ARTIFICIAL FISH SWARM OPTMIZATION FOR MULTILAYER NETWORK LEARNING IN CLASSIFICATION PROBLEMS

Shafaatunnur Hasan[1], Tan Swee Quo and Siti Mariyam Shamsuddin[2]

Soft Computing Research Group (SCRG)
Universiti Teknologi Malaysia
Skudai, Johor

shafaatunnur@gmail.com[1]
mariyam@utm.my[2]

ABSTRACT

Nature-Inspired Computing (NIC) has always been a promising tool to enhance neural network learning. Artificial Fish Swarm Algorithm (AFSA) as one of the NIC methods is widely used for optimizing the global searching of ANN. In this study, we applied the AFSA method to improve the Multilayer Perceptron (MLP) learning for promising accuracy in various classification problems. The parameters of AFSA: AFSA prey, AFSA swarm and AFSA follow are implemented on the MLP network for improving the accuracy of various classification datasets from UCI machine learning. The results are compared to other NIC methods, i.e., Particle Swarm Optimization (PSO) and Differential Evolution (DE), in which AFSA gives better accuracy with feasible performance for all datasets.

Keywords: Artificial neural network, artificial fish swarm algorithm, classification problems.

INTRODUCTION

The main implication of an Artificial Neural Network (ANN) is learning and improving through an environment where a machine learning technique enables it to learn from experience, generalize from knowledge, perform abstraction, make errors, and it does not need to be reprogrammed. According to Shamsuddin, *et al.*, (2001), the most common algorithm used in ANN is Backpropagation (BP). However, BP algorithm is always trapped in local minima and has a slow convergence rate. Due to the weaknesses, Genetic Algorithm (GA) has been introduced to improve BP network learning. Still, GA has complex functions and it is more time consuming in producing output. Subsequently, a Swarm Intelligence (SI) technique called Particle Swarm Optimization (PSO) has become a popular method because of its intuitiveness and ease of implementation. It is also more effective in solving non-linear optimization problems. Another technique, Differential Evolution (DE) has spontaneous self adaptability, diversity control and continuous improvements. Artificial Fish Algorithm (AFSA) is one of the SI techniques with many good properties; it is insensitive to initial values, flexible in practice, precise in optimization, and has strong robustness, rapid search in global optimum, tolerance of parameter setting and searching adaptation. AFSA is less likely to get stuck in local minimum. Compared to DE, AFSA can perform easily with less iteration and require less adjustment of parameters because it does not possess the mutation and crossover processes. Furthermore, the basic DE algorithm is unsteady because the individuals in size of population vector and target vector are randomly generated and selected during the period of evolutionary procedures. AFSA shares many common points with PSO. Both optimization algorithms started with population of random solutions, and fitness values is evaluated to the population. However, PSO algorithm has disadvantages such as being premature, has low precision, slow convergence and parameter selection problems.

The ANN has offered solutions to many problems, and has been successfully applied in many applications, especially in intelligence system, prediction and classification. In 2002, Wyeth, Busky and Robert (2000), proposed an intelligent flight control system using an Artificial Neural Network (ANN). The ANN is used to generate hover commands, which are used to manipulate the flight directly. The system was successfully implemented in the Helicopter Unmanned Ariel Vehicle (UAV), at the University of Queensland and CSIRO Manufacturing Science & Technology.

Profitability analysis in the Malaysian market was introduced by Shamsuddin, (2004). An improved Backpropagation NN error function has been successfully implemented for predicting profitability of selected firms at Kuala Lumpur Stock Exchange (KLSE). Subsequently, Limsombunchai, Clemes dan Weng (2000) used consumer choice prediction to understand and predict consumer decisions accurately. The proposed method is more effective in targeting the products (and/or services), has more cost effectiveness in marketing strategies, leading to an increased in sales and resulting in a substantial improvement in overall profitability of the firm. In solving classification problems, there were many studies carried out such as, multilevel Kohonen network learning by Shamsuddin, Zainal dan Yusof, Sobri dan Chiu (2008), and writer identification for Chinese handwriting by Wong and Shamsuddin, (2010). More recently, Kuok *et al.*, (2011) proposed the hourly runoff forecast at different lead-time for a small watershed using ANN. The authors adopted the MLP and Recurrent networks to forecast the hourly runoff of Sungai Bedup Basin, Sarawak. Both networks have been implemented successfully with higher accuracy and can be utilized as early warning flow forecaster.

