Evolutionary Algorithms For Neural Networks Binary And Real Data Classification

Dr. Hanan A.R. Akkar, Firas R. Mahdi

Abstract: Artificial neural networks are complex networks emulating the way human rational neurons process data. They have been widely used generally in: prediction, clustering, classification, and association. The training algorithms that used to determine the network weights are almost the most important factor that influence the neural networks performance. Recently many meta-heuristic and Evolutionary algorithms are employed to optimize neural networks weights to achieve better neural performance. This paper aims to use recently proposed algorithms for optimizing neural networks weights comparing these algorithms performance with other classical meta-heuristic algorithms used for the same purpose. However, to evaluate the performance of such algorithms for training neural networks we examine such algorithms to classify four opposite binary XOR clusters and classification of continuous real data sets such as: Iris and Ecoli.

Index Terms: Artificial neural networks, Classifications, Evolutionary algorithms, Population-based algorithms, Meta-heuristics techniques, and Optimization.

1 Introduction

Artificial Neural Networks (ANNs) are complicated networks emulating the way human rational neurons process data. They have been widely used in many life applications such as: classification [1], image recognition and processing [2], data mining [3], and robotics [4]. The most important issue related to this concept is how to train the networks to give the right solution with the most optimized weights. Training of ANN is the process of adjusting the interconnection weights of the neurons. One of the most popular training algorithms is the Back-Propagation algorithm (BP), this algorithm has been extensively used for the network training purpose. However, it seems to be suffering from multiple problems such as easy fall into local minima, and its low convergence speed [5]. Many attempts have been made to improve the performance of BP, while other just used meta-heuristic and evolutionary algorithms to replace BP algorithm in the training phase. Therefore, many meta-heuristic algorithms have been developed and applied to rise the performance of the training process such as: Genetic Algorithm (GA) [6], Simulated Annealing (SA) [7], Particle Swarm Optimization (PSO) [8], Artificial Bee Colony (ABC) [9], Differential Evolution Algorithm (DEA) [10]. This paper aims to optimize NNs weights using recently proposed meta-heuristic algorithms such as; Multi-Verse Optimizer (MVO) [11], Symbiotic Organisms Search (SOS) [12], Stochastic Fractal Search (SFS) [13], Novel Bat Algorithm (NBA) [14], and Bird Swarm Algorithm (BSA) [15], comparing them with other well-known classical algorithms abundance used in this field. Two classification experiments have been carried out for testing these algorithms performance training ANNs, the first experiment uses a binary input data for classification four different XOR problems clusters, while the second experiment are used to classify real input data such as Iris and Ecoli data sets.

- *Hanan A. Akkar is a prof. Within the Electrical Engineering department, University of Technology (UOT), Baghdad, Iraq. E-mail: dr_hananuot@yahoo.com*
- *Firas R. Mahdi is a PhD student within the Electrical Engineering department, University of Technology (UOT), Baghdad, Iraq. E-mail: firasrasool1980@gmail.com*

2 Multi-Layer Perceptron

Multi-layer Perceptron Networks (MLP) are the most popular feed-forward supervised ANNs. They essentially, consist of a single input, output layer, and one or more hidden layers. One hidden layer is usually sufficient to solve almost all types of problems. Using two hidden layers rarely improves the network performance, and it may leads for converging to a local minima. All hidden and output nodes are composed of neurons which represent the processing element of the layer. Each layer is fully connected to the next layer input and it consists of multiple neurons varies depending on the problem it solves. Each processing element consist of a summation and an activation function. Fig.1 shows a single neuron processing element. usually each processing layer have the same activation function for all of its elements, neurons may have a linear identity function or nonlinear activation function such as hyperbolic tangent, logistic and Gaussian function.

