# NEURAL NETWORK TRAINING USING HYBRID PARTICLE-MOVE ARTIFICIAL BEE COLONY ALGORITHM FOR PATTERN CLASSIFICATION

**Zakaria Noor Aldeen Mahmood Al Nuaimi & Rosni Abdullah**
*School of Computer Sciences*
*Universiti Sains Malaysia, Malaysia*

*zqttan2@yahoo.com; rosni@cs.usm.my*

## ABSTRACT

The Artificial Neural Networks Training (ANNT) process is an optimization problem of the weight set which has inspired researchers for a long time. By optimizing the training of the neural networks using optimal weight set, better results can be obtained by the neural networks. Traditional neural networks algorithms such as Back Propagation (BP) were used for ANNT, but they have some drawbacks such as computational complexity and getting trapped in the local minima. Therefore, evolutionary algorithms like the Swarm Intelligence (SI) algorithms have been employed in ANNT to overcome such issues. Artificial Bees Colony (ABC) optimization algorithm is one of the competitive algorithms in the SI algorithms group. However, hybrid algorithms are also a fundamental concern in the optimization field, which aim to cumulate the advantages of different algorithms into one algorithm. In this work, we aimed to highlight the performance of the Hybrid Particle-move Artificial Bee Colony (HPABC) algorithm by applying it on the ANNT application. The performance of the HPABC algorithm was investigated on four benchmark pattern-classification datasets and the results were compared with other algorithms. The results obtained illustrate that HPABC algorithm can efficiently be used for ANNT. HPABC outperformed the original ABC and PSO as well as other state-of-art and hybrid algorithms in terms of time, function evaluation number and recognition accuracy.

**Keywords:** Swarm Intelligence, Artificial Neural Networks, Artificial Bee Colony Algorithm, Particle Swarm Optimization, Pattern-Classification.

Received: 5 September 2016 Accepted: 13 June 2017

## INTRODUCTION

Swarm Intelligence (SI)-based optimization algorithms are the major concern for research in the field of optimization. The term swarm is used to interpret a gathering of animals like fishes, birds and insects such as ants, termites and bees performing collective behavior. SI algorithms are inspired by observing the behavior of the animals and insects when they seek food. During recent years many SI algorithms have been proposed for the optimization search process. Algorithms such as Particle Swarm Optimization (PSO)(Eberhart & Kennedy, 1995), Ant Colony Optimization (ACO)(Dorigo & Gambardella, 1997), Bacterial Foraging Optimization (BFO)(Passino, 2002), Artificial Fish Swarm Algorithm (AFSA) (Li, Shao, & Qian, 2002) and Artificial Bee Colony (ABC) algorithm (Karaboga, 2005), have been applied on different types of real-world optimization problems and they proved their powerful and effective performance.

Artificial Neural Networks is a mathematical approximation tool inspired by the human biological neural networks in the brain and has the ability to mimic its structure and functionality. ANN can be used for classification, clustering and prediction of objects. Basically it can learn from its previous experience and training in order to identify new objects. This type of training is considered as a supervised one, in which the ANN is trained on a certain type of collected data that represent the objects' specific features. Then, based on this training the ANN is able to classify object and patterns or predict information info based on its previous knowledge (Zhang, Patuwo, & Hu, 1998).

The feed forward ANN training is based on adjusting the values of a certain number of pre-initialized weights according to the ANN predefined structure. The standard gradient-descent Back Propagation (BP) learning algorithm, is one of the traditionally used algorithms for the feed forward ANN training (Kattan & Abdullah, 2013). Recently, due to the drawbacks that such trajectory-driven algorithms suffer from, algorithms that are population driven such as ABC (Karaboga, Akay, & Ozturk, 2007), Genetic Algorithm (GA) (Montana & Davis, 1989), Particle Swarm Optimization (PSO) (Eberhart & Kennedy, 1995), Ant Colony Optimization (ACO)(Blum & Socha, 2005), Harmony Search (Kattan & Abdullah, 2013) and Mussels Wandering Optimization (MWO) (Abusnaina, Abdullah, & Kattan, 2014) were introduced for ANN training.

