Comparative Analysis of Swarm Intelligence Techniques for Data Classification

A. Ashray Bhandare and B. Devinder Kaur
Department of EECS, The University of Toledo, Toledo, Ohio, USA

Abstract - *This paper investigates the effectiveness of employing two relatively new swarm intelligence (SI) metaheuristic techniques in determining the accuracy of data classification problem. The SI metaheuristics analyzed are Grey wolf optimizer and firefly algorithm (FA). In this work, Grey wolf optimizer and firefly algorithm (FA) are used in training feed-forward neural networks (FNN) for the purpose of data classification. In experiments, the iris data has been used to evaluate the performance of the proposed algorithms. The experimental results obtained from these techniques are compared with that of a similar population-based technique, namely, particle swarm optimizer (PSO). Results obtained show that both GWO and Firefly algorithms provide superior solutions for the case studied.*

Keywords: Grey Wolf Optimizer, Firefly Algorithm, Particle Swarm optimization.

1 Introduction

Artificial Neural Networks (ANN) consist of processing elements neurons in multiple layers and are connected with each other through the links. Each link is represented by a weight which represents the strength of the connection. These networks can learn an arbitrary vector mapping from the input space to the output space. These neurons process information based on the net sum function and activation function. In order to find the appropriate representation of neural network for a given problem, it is essential to find the best architecture of the neural network and the algorithm to train the network. Neural networks have been used for signal processing, pattern recognition, and control, etc.

The optimization problem for artificial neural network is to design a suitable network structure and find the optimal weights connecting successive layers of neurons and find the threshold of each neuron.

The structure of the neural network chosen for classification of iris data is shown in Figure 1. It has four neurons in the input layer, nine neurons in the hidden layer and three neurons in the output layer. The number of neurons in the input layer are defined by the input parameters in the data set and output neurons are determined by the categories of outputs. It has four inputs neurons and three output neurons. The number of neurons in the hidden layer are designer's choice.

After determining the structure of the neural network, it is essential to develop the algorithm to find the optimal set of weights and biases. Several algorithms have been developed for neural networks by different researchers. The backpropagation algorithm is one of the most popular algorithm, it can find a set of weights in a reasonable amount of time [10]. Backpropagation is a variation of gradient search and the key to backpropagation is a method for calculating the gradient of the error with respect to the weights for a given input by propagating error backwards through the network. However, some drawbacks of backpropagation are that it can get stuck in local minima and are computationally complex [10].

Many global optimization methods have been proposed for training FNN in order to overcome the disadvantages of gradient based algorithms. Some nature inspired algorithms such as genetic algorithm (GA), particle swarm optimization (PSO) algorithm and artificial bee colony (ABC) algorithm have been successfully applied for training the Feed Forward Network (FNN) [8], [11], [5], [3], [4].

In this paper, Swarm Intelligence based two novel algorithms have been used to train the neural networks. They are Grey Wolf Optimizer (GWO) and Firefly Algorithm (FA). The GWO algorithm mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. The FA is based on the idealized behavior of the flashing characteristics of fireflies [12]. The results obtained from both these algorithms are compared with that of Particle Swarm Optimization (PSO) algorithm. The GWO is compared with FA to find which is better between the two.

This paper is organized as follows. In Section 2, the dataset used for this analysis is introduced. Section 3 presents the mathematical model of FNN. The GWO and FA are explained in section 4 and section 5 respectively. Section 6 talks about our proposed method to optimize the FNN with the SI techniques. In Section 7, experiments and results are presented. Conclusions are presented in section 8.

2 Iris Dataset

The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by Ronald Fisher in his 1936 paper [2]. It is retrieved from the University of California at Irvine (UCI) Machine Learning Repository [1].

The data set consists of 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Each sample consists of four features: petal width, sepal width, petal length, sepal length.

This data set is considered as a typical test case for many statistical classification techniques in machine learning. In our case, we will classify this data set using the Swarm Intelligence (SI) techniques mentioned above.