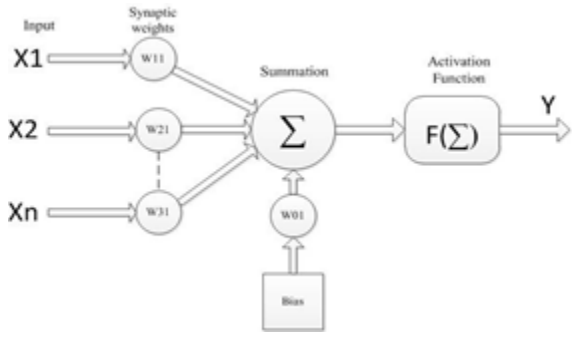

Fig.1. *single neuron processing element*

Using too many neurons in the hidden layer could result in over fitting problem, which occurs when the ANN has too much information and the amount of input data patterns information is not enough to optimize all the neurons weights in the hidden layers at the proposed time. Another problem occurs when the training data set is sufficient, but the amount of training time rise to the point that it is impossible to sufficiently train the ANN. Therefore, some compromise must be reached between too many and few neurons in the hidden layers, another compromising method using regularization the network by modifying the performance function, which is normally chosen to be the mean square error of the network on the training set. Once the number of neurons in hidden

layer, has been selected, the network's weights must be optimized to minimize the error made by the network. This is the role played by the training algorithms [16]. Fig. 2 shows a typical fully connected single hidden layer ANN with input and output layer in which each connection line represents a weight that must be adjust to adapt the suitable value to get the minimum mean square error.

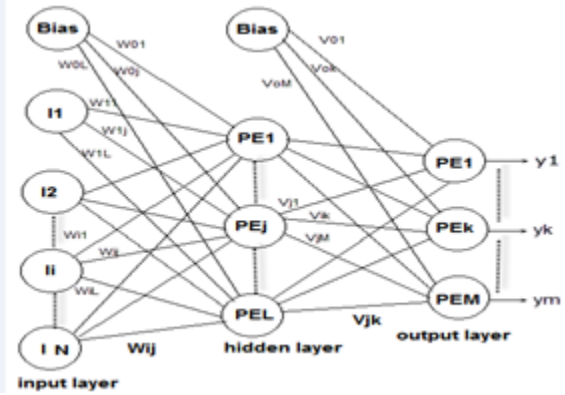

Fig. 2. *fully connected single hidden layer ANN*

3 META-HEURISTIC ALGORITHMS

The Greek words "meta" means after or beyond, while "heuristics" means to find or discover. The term meta-heuristic, can be defined as procedures that guide heuristics techniques in search space domain. While, most of the heuristic techniques are approaches to find optimal or near optimal solutions in a rational computational cost without a guarantee to find an optimal value. Meta-heuristic can be an effective ways to produce suitable solutions by trial and error to a difficult problem in an accepted time. The goal of the meta-heuristic procedures is to improve the efficiency of the heuristic algorithms. The difficulty of the problem of interest makes it impossible to search every possible solution. Therefore, it is required to find a good feasible solution in an acceptable processing period. In meta-heuristic techniques there are no assurance that the best solution can be found using these methods. Techniques which establish meta-heuristic algorithms range from simple local search procedures to complex learning processes [17]. They may include mechanisms to escape from getting trapped into local minima of the search space to a global optima. Meta-heuristic algorithms can be classified according to the number of optimal solutions found at the same time. A single solution methods called trajectory algorithms which are based on a single solution at any time. While, multi solution methods called population-based algorithms perform search with many initial points in a parallel style. Swarm intelligence algorithms represent an important population based meta-heuristic algorithms. These algorithms include simple particles or agents cooperating locally with each other depending on their environment. Each particle follows one or multiple number of simple rules that always modify the best obtained solution towards the optimum one without any centralized controlling in these agents performance. Accordingly, local and random interactions among these particles are directed to an intelligent global behavior. All meta-heuristic algorithms required a dynamic tradeoff between diversification and intensification, in other words using a certain tradeoff between local search and global exploration in a way that each particle improves its performance by cooperate with other agents, transferring information to other particles, and compete with other particles to survive [18].