Over the years, there have been significant research efforts to apply evolutionary computation (EC) techniques for the purposes of evolving one or more aspects of artificial neural networks. EC methodologies have been applied to three main attributes of neural networks; network connection weights, network architecture (network topology, transfer function), and network learning algorithms. The idea of ANN learning with Swarm Intelligence (SI) was proposed by Shi and Eberhart (1998) with a modified particle swarm optimizer. A new parameter called inertia weight has been introduced. The inertia weight in the range of 0.9 to 1.2 on average will have a better performance, and has a bigger chance to find the global optimum within a reasonable number of iterations. Many researchers have also worked on the integration of ANN with EC. For instance, Haza (2006), used a PSO for NN learning enhancement, and Firdaus (2007), used the Elman recurrent network learning with PSO. Both authors had successfully trained the NN with PSO for classification problems. In 2011, Hasan and Shamsuddin proposed the Multi-Strategy Self Organizing Map Learning for Classification Problems. The authors used PSO to optimize the weights from an improved SOM's hexagonal lattice structure. The proposed method gives a promising result compared to standard SOM.

Recently, an intelligent universal mechanism, referred to as AFSA was proposed to deal with function approximation, pattern recognition, process estimation and prediction, optimization design and other applications. Jiang, Wang, Rubio dan Yuan (2007), used a new estimation method, called the Spread Spectrum Code Estimation. The SI approach is to recover the transmitted data bits and code of spread spectrum signal over additive white Gaussian noise channel. The estimation method by AFSA is insensitive to initial values, has a strong robustness, and has a faster convergence speed and better estimation precision compared to the estimation method by GA and PSO. The result shows that the method can obtain the optimal or sub-optimal estimation of spreading code, even when the signal power is below the noise power. Subsequently, Chen, Wang, Li dan Li (2007) proposed a hybrid of AFSA and PSO for feedforward neural network training. The hybridization of AFSA and PSO has not only the artificial fish behaviours of swarm and follow, but also takes advantage of the information of the particle.

Furthermore, a new tool for data mining to discover classification rules from data, called the AF-Miner was developed by Zhang, Shao, Li and Sun (2006). Mining classification rule task is formalized into an optimization problem.

Each potential IF-THEN rule is encoded into a real-valued AF that contains the upper and lower limits of all attributes in data sets. The simulation results show that AF-Miner can mine better classification rule, including rule set with higher predictive accuracy rate, better generalization ability, and the smaller number of rules, with simpler rules with fewer terms.

However, an AFSA for Estimating Parameters in Production Function was introduced by Li, Wang and Zhou (2009). The common Cobb-Douglas production function is a multivariable non-linear function; and most of the traditional methods have some limitations in estimating parameters. The parameters of production function are constructed into the AF and estimated by the random behavior, preying behavior, swarming behavior and following behavior. The residual sum of squares is designed into the function of food consistence. The simulation results show that this algorithm has fast optimization speed and possesses minimum regression residual sum of squares. In addition, Cao, Mao, Zheng, Jiang dan Zhang (2009), proposed an AFSA-BP Network Speed Identifier and Application in Direct Torque Control System. They applied Backpropagation (BP) neural network to design a speed identifier that has a speed-sensorless direct torque control (DTC) system. The result indicates that the combination BP algorithm and AFSA are easy to pledge into local solution since AFSA has global search ability. Recently, Peng (2011), proposed an improved AFSA, the so called IAFSA for optimal operation of cascade reservoirs. IAFSA was proposed due to the poor vision of AFSA in searching and its capability to keep the balance of exploration and exploitation. The results show that IAFSA has greater improvement than standard AFSA. The concept of Artificial Fish Algorithm (AFSA) is explained in the next section.

ARTIFICIAL FISH ALGORITHM (AFSA)

AFSA is a population-based EC technique that is inspired by the natural social behavior of fish schooling and SI. AFSA, with its Artificial Fish (AF) concept was proposed by Dr. Li Xiao-Lei in 2002. The simulations on an object-oriented analytical method allow AF to adopt information by sense organs and perform stimulant reaction by controlling the tail and fin. The AFSA is a robust stochastic technique in solving optimization problems based on the movement and intelligence of swarms in the food finding process. The three main principles developed in AFSA are the fish behaviors in food searching, swarming and following.

In this study, we assume that, AF_Prey is food searching, AF_Swarm is a group of fish and AF_Follow is where the swarm of fish is tracking to the food location. AFSA constructs simple basic behaviors of AF, based on local searching behaviors to reach the global optimum. AFSA basic procedures are given in Figure 1. According to Chen et al., (2007), AF is evaluated by the current environment and would select a suitable behavior with better improvement state. Let i^th^ AF represent a D-dimensional vector $X_i = (x_1, x_{2, \dots,} x_D)$ where i= 1,2, … n and $X_j = (x_1^j, x_{2,\dots,}^j x_n^j)$, as random selected states within visual position of X_i.