The Artificial Bee Colony (ABC) optimization algorithm is considered as one of the competitive algorithms in the field of the SI-based algorithms. Karaboga in 2005, introduced the ABC algorithm inspired by the foraging behavior

of the bees in nature (Karaboga, 2005). It has the ability to find optimal solutions with variant computational requirements when applied on different optimization problems (Bahamish, Abdullah, & Salam, 2009; Karaboga, 2005; Karaboga & Basturk, 2007; Karaboga & Gorkemli, 2011; Karaboga & Ozturk, 2010; Mahmood, Mahmuddin, & Mahmood, 2012; SyarifahAdilah, Abdullah, & Venkat, 2012).

The ABC algorithm was also applied on Artificial Neural Networks Training (ANNT) for pattern classification problem. The ABC was compared with different algorithms including back-propagation and Levenberg-Marquardt, as well as other population-based algorithms, namely Particle Swarm Optimization, Differential Evolution and Genetic Algorithm. The results indicated that the ABC is a competitive algorithm and can successfully be applied to train feed-forward neural networks on classification problems (Ozturk & Karaboga, 2011) . Moreover, the ABC algorithm was used to improve the ANNT performance for other problems such as the Encoder-Decoder problems (Karaboga et al., 2007), MLP training on earthquake time series data prediction (Ozturk & Karaboga, 2011), biochemical networks approximation (Mustaffa, Yusof, & Kamaruddin, 2013), software defect prediction (Farshidpour & Keynia, 2012) and energy fuel price prediction (Mustaffa et al., 2013).

However, the original ABC algorithm was known for the weaknesses of its local search characteristics and strategies. It has been modified several times in order to improve its overall search performance. These modifications were either by adding some steps or by hybridizing it with other algorithms. Bio-inspired algorithms such as the ABC algorithm had shown their efficiency when utilized in a hybridized form (Satpathy, 2017). Different hybridization attempts were performed with the ABC algorithm in order to improve its performance in the ANNT process (Ghanem & Jantan, 2014; Ozturk & Karaboga, 2011; Yaghini, Khoshraftar, & Fallahi, 2013). The hybrid ABC results obtained from previous researches indicated that the ABC algorithm can be significantly improved to give better results than those found in other studies.

In this work, the hybrid particle-move artificial bee colony HPABC algorithm (Alqattan & Abdullah, 2015) was presented as a new evolutionary training algorithm for the feed forward ANN. The experiments were conducted using four benchmark pattern classification datasets retrieved from the UCI repository (Lichman, 2013). A comparison was made using the results of the original ABC and PSO algorithms against that of the HPABC algorithm in order to illustrate the powerful search performance of the hybrid algorithm.

## HPABC ALGORITHM

Hybrid Particle-move Artificial Bee Colony (HPABC) Algorithm was initially proposed as an optimization algorithm for numerical functions (Alqattan & Abdullah, 2015). The algorithm is basically inspired by the unique solution improvement or the particle move operation that the Particle Swarm Optimization (PSO) algorithm performs during its search operation. In other words, the HPABC is a hybrid algorithm which complements the advantages of the ABC algorithm with those of the PSO algorithm.

In the HPABC algorithm (Alqattan & Abdullah, 2015), the colony of the artificial bee contains only two groups of bees: onlookers and scouts. The hall colony members are onlooker bees. The onlooker bee of an abandoned food source becomes a scout. Note that the employed bees phase is eliminated in HPABC.

In the HPABC algorithm the number of onlooker bees is equal to the number of solutions (*SN*) in the population. Randomly initialized solutions are produced in the earlier stages of the HPABC. The solution is represented using a D-dimension vector for each solution $x_{id}$ ($i$=1, ..., *SN*), where $d$ is the number of the parameters or variables to be optimized. After the initialization stage, come repeated cycles of a searching process ($C$=1, 2, ..., *MCN*) where Maximum Cycles Number (*MCN*) is the stopping condition.