4 EVOLUTIONARY ALGORITHMS FOR NEURAL NETWORKS WEIGHTS OPTIMIZATION

This paper emulate and compare the performance of ten evolutionary population based algorithms using MATLAB 2013 for training ANNs, some of these algorithms are very well-known and have been previously used for ANNs training, such as: PSO, and GA Other algorithms are recently proposed and rarely used with NNs weight optimization such as: SOS, and SFS. Other algorithms are programmed for training neural network for the first time in this paper such as: Chicken Swarm Optimization (CSO) [19], NBA, MVO, Moth-Flame Optimization (MFO) [20], and States of Matter search algorithm (SOM) [21]. All these algorithms are implemented and simulated using MATLAB. In general, we have considered the weights of ANNs are a vector in the population particle matrix, in which each particle represents a number of candidate solutions equal to the number of population to minimize the Mean Square Error (MSE) which represents the objective or cost function of the ANN. The number of weights will represent the problem dimension, initial weights are generated randomly, and the search space scope of the problem were bonded between minimum and maximum values. The general training pseudo code can be described as shown in table 1 below, while the corresponding training flow chart is shown in fig. 4.

TABLE 1

EVOLUTIONARY ALGORITHMS TRAINING ANNS PSEUDO CODE

```
1: Initialize algorithm general parameters.
2: For epoch=1:cycle
3: Randomly initialize population particles as a candidate solutions.
4:    For i=1:population size
5:    Evaluate each particle MSE.
6:    End for i
7: Evaluate the minimum MSE and indicate its corresponding particle
   global best.
8:    For iteration=1:maximum iteration
9:       For ii=1:population size
10:      Run the optimization algorithm on each individual particle
         and update them.
11:      Evaluate new min MSE and its corresponding new global
         best.
12:      If new min MSE < minimum MSE
13:      Update global best particle, and Minimum MSE
14:      End if
15:      If minimum MSE < error goal
16:      Save global best as the best solution, and the minimum MSE.
17:      Terminate the operation
18:      End if
19:      End for ii
20:   End for iteration
21:   Save the best solution obtained corresponding to the current epoch.
22: End for epoch.
```

For all used algorithms we have evaluate MSE using 30 particles population size, 10 epochs, 200 iterations for each epoch, hyperbolic tangent activation function for hidden layer, and log sigmoid for the output layer. Table 2 shows the examined algorithms parameters. This paper have considered two experiments for ANNs training using mentioned algorithms; the first experiment records the classification rate

of XOR problem with four clusters, 100 instances for each class, and three attributes, fig. 4 shows the XOR four opposite binary classes problem. In the second experiment we have classified iris real data set for the first case, which consist of 3 classes, 50 instances for each class, and 4 attributes, while the second case classified ecoli continuous data set with 8 classes, 336 instances, and 7 attributes. In both experiments we have train the ANN with the minimum required hidden neurons in the hidden layer to perform the higher classification rate. For both experiments we have divided the input data set patterns into two sets, the first subdivided data set consists of 90% of the total data set patterns which are used for training purpose, while the remaining 10% of data set patterns are used for testing the trained network. So that we could discover any over fitting in the training process. The first experiment required 6 boundary decisions, which means 6 hidden neurons, each neuron represents a boundary decision, while the second experiment we have used 3 hidden neurons for iris classification and 4 hidden neurons for ecoli data set. Recording the Average Classification Rate (ACR), Average Training Mean Square Error (ATRMSE), Average Testing Mean Square Error (ATEMSE), Average Error between MSE of Training and Testing data sets (AETT), and the Average Processing Time (APT) in seconds for both of the experiments.

