ARTIFICIAL NEURAL NETWORK LEARNING ENHANCEMENT USING ARTIFICIAL FISH SWARM ALGORITHM

Shafaatunnur Hasan, Tan Swee Quo, Siti Mariyam Shamsuddin and Roselina Sallehuddin.

Soft Computing Research Group (SCRG), Universiti Teknologi Malaysia, Skudai, Johor, Malaysia.
{shafaatunnur@gmail.com, mariyam@utm.my, roselina@utm.my}

ABSTRACT. Artificial Neural Network (ANN) is a new information processing system with large quantity of highly interconnected neurons or elements processing parallel to solve problems. Recently, evolutionary computation technique, Artificial Fish Swarm Algorithm (AFSA) is chosen to optimize global searching of ANN. In optimization process, each Artificial Fish (AF) represents a neural network with output of fitness value. The AFSA is used in this study to analyze its effectiveness in enhancing Multilayer Perceptron (MLP) learning compared to Particle Swarm Optimization (PSO) and Differential Evolution (DE) for classification problems. The comparative results indeed demonstrate that AFSA show its efficient, effective and stability in MLP learning.

Keywords: Artificial Neural Network; Artificial Fish Swarm Algorithm; Classification problems.

INTRODUCTION

The main implication of Artificial Neural Network (ANN) is learning and improving through the environment where machine learning technique enables it to learn from experience, generalize on their knowledge, perform abstraction, make errors and does not need to be reprogrammed. The most common algorithm used in ANN is Backpropagation (BP) (Shamsuddin *et.al,* 2001). However, BP algorithm is always trapped in local minima and has slow convergence rate. Due to the weaknesses, Genetic Algorithm (GA) has been introduced to improve the BP network learning. Still, GA has complex functions and it is more time consuming in producing output. Subsequently, Swarm Intelligence (SI) technique called Particle Swarm Optimization (PSO) is became a popular method in because of its intuitiveness, its ease to be implemented and where it is more effective in solving nonlinear optimization problems. Otherwise, Differential Evolution (DE) has spontaneous self adaptability, diversity control and continuous improvements. Artificial Fish Algorithm (AFSA) is one of SI technique. The AFSA is intelligence and random search algorithm generated by studying the behavior of fish swarm in nature, solving optimization problem using swarm and artificial where it is less likely to get stuck in local minima, have adaptive ability and more capable in getting global optimum. It is a robust stochastic technique in solving optimization problem based on the movement and intelligence of swarms in food finding process. In previous related applications and research, AFSA has an advantages such as forming gradient information independently with objective function, the ability to solve complex nonlinear high dimensional problems, act as good global astringency, has strong robustness, insensitive to initial values, tolerance to parameters and simplicity of implementation. 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.

ARTIFICIAL NEURAL NETWORK (ANN)

An artificial neural network consists of a pool of simple processing units which communicate by sending signals to each other over a large number of weighted connections. In creating a functional model, there are three main basic components:

- The weight of the model which decides the strength of the connection between input and neuron. Negative weight values reflect inhibitory connections, while positive values designate excitatory connections (Haykin, 2008).
- The linear combination where it sums up all the inputs (from *j* to *p*) modified by their respective weights for neuron *i*. $v_k = \sum_{j=1}^{p} w_{kp} x_p$ (1)
- The sigmoid activation function that controls the amplitude of the output of neuron, between 0 and 1, denoted by $X = v_k + \theta_k$:
  $Y^{sigmoid} = \tanh\left(\frac{v}{2}\right) = \frac{1}{1+e^{-X}}$ (2)

In order to be a good predictor, an error is composed from the different between desired and actual output. The information in this error is feed back to the system so that the parameters can be adjusted in a systematic fashion (the learning rule). The network error, Mean Square Error (MSE),

$$\text{MSE} = \frac{1}{2}\sum(desired - actual)^2 \quad (3)$$

ARTIFICIAL FISH SWARM ALGORITHM (AFSA)

The basic behaviors of AFSA are AF_Prey, AF_Swarm, AF_Follow where AF_Prey is food searching, AF_Swarm is the fish get together in group and AF_Folllow is where the swarm will follow a fish when it found food. Firstly, the algorithm constructs simple basic behaviors of artificial fish, then based on local searching behaviors, come out with global optimum finally. AFSA can search global optimum efficiently and has certain ability in adapting to space searching. Acording to Chen, *et al.*, (2007), The AF will evaluate current environment and select suitable behavior that has better improvement state to be executed. Let i^{th} AF represented with a D-dimensional vector $X_i = (x_1, x_{2, \dots,} x_D)$ where i= 1,2, … n and a random selected states within visual position of X_i is $X_j = (x_1^j, x_2^j, \dots, x_n^j)$, where

$$y = f(x) \quad (4)$$
$$d_{ij} = \| x_i - x_j \| \quad (5)$$
$$x_j = x_i + \text{Visual.Rand()},\ i \in (O,n] \quad (6)$$
$$S = \{X_f \mid \|X_i - X_j\| < \text{Visual}\} \quad (7)$$
$$X_{next} = X_i + \frac{x - x_i}{\|x - x_i\|} \cdot \text{Step.Rand()} \quad (8)$$