In HPABC the artificial onlooker bee selects a food source based on the probability value $p_i$ associated with it using the following equation:

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{2.1}$$

where $fit_i$ is the fitness value of the solution $i$ which represents the nectar amount of the food source in the position $i$ SN is the number of food sources. The HPABC algorithm uses the particle movement equations of the PSO algorithm, represented in Eq. 2.2 and 2.3, for the new food source position production.

$$V_{i,d}^{t+1} = w^t * V_{i,d}^t + c_1^t * r_1 * \left(pbest_{i,d}^t - X_{i,d}^t\right) + \tag{2.2}$$

$$c_2^t * r_2 * \left(gbest_{i,d}^t - X_{i,d}^t\right) \tag{2.3}$$

$$X_{i,d}^{t+1} = X_{i,d}^t + V_{i,d}^{t+1}$$

where $w$ represents inertia weight; $c_1$ and $c_2$ are learning factors which determine the relative influence of cognitive (self-confidence) and social (swarm-confidence) components, respectively, $r_1$ and $r_2$ are independent random numbers uniformly distributed in the range [-1,1]. $V_{i,d}^{t+1}$, $X_{i,d}^{t}$ and $pbest_{i,d}^{t}$ are the velocity, current position and the personal best of the $i$th particle in $d$th dimension for the $t$th iteration, respectively. The $gbest_{i,d}$ is the $d$th dimension of the best particle in the swarm for the $t$th iteration. In HPABC algorithm, the particle position in the equations (2.2 and 2.3) will be the onlooker bee position, and the values of the velocity parameter, the local and global best parameters will also be calculated (same as in the original PSO algorithm).

After a number of HPABC search cycles the food source with unimproved fitness will be abandoned based on the *limit* parameter value (same as in the original ABC), and the scout bee will be sent to find a new food source based on Eq. (2.4). Then the food source found by the scout bee will take the place of the abandoned one.

$$x_{ij} = x_{ij} + rand[0,1]\left(x_{jmax} - x_{jmin}\right) \qquad (2.4)$$

However, for the HPABC algorithm and unlike the ABC algorithm, it has six controlled parameters: number of food sources *SN*, *limit*, inertia weight $w$, $c_1$ and $c_2$ learning factors and the maximum cycle number *MCN*.

The pseudo-code of the HPABC algorithm is given below:

*1:* *Initialize the population of solutions xi, i=1,...,SN.*
*2:* *Evaluate the population using the probability factor using Eq. (2.1).*
*3:* *cycle =1.*
*4:* ***repeat***
*5:* *Produce new solution* $X_{i,d}^{t+1}$ *for the onlooker bee using Eq. (2.2&2.3) from solution* $X_{i,d}^{t}$ *which is selected based on the its probability value.*
*6:* *Apply the greedy selection process.*
*7:* *Calculate the probability value* $p_i$ *for the new solution* $X_{i,d}^{t+1}$ *using Eq. (2.1).*
*8:* *Determine the abandoned solution for the scout, if it exists, replace it with a new random solution using Eq. (2.4).*
*9:* *Memorize the best solution found so far.*
*10:* *cycle = cycle +1.*
*11:* ***Until*** *cycle = MCN.*

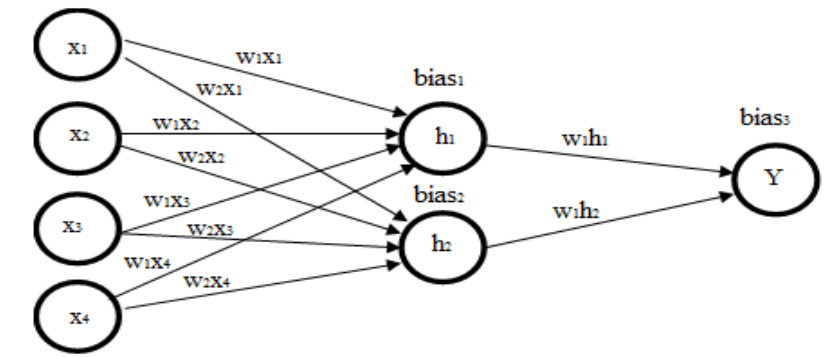


*Figure 1*. ANN basic units.