Assume,

$$y = f(x) \tag{1}$$

$$d_{ij} = \| x_i - x_j \| \tag{2}$$

$$x_j - x_i = + \text{Visual.Rand ()}, i \ \ (O, n] \tag{3}$$

$$S = \{X_f | \; \|X_i - X_j\| < \text{Visual}\} \tag{4}$$

$$X_{next} = X_i + \frac{x - x_i}{\|x - x_i\|}. \text{ Step.Rand ()} \tag{5}$$

where,

NUM	=	the maximum iteration
y	=	fitness function at position x (represent food concentration, FC)
d_{ij}	=	distance between the AF i and j
Visual	=	visual distance
Rand ()	=	function produces random numbers between 0 and 1
Step	=	moving step length
S	=	set of AF exploring area at present position (neighborhood)
x	=	x_j for prey, x_c for swarm *or* x_{max} for follow behavior
x_i	=	optimizing variables
n	=	total number of AF / swarm size
n_f	=	number of its companions fellow in the current neighborhood S
δ	=	crowd factor (0 < δ < 1)
try_num	=	maximum number of chances that x_j being randomly chosen

Figure 1 illustrates the procedures of AFSA. Initially, all parameters are set to VISUAL, Step, δ, *nf*, NUM, and N, stopping condition, iteration counter and get fitness value for every AF. In the population of *N*, each of their action function values is evaluated. If the function value criterion is met, AF_swarm and AF_follow behavior are executed; otherwise AF_prey behavior is implemented by default. Perhaps or possibly not a companion in AF vision, prey behavior is executed and the callboard will be updated with its position and FC; once no companion AF is around it. Otherwise, both swarm and follow behavior with defaults prey behavior will be executed and behavior with higher FC is chosen for updating process. Afterwards, the call board on AF with best food concentration (bestFC) is updated. As the stopping condition is met, the process is ended with output result of classification percentage and convergence rate. Besides, the process continues to the next iteration. Furthermore, the methodology of Artificial Fish Algorithm (AFSA) in MLP Learning is discussed in the next section.

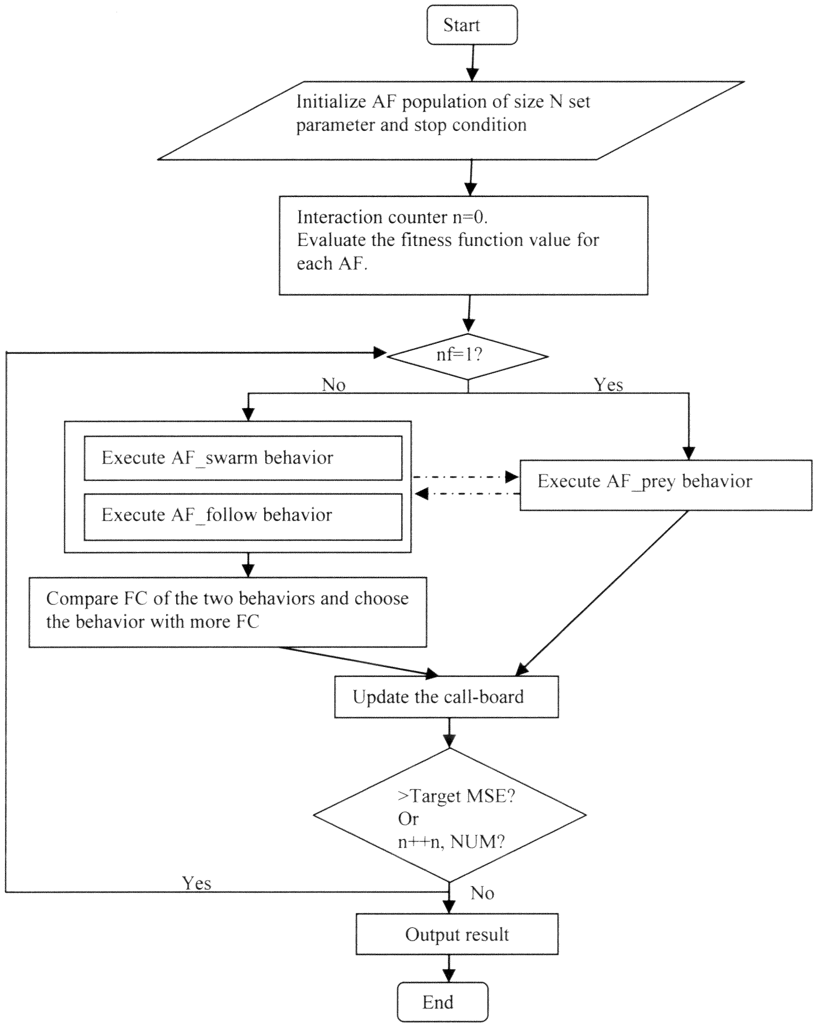

Start
Initialize AF population of size N set parameter and stop condition
Interaction counter n=0.
Evaluate the fitness function value for each AF.
nf=1?
No
Yes
Execute AF_swarm behavior
Execute AF_follow behavior
Execute AF_prey behavior
Compare FC of the two behaviors and choose the behavior with more FC
Update the call-board
>Target MSE?
Or
n++n, NUM?
Yes
No
Output result
End

Figure 1. Flowchart of AFSA.