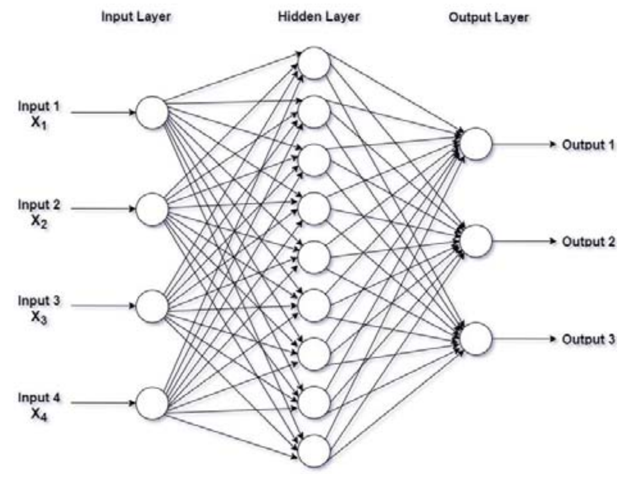

Figure 1: Neural Network Structure

3 Feed-Forward Neural Network

Feed-forward neural networks (FNN) are one-directional neural networks with one-way connections between their neurons. They consist of neurons arranged in different parallel layers. The first layer is always called the input layer, whereas the last layer is called the output layers. The layers in between the input and the output are called the hidden layers. After providing the inputs, weights, and biases, the outputs of this network are calculated using the following steps: [7]

1) The weighted sums of inputs are first calculated by Equation (1)

$$s_j = \sum_{i=1}^{n} \left(W_{ij} \,.\, x_i\right) - \theta_j, \qquad j = 1,2,\ldots.h \tag{1}$$

Where n is the number of the input nodes, W_{ij} shows the connection weight from the i^{th} node in the input layer to the j^{th} node in the hidden layer, θ_j is the bias (threshold) of the j^{th} hidden node, and $\mathbf{X}_i$ indicates the i^{th} input

2) The output of each hidden node is calculated using Equation (2)

$$H_j = sigmoid\left(s_j\right) = \frac{1}{\left(1 + e^{-s_j}\right)}, \quad j = 1,2,\ldots.h \tag{2}$$

3) The final outputs are defined based on the calculated outputs of the hidden nodes using equations (3) and (4)

$$y_k = \sum_{j=1}^{n} (W_{jk}.H_i) - \theta'_k, \quad k = 1,2,\ldots.m \tag{3}$$

$$Y_j = sigmoid(y_k) = \frac{1}{\left(1 + e^{-y_k}\right)}, \quad k = 1,2,\ldots.m \tag{4}$$

Where w_{jk} is the connection weight from the j^{th} hidden node to the k^{th} output node, and θ'_k is the bias (threshold) of the k^{th} output node.

4 Grey Wolf Optimizer

Grey wolf (Canis lupus) belongs to Canidae family. Grey wolves are at the top of the food chain and are considered as apex predators. Grey wolves mostly prefer to live in a pack. The leaders are called alphas and they can be either a male or a female. The alpha is mostly responsible for making decisions about hunting, sleeping place, time to wake, and so on. The second level in the hierarchy of grey wolves is the beta. The betas help the alpha in decision-making or other pack activities. They act as the subordinate wolves. The lowest ranking grey wolf are the omegas. The omega wolves always have to submit to all the other dominant wolves. The wolves which are not alphas, betas or omega are called deltas. The delta wolves submit to the alpha and betas but dominate the omegas. [6]

In addition to the social hierarchy of wolves, group hunting is another interesting social behavior of grey wolves. The main phases of grey wolf hunting are as follows: [9]

- Tracking, chasing, and approaching the prey
- Pursuing, encircling, and harassing the prey until it stops moving
- Attack the prey

When designing the Grey wolf optimizer, we mathematically model the social hierarchy of wolves by considering the fittest solution as the alpha (α). Accordingly, the second and third best solutions are named beta (β) and delta (δ) respectively. The rest of the candidate solutions are called omega (ω). In the GWO algorithm the hunting (optimization) is guided by α, β, and δ. The ω wolves follow these three wolves.