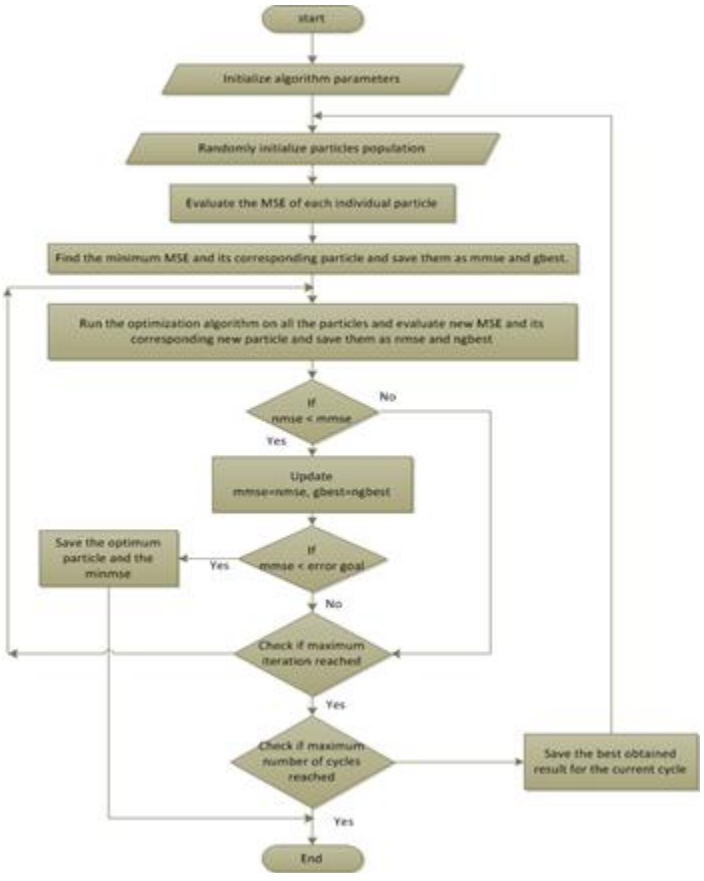

Fig. 3. *NNs general training flow chart with population based evolutionary algorithms.*

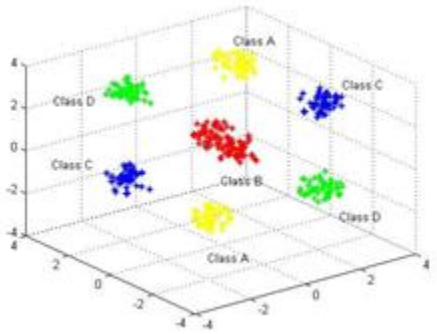

Fig. 4. *XOR four classes problem*

TABLE 2. *USED ALGORITHMS AND THEIR PARAMETERS.*

No.	Alg.	Algorithms parameters
1.	BSA	The frequency of birds flight behavior=10, cognitive accelerated coefficients c_1=c_2= 1.5, indirect and direct effect on the birds vigilance behavior=a_1=a_2= 1, population size= 30, maximum iterations= 200.
2.	CSO	Roosters percent= 0.15, hens percent= 0.7, mother hens Percent= 0.5, population size= 30, maximum iterations= 200.
3.	DEA	Mutation factor= 0.5, crossover rate= 0.9, population size= 30, maximum iterations= 200.
4.	GA	Mutation rate= 0.2, fraction of population= 0.5, crossover type single point, population size= 30, maximum iterations= 200.
5.	PSO	Upper bound of the search scope =maximum of search space range, lower bound of the search scope=minimum of search space range, acceleration coefficients c_1=c_2= 2. Inertia weight is time varying linear decreasing. Maximum and minimum inertia weight are 0.9 and 0.4, population size=30, maximum iterations= 200..
6.	MVO	Minimum wormhole existence probability= 0.2, maximum wormhole existence probability= 1, population size= 30, maximum iterations= 200.
7.	NBA	Maximal and minimal pulse rate are 1 and 0 respectively, maximal and minimal frequency are 1.5 and 0 respectively, maximal and minimal loudness are 2 and 1 respectively, gamma = 0.9, alpha = 0.99.The frequency of updating the loudness and pulse emission rate= 10,the maximal and minimal probability of habitat Selection are 0.9 and 0.6, The maximal and minimal compensation rate for Doppler effect in echoes are 0.9 and 0.1,the maximal and minimal contraction coefficient are 1 and 0.5,the maximal and minimal inertia weight are 0.9 and 0.5, population size= 30, maximum iterations= 200.
8.	SFS	Maximum diffusion = 2, walk=1, population size=30, maximum iterations=200.
9.	SOM	Gas phase= 1, movement = [0.9, 0.5, 0.1], collides = [0.3, 0.05, 0], random= [0.9, 0.2, 0], percent of phases = [0.5, 0.1,-0.1], adjust parameter = [0.85 0.35 0.1], population size=30, maximum iterations=200.
10.	SOS	Population size= 30, maximum iterations= 200.