FSA IN MLP LEARNING

ANN learning is very important to the realization of ANN structure and functions, particularly in network weights and biases. In optimizing the feed forward neural networks with AFSA, the network depends on the structure of AF. The network architecture consists of input layer, hidden layer and output layer. The total number of nodes for every layer is different, depending on the classification problem. The number of input layer and output layer usually come from the number of attributes and class attributes. However, there is no appropriate standard rule or theory to determine the optimal number of hidden nodes (Kim, Yoon, An, Cho & Kang 2004). In this study, each AF represents a feed forward neural network with inputs; (x_1, x_2, ... x_n), outputs; (y_1, y_2, ... y_m), hidden layer inputs; (s_1, s_2, ... s_h), hidden layer outputs; (z_1, z_2, ... z_h). The optimizing variables are represented as weights (w_1, w_2, ... w_h), and biases (θ_1, θ_2). Figure 2 depicts the architecture of the MLP feed forward network.

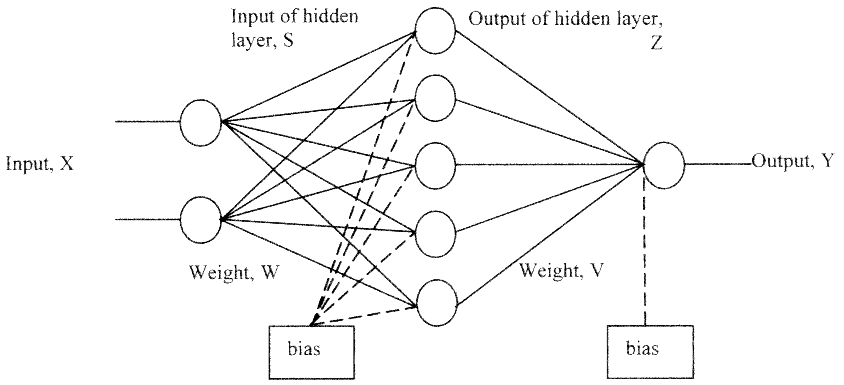


Figure 2. The MLP feedforward network.

In this study, sigmoid activation function was used to calculate an output for each neuron. The equation is:

$$f(x) = \frac{1}{(1+ e^{-x})} \tag{6}$$

where, x = input.

and feedforward NN formulas are:

$$s_i = \sum_{j=1}^{n} w_{ij}x_j + w_{i0} \quad , \quad 1 \le i \le h \tag{7}$$

$$z_i = f(s_i) \quad , \quad 1 \le i \le h \tag{8}$$

$$y_k = \sum_{i=1}^{h} v_{ki} z_i + v_{k0} \quad , \quad 1 \leq k \leq m \tag{9}$$

$$y_k = \sum_{i=1}^{h} v_{ki} f\left(\sum_{j=1}^{n} w_{ij} x_j + w_{i0}\right) + v_{k0} \text{ , } 1 \leq k \leq m \tag{10}$$

where,

w_{ij} = connecting weight between input and hidden layer,
w_{i0} = threshold values to hidden layer,
v_{ki} = connecting weight between hidden and output layer,
w_{i0} = threshold values to output layer.

The aim of feedforward NN training process is to get the minimum value of network error, E; by adjusting the weights and biases values. Hence, Mean Squared Error (MSE) function is used to quantify the error of the network. Assume, a dataset for training in neural network is A where is the input of the neural network, and is the desired output values:

$$A = \{(X^i, T^i) | \; i = 1, 2, \ldots n\} \tag{11}$$

$$E = \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{m} (T_k^i - y_k^i)^2 \tag{12}$$

For the enhancement of MLP learning with AFSA, each AF position represents a set of weights in NN. The AF adjusts its position based on the evaluation of current environment and selects a suitable behavior that has better improvement state. Food Concentration (FC) of an AF is obtained using the E value which is calculated from output of feedforward NN, and the formulation is:

$$\text{FC} = 1/(1+\text{E}). \tag{13}$$

If there are no companion AFs around AF_i, the states within visual position of AF_i in D-dimensional vector (X_i) is selected randomly as X_j. Let be the current state, randomly generate a number, *Xj* Visual. *try_num* is the maximum number of times that choose *Xj* to compare its FC with *Xi*, if FC of *Xj* is higher than FC of *Xi* (*FCj* > *FC$_i$*), *Xi* moves towards *Xj* in the range of *Step* with the food searching behavior. Figure 3 described the behaviour and is expressed as:

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_j - X_i^{(t)}}{||X_j - X_i^{(t)}||} \text{. Step.Rand (0,1)} \tag{14}$$