## ANN TRAINING

Basically, the ANN has simple units called neurons or nodes, each of which is interconnected with each other. These nodes are divided into three types: input, hidden and output nodes (see Figure 1). Input nodes represent the input data, hidden nodes are the main processing units that hold the calculation process, while the output nodes represent the ANN output results. However, the hidden nodes can be structured as a single or multiple layers of nodes that are connected to each other. The output of the *i*th node of those hidden nodes can be described by Eq. (3.1).

$$y_i = f_i\left(\sum_{j=1}^{n} w_{ij}\, x_j + \theta_i\right) \tag{3.1}$$

where $y_i$ is the output of the hidden node $i$, $x_j$ is the $j$th input to the hidden node; $w_{ij}$ is the connected weight between the hidden node and the input node $x_j$, $\theta_i$ is the threshold (bias) of that hidden node and $f_i$ is the hidden node transfer function (sigmoid function is used in this work).

The training of the feed forward ANN is based on adjusting the corresponding weights and biases of the specified ANN structure in order to minimize the error rate between the target and the actual output of a supervised ANN. In

this work we are tried to use the HPABC algorithm in order to adjust those weights and biases and improve the speed and the performance of the ANN. As presented in previous research (Abusnaina et al., 2014; Karaboga et al., 2007; Kattan & Abdullah, 2013), optimization search algorithms can be applied for ANN training, where the weights and biases of a fixed ANN structure can be fed into the algorithm as a vector of values representing the weights and biases of the corresponding nodes.

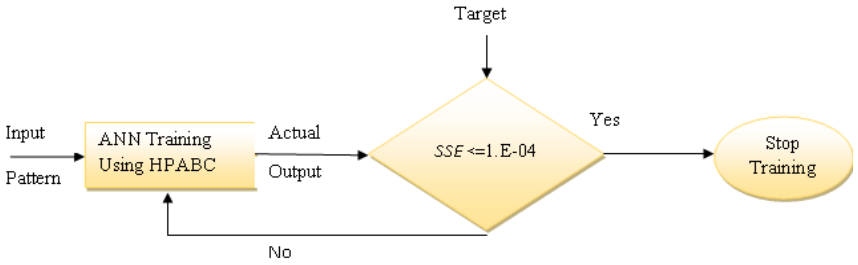


*Figure 2.* ANN training using optimization algorithm.

The error rate function (evaluation function) used in this work is the Sum Squared Error (*SSE*) (see Eq. (3.2)). Thus, the objective of the HPABC algorithm is to search for the minimum *SSE* value by adjusting the values of the ANN vector correspondingly. The ANN training mechanism using HPABC is illustrated in Figure 2.

$$SSE = \frac{1}{n}\sum_{i=1}^{n}(t_i - y_i)^2 \qquad (3.2)$$

where: $n$ is the number of the patterns in the selected benchmark dataset (training or testing set), $t$ is the target of the $i^{th}$ pattern, y is the actual output of the $i^{th}$ pattern. Thus, the ANN training using optimization algorithm is basically to find the optimal ANN weights in which the *SSE* reaches the lowest value.

## EXPERIMENTAL STUDY

To fairly evaluate the HPABC algorithm performance, we did a number of experimental tests. The first test was done on the parameters settings, which basically focus on finding the best settings of the classical ABC and PSO algorithms when applied on the ANN training problem. Thus, we tested the

main parameters of ABC which are the Colony size and limit and the main parameters of the PSO algorithm which are the *Swarm Size* (*SS*), $w$, $c_1$ and $c_2$. The best parameter settings of the ABC and PSO will be used for the HPABC algorithm for comparison and evaluation purpose. The other tests were on the evaluation process of the HPABC which are presented in the Evaluation subsection.