As mentioned above, grey wolves encircle their prey during the hunt. In order to mathematically model encircling behavior Equations (5) and (6) are proposed. [6].

$$\vec{D} = \left|\vec{C}.\overrightarrow{X_p}(t) - \vec{X}(t)\right| \tag{5}$$

$$\vec{X}(t+1) = \overrightarrow{x_p} - \vec{A}.\vec{D} \tag{6}$$

Where t indicates the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors, $\overrightarrow{X_p}$ is the position vector of the prey, and $\vec{X}$ indicates the position vector of a grey wolf. $\vec{A}$ and $\vec{C}$ are given by Equations (7) and (8)

$$\vec{A} = 2.\vec{a}.\overrightarrow{r_1} - \vec{a} \tag{7}$$

$$\vec{C} = 2.\overrightarrow{r_2} \tag{8}$$

Where components of $\vec{a}$ are linearly decreased from 2 to 0 over the course of iterations and r_1, r_2 are random vectors in the interval [0, 1]. As mentioned above the grey wolves finish the hunt by attacking the prey when it stops moving. In order to mathematically model this behavior of approaching the prey, the value of $\vec{a}$ is decreased in each iteration. Note that the fluctuation range of A is also decreased by $\vec{a}$.

By using the above mentioned equations, a grey wolf updates its position inside the space around the prey in any random location. Grey wolves can recognize the location of prey and encircle it. The hunt is usually guided by the alpha. A new beta and delta emerge in each iteration as all the other wolves update their positions. We assume that the alpha (best candidate solution) beta, and delta have better knowledge about the potential location of prey. This information is used in order to mathematically simulate the hunting behavior of grey wolves. Therefore, we save the first three best solutions obtained so far and update the positions of the other search agents (including the omegas) according to the position of the best search agent. The following Equations (9) to (15) are proposed in this regard. [6].

$$\overrightarrow{D_\alpha} = |\overrightarrow{C_1}.\overrightarrow{X_\alpha} - \vec{X}| \tag{9}$$

$$\overrightarrow{D_\beta} = |\overrightarrow{C_2}.\overrightarrow{X_\beta} - \vec{X}| \tag{10}$$

$$\overrightarrow{D_\gamma} = |\overrightarrow{C_3}.\overrightarrow{X_\delta} - \vec{X}| \tag{11}$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}.(\overrightarrow{D_\alpha}) \tag{12}$$

$$\overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2}.(\overrightarrow{D_\beta}) \tag{13}$$

$$\overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3}.(\overrightarrow{D_\delta}) \tag{14}$$

$$\vec{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \tag{15}$$

The pseudo code for the Grey Wolf Optimizer is shown in algorithm 1.

Algorithm 1 GWO Algorithm

Input: $f(\mathbf{x})$, $\mathbf{x} = (x_1, x_2, ..., x_d)$
Initialisation:
Generate an initial population of grey wolves $\mathbf{x}_i (i = 1, 2, \dots, n)$.
Initialize a, A, and C.
Calculate the fitness of each search agent.
$\mathbf{X}_\alpha$ =the best search agent
$\mathbf{X}_\beta$ =the second best search agent
$\mathbf{X}_\delta$ =the third best search agent
LOOP Process
while t < Max Generation **do**
  for i = 1 : n (all n search agent) **do**
    Update the position of the current search agent by above equations
  end for
  Update a, A, and C.
  Calculate the fitness of all search agents
  Update $\mathbf{X}_\alpha$, $\mathbf{X}_\beta$, and $\mathbf{X}_\delta$
end while
return $\mathbf{X}_\alpha$

5 Firefly Algorithm

The firefly algorithm (FA) is a meta-heuristic algorithm, inspired by the flashing behavior of fireflies. A Firefly's flash acts as a signal system and attracts other fireflies. This is its primary purpose. Flashing characteristics of fireflies can be summarized by the following three rules: [12].

- All fireflies are unisex so that one firefly will be attracted to other fireflies regardless of their sex.
- Attractiveness is proportional to firefly brightness. For any couple of flashing fireflies, the less bright one will move towards the brighter one. The brightness decreases when the distance between fireflies is increased. The brightest firefly moves randomly, because there is no other bug to attract it.
- The brightness of a firefly is affected or determined by the landscape of the objective function to be optimized.