Otherwise,

$$X_i^{(t+1)} = X_i^{(t)} + \text{Step.Rand (-1,1),} \tag{15}$$

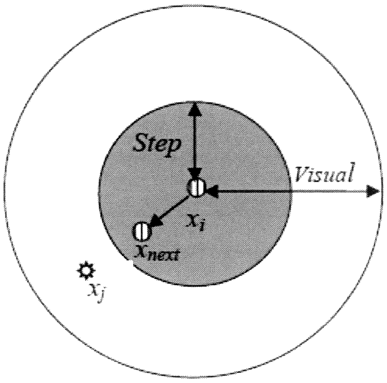


Figure 3. AF_prey X_i move towards $X_{j.}$

After *try_num* times, if none of the X_j has a higher FC, compare its FC with X_i , the X_i moves randomly in range of Step as shown in Fig.4. The procedure for *AF_prey* is shown in Figure 5.

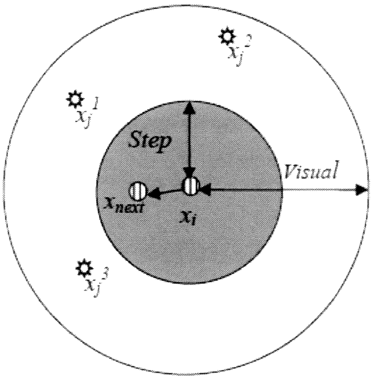


Figure 4. AF_prey X_i random move.

```
If the visual scope of the AF is not vacant
        Select an AF randomly in its visual scope;
        If the selected AFs FC is better;
                The AF moves towards the selected ones;
        Otherwise, the AF moves one step at random;
Else the AF moves one step randomly;
```

Figure 5. AF_prey behavior procedure[a]

Once there are companion AFs around the AF_i, they may swarm spontaneously and swim to share food in swarm. Referring to Figure 6, assume that is the current AF state and, is the center position of the food concentration between and its companion AF in *Visual*. If , it means that the companion center has more food (higher FC value) and is not very crowded. So, AF_i goes forward a step to the companion center and crowd together to share the same food.

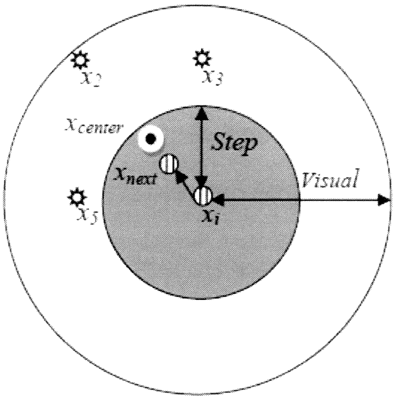


Figure 6. AF_swarm - X_i move towards X_{center}.

The swarm behavior is expressed as the procedure shown in Figure 7 and the function for to swarm is expressed as:

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_c - X_i^{(t)}}{||X_c - X_i^{(t)}||} \,.\, \text{Step.Rand}(0,1) \;, \qquad (16)$$

otherwise,

$$X_i^{(t+1)} = AF\_prey(X_j) \qquad (17)$$

```
Search the center site in an AF's visual scope
If the visual scope of the AF is not vacant
        If the FC is better and the center site is not crowded,
                The AF moves towards the center site;
        Otherwise, the AF executes AF_prey behavior;
Else the AF executes AF_prey behavior;
```

Figure 7. AF_swarm behavior procedure[2].

Referring to Figure 8, *AF_Follow* has as current state and X_{max} = max which is the AF with highest FC in visual range of AF$_i$. If , means that the companion with highest FC with a surrounding which is not very crowded will lead the swarm to follow the AF which discovers more food to share with it. The procedure of is shown in Figure 9 (CR.Wang.et.al. 2005) and the next position of can be expressed as: = + . Step.Rand(0,1)

$$X_i^{(t)} + \frac{X_{max} - X_i^{(t)}}{||X_{max} - X_i^{(t)}||} \text{. Step.Rand(0,1)} \qquad (18)$$

Otherwise,

$$X_i^{(t+1)} = AF\_prey(X_i) \qquad (19)$$

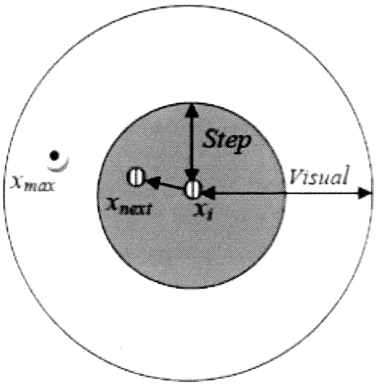
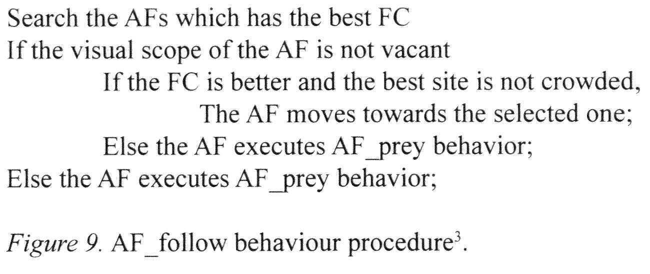