**Benchmark Datasets**

For the experimental tests and evaluation purpose, four types of benchmark pattern-classification datasets were used namely; Iris, Cancer, Diabetes and Glass. The main reason for selecting these data sets among many others is that they have no missing input feature values. In addition, these problems have been used and cited in the neural networks, classification and machine learning literature (Abusnaina et al., 2014; Camargo, Correa Tissot, & Ramirez Pozo, 2012; Kattan & Abdullah, 2013; Nur, 2015; Vazquez & Garro, 2015). The datasets were obtained from the Machine Learning Repository (UCI). The datasets description is shown in Table 1.

Table 1

*Datasets Description*

| Dataset name | No. of attributes | No. of instances | No. of classes |
|---|---|---|---|
| Iris | 4 | 150 | 3 |
| Cancer | 9 | 699 | 2 |
| Diabetes | 8 | 768 | 2 |
| Glass | 9 | 214 | 6 |

The ANN structure was designed based on a 3-layer architecture (input-hidden-output). The structure of each dataset is shown in Table 2. The ANN structure was selected based on previous work (Abusnaina et al., 2014; Kattan & Abdullah, 2011, 2013).

In Tables 1 and 2, the Cancer dataset has a moderate complexity in terms of ANN architecture and a sufficient number of instances compared to the other datasets. Therefore, the Cancer dataset was selected for the parameter settings experiments. The best parameter settings (for the compared algorithms) obtained using the Cancer dataset can be generalized and used for the other datasets.

Table 2

*ANN Datasets Structure*

| Dataset name | No. of input nodes | No. of hidden nodes | No. of output nodes |
|---|---|---|---|
| *Iris* | 4 | 5 | 3 |
| *Cancer* | 9 | 8 | 2 |
| *Diabetes* | 8 | 7 | 2 |
| *Glass* | 9 | 12 | 6 |

**ABC Parameter Settings**

In order to fairly compare the search performance of the proposed HPABC algorithm with the original ABC algorithm, the parameter settings of the ABC which enable it to reach the best results when used on the benchmark datasets should also be used by the HPABC.

The Cancer dataset was randomly split into 70% for training and 30% for testing. The main goal of this experiment was to find the best Colony Size number and the best value of limit which affect the performance of ABC. Table 3 illustrates the performance of the ABC algorithm on the Cancer dataset. Each Colony Size (CS) value evaluation was done 30 times, the limit was set to 100 and the stopping condition was when SSE reaches (0.0001). The results are illustrated using the Mean, Best, Worst evaluation parameters for the time elapsed in seconds (S), and the mean value for the number of Function Evaluation (Mean (FE)).

Table 3

*The ANN Training for "Cancer" Dataset with Different ABC Colony Size*

| Colony size (CS) | Mean (FE) | Time elapsed (S) | | |
|---|---|---|---|---|
| | | Best | Worst | Mean |
| **10** | **4750** | **3.35** | **12.03** | **8.14** |
| 20 | 9188 | 6.20 | 22.19 | 14.53 |
| 30 | 13202 | 14.55 | 28.75 | 21.06 |
| 40 | 16336 | 15.38 | 35.76 | 27.11 |
| 50 | 19983.3 | 21.70 | 51.53 | 36.34 |
| 60 | 23176 | 26.64 | 60.70 | 45.63 |

(continued)

| Colony size (CS) | Mean (FE) | Time elapsed (S) | | |
|---|---|---|---|---|
| | | Best | Worst | Mean |
| 70 | 26502 | 27.40 | 73.83 | 53.76 |
| 80 | 28597.3 | 30.12 | 124.48 | 68.79 |
| 90 | 30780 | 26.29 | 69.12 | 51.15 |
| 100 | 35546.6 | 38.65 | 110.78 | 74.84 |

Table 3, shows that increasing the *Colony Size* number has a negative influence on the ABC algorithm in training. This means that when the number of bees increase, the function evaluation number will also increase, with no improvement in the results. The best *CS* number found in this experiment was 10, where the best time for the ABC to reach the training performance of *SSE* = 0.0001 was 3.35s, the *Worst* was 12.03s and the *Mean* was 8.14s.

