830×10^{-2}, Clay Bricks Samples 2.640×10^{-2}, and Raw Steel Samples 3.477×10^{-3} where the error is less than the CS-BP, IM-COH, and BP algorithms in every time series samples. For illustration, Table III shows the number of inputs, hidden nodes, and total minimum time of network (second) of the two best algorithms optimized by the CS-BP and CS-IM-COH algorithms. The result indicates that, the total minimum time of the CS-IM-COH algorithm is always better than the CS-BP algorithm and greatly require less computation time in all of time series samples.

The CS-IM-COH algorithm has the best performances in all four time series samples. The CS-IM-COH algorithm for Milk Production Samples is used epochs = 1000, learning rate =

TABLE I
THE ACCURACY TRAIN AND TEST DATASET OF ALGORITHMS

Algorithms	Train dataset			
	Mean	Best	Worst	Std.
BP	87	89	87	1.69
IM-COH	88	89	88	0.95
CS-BP	88.75	89	88	0.74
CS-IM-COH	89	90	89	0.45
	Test dataset			
Algorithms	Mean	Best	Worst	Std.
BP	87.25	89	86	1.87
IM-COH	88.33	89	87	1.84
CS-BP	88.50	89	87	1.09
CS-IM-COH	89	90	88	0.97

TABLE II
THE FORECASTING COMPARISON (MSE ERRORS)

Time Series	MSE			
	BP	IM-COH	CS-BP	CS-IM-COH
Milk Production * (MSE $\times10^{-3}$)	115.862	53.865	4.507	4.385
Pigs * (MSE $\times10^{-2}$)	7.037	4.929	5.731	4.830
Clay Bricks * (MSE $\times10^{-2}$)	6.423	5.531	4.640	2.640
Raw Steel * (MSE $\times10^{-3}$)	102.431	49.153	4.785	3.477

TABLE III
COMPARISON OF THE BEST ALGORITHMS FOR OPTIMIZATION

Time Series	CS-BP		
	Input	Hidden	Time (sec.)
Milk Production	118	64	107
Pigs	132	67	121
Clay Bricks	109	59	98
Raw Steel	319	120	337
Time Series	CS-IM-COH		
	Input	Hidden	Time (sec.)
Milk Production	118	58	85
Pigs	132	60	105
Clay Bricks	109	46	74
Raw Steel	319	98	290

0.1, sigmoid activation function and linear activation function for the output. For the CS algorithm part in CS-IM-COH algorithm, discovery rate P_a= 0.1, population size = 100, and maximum generation = 100. The CS-IM-COH algorithm for Pigs Samples is used epochs = 1000, learning rate = 0.1, sigmoid activation function and linear activation function for the output. For the CS algorithm part in CS-IM-COH algorithm, discovery rate P_a= 0.2, population size = 100, and maximum generation = 100. The CS-IM-COH algorithm for Clay Bricks is used epochs = 1000, learning rate = 0.2, sigmoid activation function and linear activation function for the output. For the CS algorithm part in CS-IM-COH algorithm, discovery rate P_a= 0.4, population size = 100, and maximum generation = 100.The CS-IM-COH algorithm for Raw Steel Samples is used epochs = 1000, learning rate = 0.1, sigmoid activation function and linear activation function for the output. For the CS algorithm part in CS-IM-COH algorithm, discovery rate P_a= 0.3, population size = 100, and maximum generation = 100.

The simulation shows that, the best performances of total minimum time for Clay Bricks, Milk Production, Pigs, and Raw Steel Samples of the CS-IM-COH algorithm (74, 85, 105, and 290 sec.) is clearly faster than the best performances of total minimum time of the CS-BP algorithm, respectively.

V. Conclusions

This paper, proposes an improvement of a BP algorithm, called the IM-COH. By combining cuckoo search algorithm (CS) with IM-COH, the result algorithm is called the CS-IM-COH. This new algorithm is proposed to be used to optimize the initial weights and bias of artificial neural network. In this paper, the CS-IM-COH is compared with the original BP, the original CS, and the original CS-BP for illustration. Furthermore, four time series samples are selected as case studies to test its performances. The comparing parameters that are used in this paper are; the accuracy of train and test, the MSE errors of forecasting, and the total minimum time. The research result confirms that CS-IM-COH performs best in convergence rate and local minima problem.