Figure 8. AF_follow - X_i move towards X_{max}.



```
Search the AFs which has the best FC
If the visual scope of the AF is not vacant
        If the FC is better and the best site is not crowded,
                The AF moves towards the selected one;
        Else the AF executes AF_prey behavior;
Else the AF executes AF_prey behavior;
```

Figure 9. AF_follow behaviour procedure[3].

Figure 9. AF_follow behavior procedure[1]

Consequently, the process of AFSA for MLP learning enhancement is as illustrated in Figure 10 and the experimental results will be discussed in the next section.

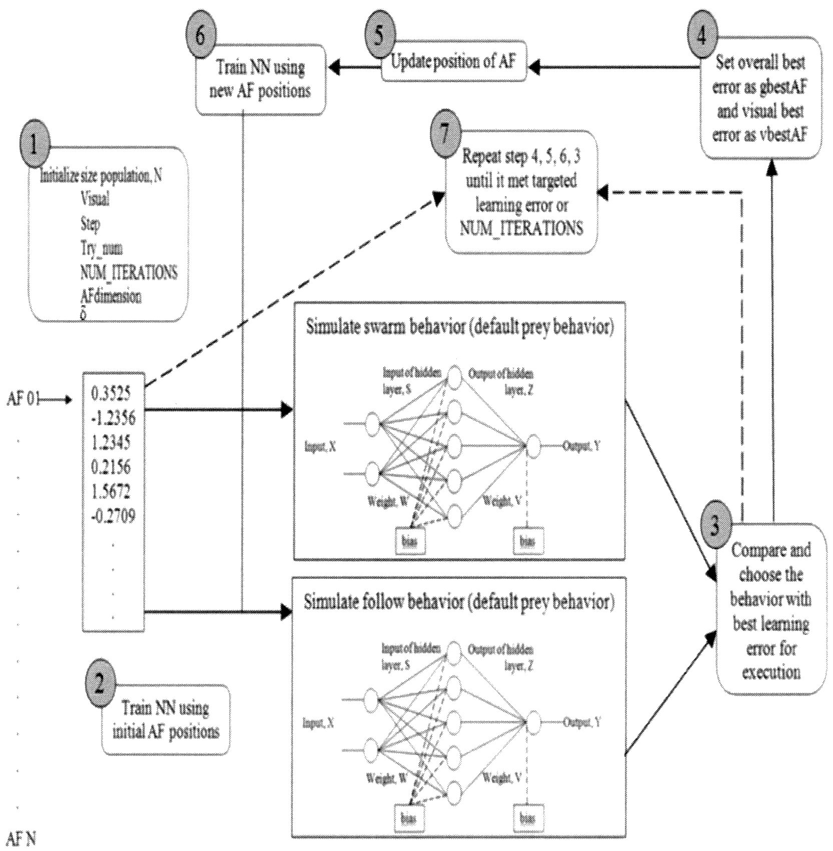

6
Train NN using new AF positions
5
Update position of AF
4
Set overall best error as gbestAF and visual best error as vbestAF
1
Initialize size population, N
Visual
Step
Try_num
NUM_ITERATIONS
AFdimension
δ
7
Repeat step 4, 5, 6, 3 until it met targeted learning error or NUM_ITERATIONS
AF 01
0.3525
-1.2356
1.2345
0.2156
1.5672
-0.2709
Simulate swarm behavior (default prey behavior)
Input of hidden layer, S
Output of hidden layer, Z
Input, X
Output, Y
Weight, W
Weight, V
bias
bias
3
Compare and choose the behavior with best learning error for execution
Simulate follow behavior (default prey behavior)
Input of hidden layer, S
Output of hidden layer, Z
Input, X
Output, Y
Weight, W
Weight, V
bias
bias
2
Train NN using initial AF positions
AF N

Figure 10. MLP learning process using AFSA.

EXPERIMENTAL SETUP AND RESULT

The parameter setting, experimental results and analysis of MLP learning enhancement with AFSA, referred to as AFSANN are discussed in this section. In addition, PSO and DE algorithms with MLP; referred to as PSONN and DENN are also analyzed and compared with AFSA. Each of the methods was tested with three datasets of scale, cancer and Iris. The results were analyzed based on the convergence rate and classification performance measurements.