5 EXPERIMENTS AND RESULTS

Table 3 shows the first experiment population based algorithms performance training NN consisting of 3 input, 6 hidden, 2 output neurons for classification of four clusters XOR problem, with 400 total input patterns divided into 360 training patterns and 40 testing patterns, Fig. 4 shows the A,B,C,D XOR clusters spreading in the search space, we have used tan sigmoid activation function for the hidden layer and the log sigmoid for the output layer.

TABLE 3. *ALGORITHMS PERFORMANCE 1ST EXPERIMENT*

Alg.	ATRMSE	ATEMSE	AETT	ACR%	APT(sec)
CSO	0.164	0.167	0.008	66.08	6.01
GA	0.093	0.093	0.003	94.22	4.49
MFO	0.085	0.086	0.005	95.75	3.79
MVO	0.174	0.177	0.003	55.97	5.17
NBA	0.167	0.169	0.004	63.00	3.47
PSO	0.173	0.172	0.004	60.67	3.39
SFS	0.092	0.089	0.003	93.36	18.92
SOM	**0.045**	**0.045**	**0.003**	97.58	3.44
SOS	0.073	0.076	0.004	**99.47**	12.94

From table 3 we have seen that SOS algorithm has gotten the highest ACR with 99.47% with relatively high processing time equal to 12.94 sec, while SOM algorithm has gotten the minimum ATRMSE, ATEMSE, and AETT with relatively low processing time 3.44 sec and high classification rate 97.47 which makes her the best training algorithm for classification XOR problem. Table 4 shows the 2nd experiment that classifies 150 input iris real data patterns, 135 for training and 15 for testing into three clusters, using a network of 4, 3, 2 input, hidden, output layer neurons respectively with tan sigmoid activation function for hidden layer, and log sigmoid for output layer.

TABLE 4. *ALGORITHMS PERFORMANCE FOR IRIS CLASSIFICATION*

Alg	ATRMSE	ATEMSE	AETT	ACR %	APT(sec)
CSO	0.096	0.102	0.013	66.37	4.19
GA	0.086	0.087	0.010	85.93	2.69
MFO	0.074	0.075	0.007	88.89	2.39
MVO	0.105	0.108	0.010	69.11	3.27
NBA	0.088	0.089	0.010	76.00	2.60
PSO	0.095	0.098	0.013	70.07	2.27
SFS	0.063	0.065	0.010	95.11	13.08
SOM	**0.034**	**0.035**	**0.010**	**96.74**	2.64
SOS	0.059	0.060	0.009	95.41	8.69

Table 4 has been shown that SOM has gotten the minimum ACR, ATRMSE, ATEMSE, and AETT with relatively low average processing time 2.64 sec. Table 5 stands for the classification of ecoli real data set which consists of 336 input patterns divided into 302 training patterns and 34 testing, to be classified into 8 classes using NN of 7 input, 4 hidden, and 3 output neurons with tan sigmoid activation function for hidden layer and log sigmoid for output layer. Table 5 shows that SOS has gotten the highest ACR, but with relatively high processing time, the lowest error between training and testing recorded according to MVO algorithm, SOM has gotten the lowest ATRMSE.

TABLE 5. *ALGORITHMS PERFORMANCE FOR ECOLI CLASSIFICATION.*

Alg	ATRMSE	ATEMSE	AETT	ACR %	APT(sec)
CSO	0.119	0.124	0.017	65.26	5.06
GA	0.094	0.102	0.018	74.40	3.26
MFO	0.090	0.097	0.014	74.47	2.95
MVO	0.123	0.124	**0.013**	64.87	4.80
NBA	0.101	0.108	0.014	70.53	3.26
PSO	0.111	0.114	0.017	68.71	2.83
SFS	0.089	0.094	0.014	75.83	16.61
SOM	**0.080**	0.095	0.022	78.28	3.10
SOS	0.081	**0.087**	0.015	**81.19**	11.55

Figures 5, 6, and 7 show the convergence curves of the applied algorithms for XOR, iris, and ecoli classification respectively.