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

AFSA IN ANN LEARNING

In optimizing the feed forward neural networks with AFSA, the network depends on the structure of AF. Each AF represents a feed forward neural network. The optimizing variables are weight matrix (w_1, w_2, … w_h), and bias (θ_1, θ_2), in the neural network. The inputs are (x_1, x_2, … x_n), outputs are (y_1, y_2, … y_m), inputs of hidden layer are (s_1, s_2, … s_h), output of hidden layer are (z_1, z_2, … z_h). The activation function used to calculate output for each neuron except input neuron is Sigmoid Activation/Transfer Function Equation as shown below:

$$f(x) = \frac{1}{(1+e^{-x})} \tag{9}$$

where, x = input

Let w_{ij} be the connecting weight between input and hidden layer, w_{i0} are the threshold values to hidden layer, v_{ki} be the connecting weight between hidden and output layer, w_{i0} are the threshold values to output layer. The computing formulas are:

$$s_i = \sum_{j=1}^{n} w_{ij}x_j + w_{i0} \quad , \quad 1 \le i \le h \tag{10}$$

$$z_i = f(s_i) \quad , \quad 1 \le i \le h \tag{11}$$

$$y_k = \sum_{i=1}^{h} v_{ki}z_i + v_{k0} \quad , \quad 1 \le k \le m \tag{12}$$

Overall, the output of feedforward NN will be:

$$y_k = \sum_{i=1}^{h} v_{ki} f\left(\sum_{j=1}^{n} w_{ij}x_j + w_{i0}\right) + v_{k0} \quad , \quad 1 \le k \le m \tag{13}$$

The feedforward NN training process is to get the minimum value of network error, E by adjusting the weights and biases values. The nonlinear error function chosen is Mean Square Error to quantify the error of the network. Suppose that the dataset for training in neural network is A where X^i is the input of the neural network and T^i is the desired/targeted output values:

$$A = \{(X^i, T^i) \mid i = 1, 2, \dots n\} \tag{14}$$

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m}(T_k^i - y_k^i)^2 \tag{15}$$

In enhancing the MLP learning using AFSA algorithm, each AF position represent set of weights in NN. The AF will adjust its position by evaluate current environment and select suitable behavior that has better improvement state to be executed. Food Concentration (FC) of an AF is obtained using the E value calculated from output of feedforward NN as:

$$\text{FC} = 1/(1+\text{E}). \tag{16}$$

If there are no companion AF around AF_i, randomly select states within visual position of AF_i in D-dimensional vector (X_i) as X_j. Let $\mathrm{X_i}$ be the current state, randomly generate a number, $Xj \in$ Visual. *try_num* is the maximum number of times that choosing 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 AF_prey($\mathrm{X_i}$) expressed as:

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_j - X_i^{(t)}}{||X_j - X_i^{(t)}||} \text{. Step.Rand(0,1)} \tag{17}$$

Otherwise,

$$X_i^{(t+1)} = X_i^{(t)} + \text{Step.Rand(-1,1)}, \tag{18}$$

After *try_num* times, if there are none of the X_{j} has higher FC compare its FC with X_{i}, the X_{i} move randomly in range of Step. When there are companion AF around the AF_i, it may swarm spontaneously when swimming to share for food in the swarm. Assume that X_i is the current AF state and $X_{center} = \frac{\sum x_j}{nf}$, is the center position of the food concentration between X_i and its companion AF in *Visual*. If $\frac{FC_c}{nf} > \delta FC_i$ 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.

The function for X_i to swarm is expressed as:

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_c - X_i^{(t)}}{||X_c - X_i^{(t)}||} . \tag{19}$$

Step.Rand(0,1), otherwise,

$$X_i^{(t+1)} = AF\_prey(X_j) \tag{20}$$

The *AF_Follow* has X_i as current state and $X_{max} = \max\{f(X_j) | X_j \epsilon\ S\}$ which is the AF with highest FC in visual range of AF_i. If $\frac{FC_{max}}{nf} > \delta FC_i$, 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 next position of X_i can be expressed as:

$$X_i^{(t+1)} = X_i^{(t)} + \frac{X_{max} - X_i^{(t)}}{||X_{max} - X_i^{(t)}||} . \text{ Step.Rand}(0,1) \tag{21}$$

Otherwise, $X_i^{(t+1)} = AF\_prey(X_i)$ (22)

In applying the AFSA in enhancing the MLP learning, the parameter of Visual is set to provide the better convergence in whole area when its value is larger. Step parameter also can increase convergence rate when it has larger value, besides it is faster in preying optimization. For δ , it should be smaller to get better convergence in the whole area because less AF will be maintained in the area resulting lower competition of food.