Parameter Setting

In AFSANN, the parameter of Visual was set to provide the better convergence in whole area when its value is larger. Step parameter also can increase convergence rate when it has a larger value, besides it is faster in preying optimization. For, it should be smaller to get better convergence in the whole area because less AF will be maintained in the area, resulting in lower competition for food. According to Shi, (2004), PSO with a well-selected parameter set can have good performance. In PSONN, particle position values are initialized randomly with initial position velocity value set to 0. The C1 and C2 constant are set to 2 (Eberhart & Shi 2001). Furthermore, in DENN, Differentiation constant, F is set to 0.9 and crossover constant, Cr is set to 0.6. The size of population, NP is equal to 60 and generations is set to 100 (GEN =100). However, there are similar parameters amongst the methods. For instance, the range and problem dimension are based on NN architecture; stopping condition is set to NN minimum error or maximum number of iterations (Eberhart & Shi 2001). The parameters for AFSANN, PSONN and DENN are as described in Table 1.

Table 1

AFSA, PSONN and DENN Parameters

AFSANN		PSONN		DENN	
Parameter	Value	Parameter	Value	Parameter	Value
Visual	28.0	C_1	2.0	Differentiation (or mutation) constant *F*	0.9
Step	22.0	C_2	2.0	Crossover constant *Cr*	0.5
	0.02 (0.0 ~1.0)		0.1	N/A	
	5	N/A		Maximum number of generation (*GEN*)	100
Number of AF	20	Number of particles	20	Size of population *NP*	60
Problem dimension	Based on NN architecture	Problem dimension	Based on NN architecture	Problem dimension	Based on NN architecture
Range of AF	Not specified. Free to move anywhere	Range of particles	Not specified. Free to move anywhere	Range-Low and high boundary constraint	Based on dataset NN architecture

(continued)

AFSANN		PSONN		DENN	
Parameter	Value	Parameter	Value	Parameter	Value
Stop condition	NN minimum error or maximum number of iterations (100)	Stop condition	NN minimum error or maximum number of iterations	Stop condition	NN minimum error or maximum number of iterations

Analysis and Discussions

There are three core functions in AFSA, the AFSA_prey, AFSA_swarm and AFSA_follow. The experiments were tested on Scale, Cancer and Iris dataset. The results were validated by probing the best error convergence rate of all the iteration until they achieved optimal solution. The training stopping conditions were based on Mean Square Error (MSE) that had reached less than minimum error of 0.005 or reached the maximum number of iteration. The AFSANN provided stochastic output; hence 10 running times for training the dataset were simulated and recorded to get the average (refer to Table 2). The AFSA parameters used for training were 20 AF, each visual range (Visual) was 28.0, Step of 22.0 and delta was 0.05. From the table, it can be shown that AFSANN provides higher accuracy for all datasets accordingly.

Table 2

Classification Results using AFSANN

Dataset	Scale	Cancer	Iris
Convergence Time (s)	2.7	7.3	3
Learning Iterations	15.6	31.6	12.4
Correct Classification (%)	99.646	99.582	99.714
Error Convergence	0.003532198	0.003390274	0.003319926

This study tried to determine the efficiency of AFSA, as compared to PSO and DE in enhancing MLP learning. As heuristic algorithms which are close to optimal solution, the result with higher correct classifications is greater, probably due to actual output that is nearer to target output. A fair comparison was made throughout the experiment with same network inputs of data, network structures, sigmoid activation function and target MSE value. As the algorithms had stochastic performance, the best results between the recorded outputs was evaluated (refer to Table 3). It would seem that AFSANN yielded better accuracy compared to PSONN and DENN for the all datasets.

Table 3

Comparisons of AFSANN, PSONN and DENN

Dataset	Algorithm	AFSANN	PSONN	DENN
Scale	Convergence Time (s)	5	6	7
	Learning Iterations	20	141	18
	Correct Classification (%)	99.80	99.57	99.46
	Error Convergence	0.000223283	0.00494936	0.00127556
Cancer	Convergence Time (s)	25	17	14
	Learning Iterations	52	240	25
	Correct Classification (%)	99.95	99.49	98.23
	Error Convergence	0.00010583	0.00498231	0.0025403
Iris	Convergence Time (s)	3	4	3
	Learning Iterations	7	12	7
	Correct Classification (%)	99.79	99.43	98.69
	Error Convergence	0.00142662	0.00379363	0.000566319

CONCLUSION

Experimental results have demonstrated the effectiveness of AFSA in solving MLP weights optimization problem. The progress of the experiment significantly shows that AFSA overcomes local optimal problem, and shows promising results in terms of convergence error and classification accuracy compared to PSONN and DENN. The AFSANN has a more complex algorithm compared to PSONN and DENN, but it shows better performance in convergence rate and correct classification. Based on the analysis, AFSA is a robust method which is effectively applied in MLP learning through weight adjustment, and it achieves optimum in most cases. AFSANN convergence is faster with less iterations and yields better correct classifications compared to PSONN and DENN. In addition, AFSA has also three basic behaviors that enable it to get out from local minima when optimizing MLP weights. Therefore, the used dataset and generated network structure affect the result of convergence time, and iteration for AFSANN is not as critical as in PSONN and DENN. Finally, AFSA has shown its stability due to accuracy consistency throughout experiments and produced the best result on each respective dataset.