Fireflies are attracted towards the brighter fireflies, when firefly j is more attractive than firefly i , then firefly i moves towards j by step changes given in Equation (16).

$$X_i^{t+1} = X_i^t + \beta_0 . e^{[-\gamma r_{ij}^2]} . (X_j^t - X_i^t) + \alpha_t . \epsilon_t \tag{16}$$

Where β_0 is the initial attractiveness of the firefly, α_t is a parameter controlling the step size, while ϵ_t is a vector drawn from a Gaussian or other distribution function. Distance r_{ij} between fireflies i and j is obtained by Cartesian distance formula given by Equation (17).

$$r_{ij} = \sqrt{\sum_{k=1}^{D} (X_{i,k} - X_{j,k})^2} \quad (17)$$

The pseudo code for the Firefly Algorithm is shown in algorithm 2.

Algorithm 2 Firefly Algorithm

```
Input: f(x), x = (x_1, x_2, ..., x_d)
  Initialisation:
  Generate an initial population of fireflies x_i (i = 1, 2, ..., n).
  Formulate light intensity I so that it is associated with f(x).
  Define absorption coefficient γ.
  LOOP Process
  while t < Max Generation do
    for i = 1 : n (all n fireflies) do
      for j = 1 : n (n fireflies) do
        if (I_j > I_i) then
          move firefly i towards j.
        end if
        Vary attractiveness with distance r via exp(−γ r) ;
        Evaluate new solutions and update light intensity.
      end for
    end for
    Rank fireflies and find the current best.
  end while
  return current Best
  Post-processing the results and visualization.
```

6 Proposed Method

When training a Feed-Forward Network (FNN) using meta-heuristics, the most crucial step is the problem representation. In other words, the training problem should be formulated in such a way that it is suitable for meta-heuristics. [7]. The variables during the training of a neural network are the weights and biases. A trainer should be able to find an optimum set of values for weights and biases such that it provides the highest classification and prediction accuracy. Since the GWO and FA algorithm accepts the variables in the form of a vector, the variables for these algorithms are presented in Equation (18).

$$\vec{V} = \{\vec{W}, \vec{\theta}\} = \{W_{1,1}, W_{1,2}, \ldots, W_{n,n}, \theta_1, \theta_2, \ldots, \theta_j\} \quad (18)$$

Where n is the number of the nodes, W_{ij} shows the connection weight from the i^{th} node to the j^{th} node, θ_j is the bias (threshold) of the hidden and output nodes.

After defining the variables, we need to define the objective function for the SI algorithms. The Mean Square Error (MSE) is the most used parameter for the evaluation of a neural network. Here, the difference between the desirable output and the actual output that is obtained from the feed-forward equations are stored. The performance of the network is evaluated based on the average MSE of all the training samples as computed by Equation (19)

$$MSE = \sum_{k=1}^{s} \frac{\sum_{k=1}^{m} (y_i^k - d_i^k)^2}{s} \quad (19)$$

Where s is the number of training samples, m is the number of outputs, d_i^k is the desired output of the i^{th} input unit when the k^{th} training sample is used, and y_i^k is the actual output of the i^{th} input unit when the k^{th} training sample appears in the input. Now the problem becomes a minimization problem with the average MSE as the variable to be minimized.

During each iteration, the weights and biases of neural network are updated based on the best solutions from the previous iteration, therefore, as the neural networks evolve, the MSE gradually decreases. If the SI algorithms are iterated over sufficient number of iterations, then the best solution can be obtained.

7 Evaluation

In this section the proposed SI techniques based trainer is used to obtain the optimum classification for the Iris data set. The parameters of the iris dataset are listed in Table 1.

Table 1: Parameters of the Iris Dataset

Parameters	values
Number of attributes	4
Number of training samples	150
Number of testing samples	150 as training samples
Number of classes	3
FNN structure	4-9-3

7.1 Experimental Setup

The Grey Wolf Optimizer and the Firefly algorithm have been implemented in MATLAB R2015b. Tests were carried on a PC with Intel(R) Core(TM) i7-4790 processor @2.10 GHz with 16GB of RAM and Windows 7x64 Professional operating system. In the experiments, the results obtained by the SI techniques were generated and recorded. They were repeated 20 times and training processes were stopped when the iterations reached a count of 100.

The results were also compared with PSO to verify that GWO and FA out performs the conventional swarm based PSO. The training process was initiated by generating random weights and biases in the range of [-20, 20] for the iris data set. Other initial parameters for the PSO, FA and GWO are shown in Table 2.