EXPERIMENTAL SETUP AND RESULT

There are three core functions in AFSA, the AFSA_prey, AFSA_swarm and AFSA_follow. The experiments are tested on Scale, Cancer and Iris dataset. The results are validated by probing the best error convergence rate of all the iteration until it achieves optimal solution. The training stopping conditions are either Mean Square Error (MSE) that has reached less than minimum error of 0.005 or reached maximum number of iteration. The AFSANN gives stochastic output; hence 10 running times for training the dataset are simulated and recorded to get the average (refer to Table 1). The AFSA parameters used for training are 20 AF, each visual range (Visual) is 28.0, Step of 22.0 and delta is 0.05. From these tables, it show that AFSANN provides higher accuracy for all datasets accordingly.

Table 1. The average result of AFSANN on Scale, Cancer and Iris Dataset.

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 aims to determine the efficient of AFSA compared to PSO and DE in enhancing the MLP learning in term of convergence rate and correct classification. As heuristic algorithms which hope for a solution that to be closed to optimal solution, result with higher correct classification is more likely to have actual output that is near similar to target output. A fair comparison will be made through training using same network inputs of data, network structure, sigmoid activation function and target MSE value. As the three algorithms have stochastic performance, the best result between the recorded outputs is compared in Table 2. It seems that AFSANN yields better accuracy compared to PSONN and DENN for the all datasets.

Table 2. The average result of AFSANN on Scale, Cancer and Iris Dataset.

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

Meanwhile, Figure 10 compares the classification percentage for all 3 datasets used to train on MLP feedforward neural network using those three algorithms. It gives a clear picture that AFSA is having higher percentage than PSO and DE in all dataset training. Calculated in average of correct classification, AFSANN gives 99.85%, PSONN achieved 99.50% and DENN has 98.80%.

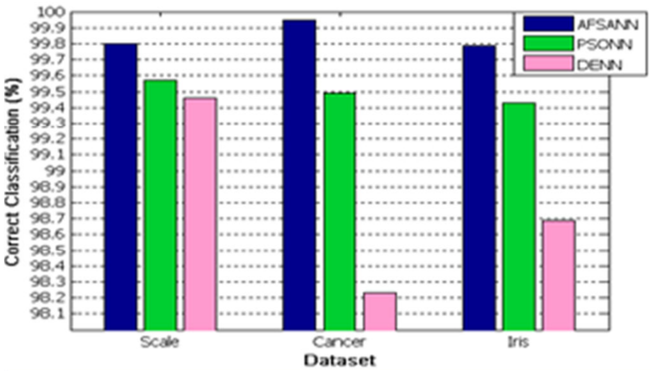

Figure 90. Classification percentage of AFSANN, PSONN and DENN.

CONCLUSION

Experiment results have demonstrated the effectiveness of AFSA in solving MLP neural network weight optimization problem. The progress of the experiment significantly shows that AFSA overcomes local optimal value problem and shows robustness in convergence error and classification accuracy compared to PSO and DE. The AFSA has more complex algorithm compared to PSO and DE, but it shows better in convergence rate and correct classification. Based on the analysis on comparison made, it can be conclude that AFSA algorithm is a robust method that can be effectively applied in MLP learning through weight adjustment and it achieves optimum in most cases. Besides that, AFSA algorithm converges faster in less iteration and with better correct classification compared to PSO and DE in enhancing MLP learning. AFSA also has 3 behavior that enable it to step out of local minima when optimizing weights if MLP in training process. Therefore, the dataset used and network structure generated in affecting the result of convergence time and iteration for AFSA is not as critical as for PSO and DE in enhancing MLP learning. AFSA has good global astringency and population position initialized in the beginning of the training does not affect efficiency of AFSA in terms of convergence time and iterations, compared to PSO and DE algorithm, in enhancing MLP learning. Finally, AFSA has good stability because the average result on a set of training is near similar to all of the results, including best result in the experiments on each respective dataset.

ACKNOWLEDGMENTS

Authors would like to thanks Research Management Centre (RMC) and *Soft Computing Research Group (SCRG)*, Universiti Teknologi Malaysia (UTM), for making this study success under vote 77316.

REFERENCES

Eberhart, R. and Shi, Y. (2001). Particle Swarm Optimization. *Developments, Application and Resources.* IEEE: 81-86.

H. Chen., S. Wang, J. Li and Y. Li..(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.

S. Haykin (2008). *Neural Network and Learning Machines*. (3rd ed.). United States of America: Pearson Prentice Hall.

S.M, Shamsuddin, M.N Sulaiman, & M. Darus (2001). An Improved Error Signal of Back propagation Model for Classification Problems. *International Journal of Computer Mathematics*, UK, **76**(2):297-305