In the future, the author would like to focus on research in three prospects which are; to improve other algorithms of neural networks, to modify the groups of metaheuristic optimizations (e.g., CS, ABC, and PSO), and to combine those results to solve other optimization problems.

Appendix

A. Summary of Acronyms and Their Meaning Used

ANN:	Artificial Neural Network
BP:	Back Propagation Algorithm
CS:	Cuckoo Search Algorithm
IM-COH:	Improve Control Output Hidden Layer Algorithm
CS-BP:	Cuckoo Search Back Propagation algorithm Algorithm
CS-IM-COH:	Cuckoo Search Improve Control Output Hidden Layer Algorithm.

Acknowledgments

This research is supported by Research and Technology Transfer Affairs, Khon Kaen University together with the Faculty of Applied Science and Engineering, Khon Kaen University, Nong Khai Campus. Furthermore, the author would like to thank all referees for thorough reading and valuable suggestions.

References

[1] W. S. Mcculloch and W. Pitts, *A logical calculus of the ideas immanent in Nervous activity*, Bulletin of Mothemnticnl Biology19, 90, 52, 99-115.
[2] S. E. Fahlman and C. Lebiere, *The cascade-correlation learning architecture*, Advances in Neural Information Processing. 1990, 2, 524532.
[3] G.Villarrubia, F. D. P. Juan, P. Chamoso and D. l. P. Fernando, *Artificial neural networks used in optimization problems*. Neurocomputing, 2017, 1-7, https://doi.org/10.1016/j.neucom.2017.04.075.
[4] S. Selva, K. Emin, K. Seyit, K. Hatem and G. Elcin, *Artificial Neural Network and Agility*, Procedia - Social and Behavioral Sciences, 2015, 195, 1477-1485, https://doi.org/10.1016/j.sbspro.2015.06.448.
[5] A. Wam, S. Esm, and A. Esa, *Modified Back Propagation Algorithm for Learning Artificial Neural Networks*, Eighteenth National Radio Science Conference (NRSC), 2001, 345-352.
[6] R. Law, *Back-propagation learning in improving the accuracy of neural network-based tourism demand forecasting*, Tourism Management, 2000, 21, 331340.
[7] Z. Yu-Rong, Z. Yi, Ch. Beomjin, and W. Lin, *Multifactor-influenced energy consumption forecasting using enhanced back-propagation neural network*, Energy. 2017, 127, 381-396, https://doi.org/10.1016/j.energy.2017.03.094.
[8] D. Shounak, S. M. Sankha and D. Swagatam, *Generalized mean based back-propagation of errors for ambiguity resolution*, Pattern Recognition Letters, 2017, 94, 2229, https://doi.org/10.1016/j.patrec.2017.04.019.
[9] A. H. Alaa, K. Bekir and Sh. S. Mohammad, *Back-propagation algorithm with variable adaptive momentum*, Knowledge-Based Systems, 2017, 114, 79-87, https://doi.org/10.1016/j.knosys.2016.10.001.
[10] V. Ramana, P. Sunthar and R. Ch. Durgaprasada, *The generalized proportional-integral-derivative (PID) gradient descent back propagation algorithm*, Neural Networks, 1995, 8, 563-569, https://doi.org/10.1016/0893-6080(94)00100-Z.
[11] N. Bighnaraj, N. Janmenjoy and S. B. Himansu, *A Global-best Harmony Search based Gradient Descent Learning FLANN (GbHS-GDL-FLANN) for data classification*, Egyptian Informatics Journal, 2016, 17, 57-87, https://doi.org/10.1016/j.eij.2015.09.001.
[12] G. K. Bahram, S. Sch. Susan, H. Troy Nagle, *Performance of the LevenbergMarquardt neural network training method in electronic nose applications*, Sensors and Actuators B: Chemical, 2005, 110, 13-22, https://doi.org/10.1016/j.snb.2005.01.008.
[13] A. Shahrokh, H. Esmaeil, M. Farhad and N. M. Masoud, *Hybridization of evolutionary LevenbergMarquardt neural networks and data pre-processing for stock market prediction*, Knowledge-Based Systems, 2012, 35, 245-258, https://doi.org/10.1016/j.knosys.2012.05.003.
[14] R. Mendes, P. Cortez, M. Rocha and J. Neves, *Particle swarm for feed forward neural network training*, Proceedings of the International Joint Conference on Neural Networks, 2002, 2, 1895-1899.
[15] J. Zhang, T. Lok, M. Lyu, *A hybrid particle swarm optimization back propagation algorithm for feed forward neural network training*, Journal Applied Mathematics and Computation, 2007, 185, 1026-1037.
[16] H. Shah, R. Ghazali, N. M. Nawi and M. M. Deris, *Global hybrid ant bee colony algorithm for training artificial neural network*, International Conference on Computational Science and Its Applications, 2012, 7333, 87-100, $DOI : 10.1007/978 - 3 - 642 - 31125 - 3_7$.
[17] H. Shah, R. Ghazali and N. M. Nawi, *Hybrid ant bee colony algorithm for volcano temperature prediction*, Communications in Computer and Information Science, 2012, 281, 453-465, $DOI : 10.1007/978 - 3 - 642 - 28962 - 0_43$.
[18] D. Karaboga, B. Basturk and C. Ozturk, *Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks*, International Conference on Modeling Decisions for Artificial Intelligence, 2007, 4617, 318-319, $DOI : 10.1007/978 - 3 - 540 - 73729 - 2_30$.