The same experiment was done for *limit* parameter using the same previous setting where *CS* was set to 10 and with different *limit* values. The results are shown in Table 4.

Table 4

*The ANN Training with Different "limit" Values*

| *limit* | Mean (FE) | Time elapsed (S) | | |
|---|---|---|---|---|
| | | Best | Worst | Mean |
| 100 | 5408 | 4.71 | 14.85 | 8.41 |
| 200 | 5028 | 4.02 | 13.46 | 8.24 |
| **300** | **4897.3** | **4.49** | **14.63** | **9.38** |
| 400 | 5323.3 | 5.32 | 15.47 | 10.37 |
| 500 | 5256 | 4.37 | 11.16 | 8.09 |
| 600 | 5206 | 4.61 | 13.52 | 8.01 |
| 700 | 5320.6 | 4.71 | 15.04 | 8.22 |
| 800 | 5212 | 2.80 | 14.26 | 8.05 |
| 900 | 5552 | 4.99 | 14.72 | 9.32 |
| 1000 | 5026.6 | 5.30 | 15.36 | 10.35 |

The results in Table 4, show no significant difference in the time elapsed for ANN training when the limit value was increased from 100-1000 and the FE

value shows no clear influence too. The best result was for limit = 300, the Best = 4.49s, Worst = 14.63s, Mean = 9.38s and the FE number = 4897.3. However, the best parameter settings for the ABC algorithm that were experimentally obtained were: Colony Size = 10 and limit = 300. These settings would be used in further comparison evaluations for the ABC and the HPABC algorithms.

**PSO Parameter Settings**

The same experiments were done on the PSO parameters in order to find their its best settings for ANN training. The PSO parameters were *Swarm Size* (*SS*), $w$, $c_1$ and $c_2$, where *SS* was the number of particles, $w$ represented inertia weight, $c_1$ and $c_2$ were learning factors (See section 2). Since, the *Swarm Size* was similar to *Colony Size* for the ABC, we suggested setting it equal to 10, similar to the ABC algorithm.

As for the parameters $c_1$ and $c_2$, the experiments were conducted on the same dataset (*Cancer*) with *Swarm Size* = 10, $w$ = 0.729 (as recommended in Sharma, Pant, & Bhardwaj (2011)) with different values of $c_1$ and $c_2$ parameters starting from (0.1 – 0.9) for both. Table 5 illustrates the final results obtained from the experiments for 30 run times of the PSO algorithm for each $c_1$ and $c_2$ settings. Only the best mean results (mean of 30 runs) are shown for the time elapsed, and Table 6 illustrates the parameter *Mean* (*FE*) for function evaluation mean values.

Table 5

*PSO Best Evaluation Parameters Values for Different Values of $c_1$ and $c_2$*

| Evaluation parameter | $c_1$ | $c_2$ |
|---|---|---|
| Best results of (time mean *Best)* when | 1.7 | 1.4 |
| Best results of (time mean *Worst)* when | 1.1 | 1.2 |
| Best results of (time mean *Mean)* when | 1.1 | 1.2 |

It can be seen in Tables 5 and 6, that as the values of $c_1$ and $c_2$ increased, the PSO results became worse. The best results for PSO in training the ANN was when $c_1$=1.1 and $c_2$=1.2 where the values of *Best* = 1.177s, *Worst* = 4.754s, *Mean* = 1.814s, and *Mean (FE)* = 850.00. We considered them as the best results based on the *Mean* (*FE)* which was the common important and considerable algorithmic issue.