Table 2: The Initial Parameters of Algorithms

Algorithm	Parameter	Description
PSO	Topology	Fully Connected
	Cognitive Constant(C_1)	1
	Social Constant(C_2)	1
	Inertia Constant(ω)	0.3
	Population size	40
	Maximum number of iterations	100
FA	Mutation Coefficient(α)	0.2
	Attraction Coefficient Base Value(β_0)	2
	Light Absorption Coefficient(γ)	1
	Population size	30
	Maximum number of iterations	100
GWO	$\vec{a}$	Linearly decreased from 2 to 0
	Population size	150
	Maximum number of iterations	100

7.2 Analysis of Results Obtained

In our experiments, the neural network architecture has four inputs, three outputs and nine hidden neurons. Therefore, it has 75 parameters consisting of weights and biases as shown in Equation (20).

$$\begin{aligned} &No. of\ Variables \\ &= (input\ neurons \\ &+ output\ neurons)\ hidden\ neurons \\ &+ (hidden\ neurons\ +\ output\ neurons) \end{aligned} \tag{20}$$

The results of the training algorithms are presented in Table 3.

Table 3: Experimental Results after 10 Runs for PSO, FA and GWO algorithms

Algorithm	MSE (avg)	MSE (Best)	Classification Rate	Time (Seconds)
PSO	0.23149	0.0491	78.60%	182.9
FA	0.0124	0.0027	99.33%	521.5
GWO	0.01585	0.0051	99.33%	178.5

Table 3 shows that the GWO and FA outperform PSO by achieving lower MSE and better classification accuracy.

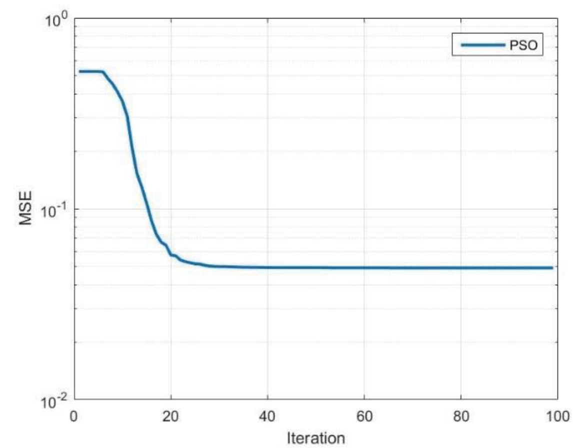

Figure 2: Best cost vs iteration number for Particle Swarm Optimization algorithm

PSO was not able to find the optimal solutions for the iris data. Figure 2 depicts the training performance of the Particle data in 100 iterations with an average classification rate of only 78%.

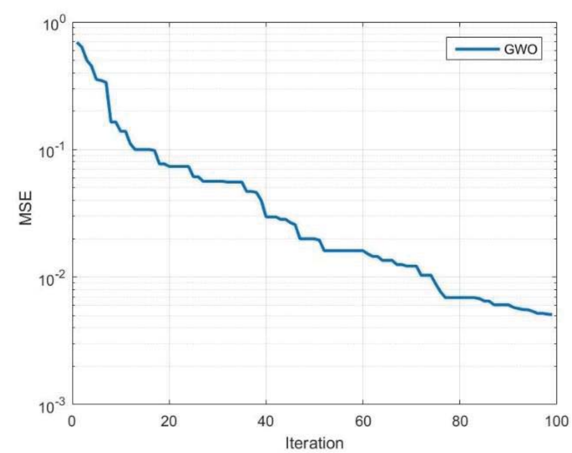

Figure 3: Best cost vs iteration number for Grey Wolf Optimizer algorithm.

Figure 3 shows the best MSE achieved with the GWO algorithm. Figure 4 shows the best MSE achieved with the Firefly Algorithm.

Both GWO and FA algorithms achieved an average classification rate of 99.33%. The results testify that these algorithms have superior local optima avoidance and achieve better accuracy.