ACKNOWLEDGMENTS

The authors would like to thank the *Soft Computing Research Group (SCRG)*, Universiti Teknologi Malaysia (UTM) for their continuous support in making this study a success.

REFERENCES

Cao, C., Mao, C., Zheng, H., Jiang, X., & Zhang, K. (2009). AFSA-BP network speed identifier and application in direct Torque Control system. *Journal of Electric Drive, 39*(3), 3-6, 17.

Chen, H., Wang, S., Li, J., & Li, Y. (2007). *A hybrid of Artificial Fish Swarm algorithm and particle swarm optimization for feedforward neural network training*. Research Institute New Weaponry Technology and Application, Naval University of Engineering, Wuhan.

Eberhart, R., & Shi, Y. (2001). Particle swarm optimization. *Developments, Application and Resources*. IEEE, 81-86.

Firdaus, M. (2007). *Enhancement of elman recurrent network learning with particle swarm optimization* (Unpublished Master Thesis). University Teknologi Malaysia, Skudai.

Hasan, S., & Shamsuddin, S.M. (2011), Multi-Strategy self organizing map learning for classification problems. *Computational Intelligence and Neuroscience*. Hindawi Publisher.

Haza (2006). *Particle swarm optimization for neural network learning enhancement* (Unpublished master thesis). Universiti Teknologi Malaysia, Skudai.

Haykin, S. (2008). *Neural network and learning machines* (3rd ed.). United States of America: Pearson Prentice Hall.

Jiang, M., Wang, Y., Rubio, F., & Yuan, D. (2007). Spread spectrum code estimation by artificial fish swarm algorithm block-coding and antenna selection. *IEEE International Symposium on Intelligent Signal Processing (WISP)*.

Kim, G. H., Yoon, J. E., An, S. H., Cho, H. H., & Kang, K. I. (2004*)*. Neural network incorporating a genetic algorithm in estimating construction cost. *Building and Environment*, *39*(11), 1333-1340.

Kuok, K. K., Sobri, H., & Chiu, P. C., (2011). Hourly runoff forecast at different lead-time for a small watershed using artificial neural networks. *International Journal of Advances in Soft Computing and Its Applications (IJASCA*), *3* (1), 68-86.

Li, Z., Wang, D., Liang, L., & Zhou, Y. (2009). Artificial fish school algorithm for estimating parameters in production function. *Journal of Chongqing Normal University (Natural Science)*, *25*(4), 1-4.

Limsombunchai, M. C., & Weng, A. (2000). *Consumer choice prediction: artificial neural networks versus Logistic Mode*l. New Zealand: Lincoln University.

Peng, Y. (2011). An improved artificial fish swarm algorithm for optimal operation of cascade reservoirs. *Journal of Computers*, *6*(4), 740-746.

Shamsuddin, S. M., Sulaiman, M. N., & Darus, M. (2001). An improved error signal of back propagation model for classification problems. *International Journal of Computer Mathematics*, UK, *76*(2), 297-305.

Shamsuddin, S. M. (2004). *Lecture note advanced artificial intelligence: Number of hidden neurons* (Unpublished note). Universiti Teknologi Malaysia, Skudai.

Shamsuddin, S. M., Zainal, A., & Yusof, N. (2008). Multilevel kohonen network learning for clustering problems. *Journal of Information and Communication Technology, 7,* 1-25.

Shi, Y., & Eberhart, R. C. (1998). A modified particle swarm optimizer. *Proceedings of the IEEE International Conference on Evolutionary Computation,* 69-73.

Shi, Y. (2004). Particle swarm optimization. *IEEE Neural Network Society*. IEEE, 8-13.

Wang, C. R., Zhou, C. L., & Ma, J. W. (2005). An improved artificial swarm algorithm and its application in feedforward neural networks. *Proceeding of the 4th International Conference on Machine Learning and Cybernetics*, 2890 - 2894.

Wong, Y. L., & Shamsuddin, S. M. (2010). Writer identification for Chinese handwriting. (2010). *International Journal of Advances in Soft Computing and Its Applications (IJASCA)*, *2*(2), 147-173.

Wyeth, G., Buskey, G., & Roberts, J. (2000). *Flight control using an artificial neural network.* University of Queensland.

Zhang, M., Shao, C., Li, M., & Sun, J. (2006). Mining classification rule with AFS. *Proceeding of the 6th World Congress on Intelligent Control and Automation*. IEEE, 5877-5881.