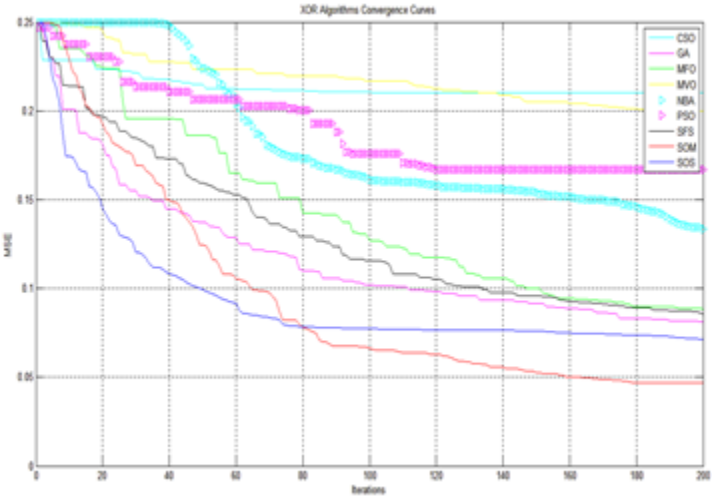

Fig.5. *XOR classification algorithms convergence curves.*

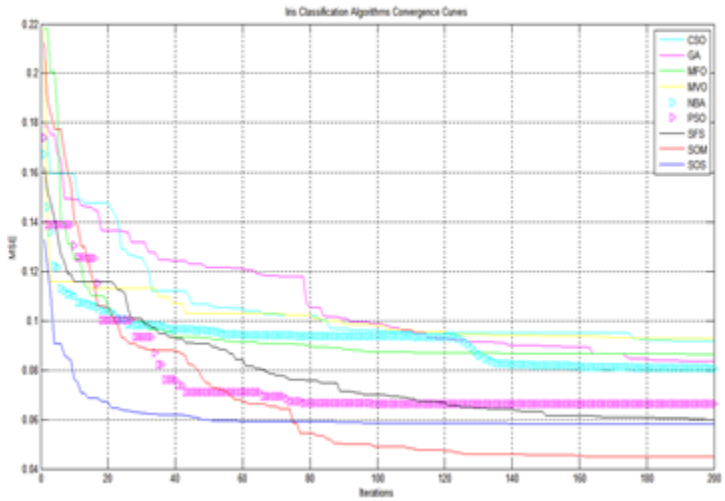

Fig. 6 *Iris classification algorithms convergence curves.*

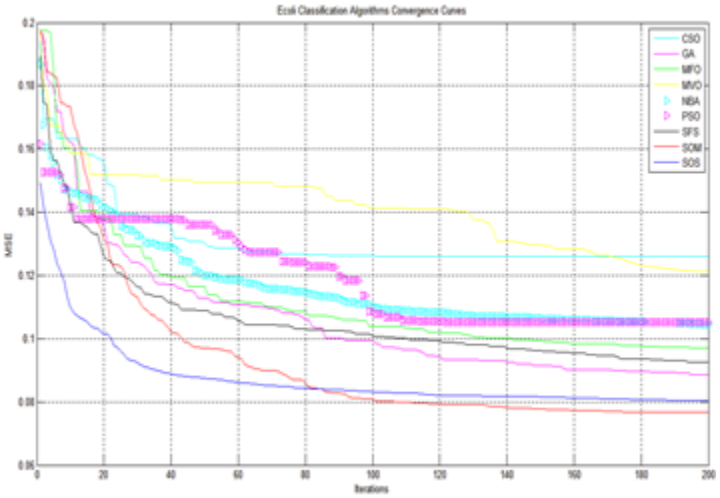

Fig.7. *Ecoi classification algorithms convergence curves.*

Figure 5, 6, 7 show that SOS algorithm has gotten the faster convergence curve to an acceptable solution, while SOM curve converge to the minimum MSE value at the end of iterations and for both of experiments.