[19] J. N. D. Gupta and R. S. Sexton, *Comparing backpropagation with a genetic algorithm for neural network training*, Omega The International Journal of Management Science, 1999, 27, 679-684, https://doi.org/10.1016/S0305-0483(99)00027-4.

[20] A. U. Khan, T. K. Bandopadhyaya and S. Sharma, *Comparisons of Stock Rates Prediction Accuracy using Different Technical Indicators with Backpropagation Neural Network and Genetic Algorithm Based Backpropagation Neural Network*, Proceedings of the First International Conference on Emerging Trends in Engineering and Technology IEEE Computer Society, 2008, 575-580.

[21] X. S. Yang and S. Deb, *Engineering Optimisation by Cuckoo Search*, International Journal of Mathematical Modelling and Numerical Optimisation, 2010, 1, 330-343.

[22] M. Tuba, M. Subotic and N. Stanarevic, *Modified cuckoo search algorithm for unconstrained optimization problems*, Proceedings of the European Computing Conference (ECC 11), 2011, 263-268.

[23] M. Tuba, M. Subotic N. and Stanarevic, *Performance of a Modified Cuckoo Search Algorithm for Unconstrained Optimization*, WSEAS TRANSACTIONS on SYSTEMS, 2012, 11, 263-268.

[24] M. N. Nazri, K. Abdullah and M. Z. Rehman, *A New Levenberg Marquardt based Back Propagation Algorithm Trained with Cuckoo Search*, Journal of Procedia Technology. 2013, 11, 18-23.

[25] Y. Jiao-hong, X. Wei-hong and C. Yuan-tao, *Novel Back Propagation Optimization by Cuckoo Search Algorithm*, Journal of The Scientific World Journal, 2014, 2014, Article ID 878262, http://dx.doi.org/10.1155/2014/878262.

[26] M. N. Nazri, K. Abdullah and M. Z. Rehman, *A New Back-Propagation Neural Network Optimized with Cuckoo Search Algorithm*, International Conference on Computational Science and Its Applications, 2013, 7971, 413-426.

[27] W. Wongsinlatam, *Manipulation of hidden layers to improve the generalization ability of neural networks*, AIP Conference Proceedings, 2016, 1705, 0200201-0200208, http://dx.doi.org/10.1063/1.4940268.

[28] H. Yonaba, F. Anctil and V. Fortin, *Comparing sigmoid transfer functions for neural network multistep ahead streamflow forecasting*, Journal of Hydrologic Engineering, 2010, 15, 275283.

[29] W. Wongsinlatam and S. Buchitchon, *The Comparison between Dragonflies Algorithm and Fireflies Algorithm for Court Case Administration: A Mixed Integer Linear Programming*, Journal of Physics: Conference Series, 2018, 1061(1), 012005.

[30] S. Kulkarni and I. Haidar, *Forecasting model for crude oil price using artificial neural networks and commodity futures prices*, International Journal of Computer Science and Information Security, 2009, 2, 18.