The time taken by FA to complete 100 iterations is way higher than GWO. As shown in Table 2, the population size considered for GWO was way higher than the population size considered for FA and yet the overall execution time of the GWO is far less than FA. These results prove that GWO is

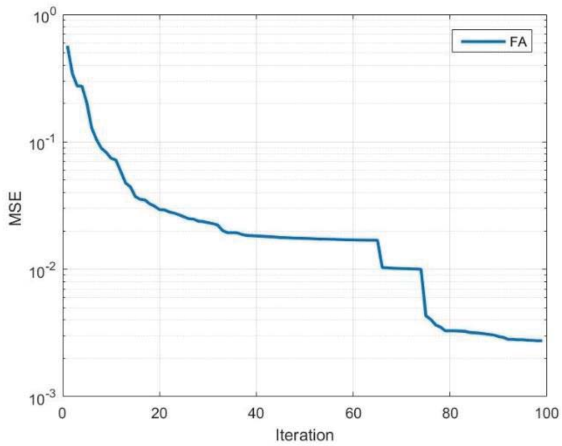

Figure 4: Best cost vs iteration number for Firefly algorithm

suitable to tackle complex problems with larger population size and still solve them in lesser time.

8 Conclusion

In this paper, we carried out a comparative study of the Firefly Algorithm, Grey Wolf Optimizer algorithm and Particle Swarm Optimization. Firefly and GWO algorithms are variants of Swarm intelligence. Both these algorithms were used to train artificial neural networks for classification of Iris dataset. The results from these algorithms were compared to the particle swarm optimization algorithm.

Our simulation results for finding the optimal values of weights and biases for the classification of the iris data set suggest that the Grey wolf optimizer and the firefly algorithm outperform the Particle Swarm Optimization (PSO) in terms of shorter run time and better classification rate. The success rate of 99.33% was achieved in just 100 iterations. However, it was found that the Grey Wolf Optimizer finds the optimal weights and biases of the neural network in nearly one third the time required by the firefly algorithm.

Therefore, it can be concluded that swarm based GWO and FA have the potential of solving a large number of NP hard problems in the Engineering domain.

9 References

[1] Blake C, Merz CJ (1998) UCI Repository of machine learning databases.

[2] R. A. Fisher (1936)." The use of multiple measurements in taxonomic problems". Annals of Eugenics 7 (2): 179188. doi:10.1111/j.14691809. 1936.tb02137.x.

[3] Karaboga D., Akay B., Ozturk C., Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks, LNCS: Modeling Decisions for Artificial Intelligence, Volume 4617, 2007, pp.318-329.

[4] D. Karaboga, C. Ozturk, Neural Networks Training by Artificial Bee Colony Algorithm on Pattern Classification, Neural Network World, Volume 19, Number 3, 2009, pp.279-292.

[5] Mendes, R., Cortez, P., Rocha, M., Neves, J., Particle swarm for feedforward neural network training. In: Proceedings of the International Joint Conference on Neural Networks, Vol. 2, 2002, pp. 18951899.

[6] S. Mirjalili, S. M. Mirjalili, and A. Lewis," Grey Wolf Optimizer," Advances in Engineering Software, vol. 69, pp. 46-61, 2014.

[7] Seyedali Mirjalili." How effective is the Grey Wolf optimizer in training multi-layer perceptrons." Applied Intelligence (2015): 1-12.

[8] D. J. Montana and L. Davis, Training Feedforward Neural Networks Using Genetic Algorithms, Proceedings of the International Joint Conference on Artificial Intelligence, 1989, pp. 762-767.

[9] Muro C, Escobedo R, Spector L, Coppinger R. Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations. Behav Process 2011;88:192–7.

[10] Rumelhart, D.E., Williams, R.J., Hinton, G.E.,Learning internal representations by error propagation, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1, 1986, pp. 318362.

[11] Tsai, J.T., Chou, J.H., Liu, T.K.: Tuning the Structure and Parameters of a Neural Network by Using Hybrid Taguchi-Genetic Algorithm. IEEE Transactions on Neural Networks, Volume 17, Issue 1, 2006, pp.69-80.

[12] X.S.Yang, Firefly algorithms for multimodal optimization, in: Stochastic Algorithms: Foundations and Applications, SAGA 2009, Lecture Notes in Computer Sciences, Vol.5792, 2009, pp. 169-178.