6 CONCLUSION

This paper performs the classification of 4 clusters XOR, iris, and ecoli data sets using ANNs trained with recently proposed evolutionary population based algorithms compared with other classical algorithms like PSO and GA. The results have shown that the performance of these algorithms depend on the parameters of these algorithms and network structure, some algorithms have gotten highest classification rates but with relatively high processing time such as SFS and SOS algorithms, other algorithms have acceptable classification rates with low average processing time such as SOM. SOS algorithm has the faster convergence curve, while SOM has the lowest one. However, the obtained results shows that the performance of the new proposed algorithms is better than other classical ones.

REFERENCES

[1] G. P. Zhang, "Neural networks for classification: a survey," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 30, no. 4, pp. 451-462, Nov 2000. doi: 10.1109/5326.897072.

[2] M. Iqbal Quraishi, J. Pal Choudhury and M. De, "Image recognition and processing using Artificial Neural Network," Proc. IEEE, Recent Advances in Information Technology (RAIT), 2012 1st International Conference on, Dhanbad, 2012, pp. 95-100. 2012. doi: 10.1109/RAIT.2012.6194487.

[3] P. V. Rao, S. Madhusudana, N. S. S. and K. Keerthi, "Image Compression using Artificial Neural Networks," Proc. IEEE , Machine Learning and Computing (ICMLC), 2010 Second International Conference on, Bangalore, pp. 121-124, 2010. doi: 10.1109/ICMLC.2010.33

[4] S. Nirkhi, "Potential use of Artificial Neural Network in Data Mining," Proc. IEEE, Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on, Singapore,

pp. 339-343. 2010. doi: 10.1109/ICCAE.2010.5451537.

[5] D. Katic and M. Vukobratovic, "Intelligent control of humanoid robots using neural networks," Neural Network Applications in Electrical Engineering, 2004. NEUREL 2004. 2004 7th Seminar on, pp. 31-35, 2004. doi: 10.1109/NEUREL.2004.1416526M.

[6] M. N. H. Siddique and M. O. Tokhi, "Training neural networks: backpropagation vs. genetic algorithms," Proc. IEEE, Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on, Washington, DC, pp. 2673-2678 vol.4.. 2001. doi: 10.1109/IJCNN.2001.938792

[7] K. Bai and J. Xiong, "A Method of Improved BP Neural Algorithm Based on Simulated Annealing Algorithm," proc. IEEE, Genetic and Evolutionary Computing, 2009. WGEC '09. 3rd International Conference on, Guilin, pp. 765-768. 2009. doi: 10.1109/WGEC.2009.39

[8] A. Rakitianskaia and A. P. Engelbrecht, "Training neural networks with PSO in dynamic environments," IEEE Congress on Evolutionary Computation, Trondheim, pp. 667-673. 2009. doi: 10.1109/CEC.2009.4983009

[9] C. Ozturk and D. Karaboga, "Hybrid Artificial Bee Colony algorithm for neural network training," IEEE Congress of Evolutionary Computation (CEC), New Orleans, LA, pp. 84-88, 2011. doi: 10.1109/CEC.2011.5949602.

[10] A. Slowik and M. Bialko, "Training of artificial neural networks using differential evolution algorithm," IEEE Conference on Human System Interactions, Krakow, pp. 60-65, 2008. doi: 10.1109/HSI.2008.4581409

[11] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization", Neural Computing and Applications, vol. 27, no.2, pp. 495-513, February 2016.

[12] M. Cheng, and D. Prayogo, " Symbiotic Organisms Search: A new metaheuristic optimization algorithm", Elsevier Ltd. Computers & Structures, vol. 139, pp. 98–112, 2014.

[13] H. Salimi, "Stochastic Fractal Search: A powerful metaheuristic algorithm", Elsevier Ltd., Knowledge-Based Systems, vol. 75, pp. 1–18, 2015.

[14] X. Menga, X.Z. Gaob, Y. Liuc, and H. Zhanga, "A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization" ,Elsevier Ltd., Expert Systems with Applications, vol. 42, no. 17–18, pp. 6350–6364, 2015.

[15] X. Menga, X.Z. Gao, L. Lu, Y. Liu, and H. Zhang, “A new bio-inspired optimization algorithm: Bird Swarm Algorithm,” Experimental & Theoretical Artificial Intelligence, Mar. 2015.