Table 6

*ANN Training using PSO (parameters $c_1$ and $c_2$ Mean (EF) means)*

| $c_2 \backslash c_1$ | 1.1 | 1.2 | 1.3 | 1.4 | 1.5 | 1.6 | 1.7 | 1.8 | 1.9 |
|---|---|---|---|---|---|---|---|---|---|
| 1.1 | 1323.00 | 2099.67 | 1447.00 | 1549.00 | 1364.33 | 1411.33 | 1867.33 | 1794.67 | 2036.33 |
| 1.2 | **850.00** | 3175.67 | 1507.00 | 3082.67 | 1589.67 | 1757.00 | 1798.00 | 1871.67 | 2520.67 |
| 1.3 | 2536.00 | 5507.67 | 1207.33 | 1453.00 | 1285.67 | 1421.00 | 1832.67 | 2366.67 | 2051.67 |
| 1.4 | 2629.33 | 10326.00 | 1628.33 | 1445.67 | 1872.67 | 1885.00 | 1976.00 | 1954.67 | 2748.33 |
| 1.5 | 1220.33 | 12563.67 | 1295.00 | 1733.00 | 1591.33 | 2618.00 | 2854.33 | 3679.00 | 3500.67 |
| 1.6 | 2055.00 | 31166.67 | 1486.00 | 1543.00 | 2603.67 | 2755.33 | 2928.67 | 2855.67 | 5647.33 |
| 1.7 | 1364.33 | 46613.33 | 1617.33 | 2080.33 | 2274.33 | 2157.00 | 2980.33 | 5138.00 | 8297.00 |
| 1.8 | 1304.00 | 47059.00 | 1960.33 | 2416.67 | 2786.33 | 3351.00 | 5972.00 | 6212.67 | 13085.67 |
| 1.9 | 1254.33 | 41027.33 | 1812.00 | 2486.33 | 3816.33 | 3431.00 | 11639.00 | 16754.33 | 17383.00 |
| 2 | 1700.00 | 37401.33 | 2481.33 | 3647.67 | 2951.00 | 6334.67 | 14419.33 | 23287.67 | 31785.33 |

The next experiment was done for the *w* parameter, using the same dataset (*Cancer*) with *SS* = 10, and $c_1$=1.1, $c_2$= 1.2 and different values of *w*, from (0.1-0.9) as shown in Table 7.

Table 7

*ANN Training using PSO Algorithm with Different "w" Values*

| *W* | Mean (FE) | Time elapsed (S) | | |
|---|---|---|---|---|
| | | Best | Worst | Mean |
| 0.1 | 50011 | 95.339 | 115.807 | 101.307 |
| 0.2 | 50011 | 96.469 | 101.576 | 99.387 |
| 0.3 | 43421.66 | 0.702 | 101.612 | 86.790 |
| 0.4 | 36781.66 | 0.308 | 103.195 | 73.865 |
| 0.5 | 12424.33 | 0.770 | 102.936 | 25.219 |
| 0.6 | 5872.66 | 0.709 | 102.760 | 12.031 |
| **0.7** | **1453.33** | 0.691 | 12.606 | 2.949 |
| 0.8 | 1824.66 | 1.656 | 11.478 | 3.676 |
| 0.9 | 37000 | 5.884 | 91.954 | 67.089 |
| 1 | 49741.66 | 73.329 | 88.065 | 87.202 |

Table 7 shows that the best results were obtained when the w parameter was (0.7), while the results started to become worse when w was less or over 0.7 (based on Mean (FE) values). However, the results obtained from the ANN

training with PSO were much better than that of the ABC algorithm especially in terms of the function evaluation number (Mean (FE)) value and the time elapsed.

In order to investigate more, we performed another experiment using the recommended setting of Sharma et al. (2011) to compare with our setting obtained from the previous experiments. The results for the different parameter settings are shown in Table 8.

Table 8

*ANN Training using PSO & HPABC Algorithms (Parameter Comparison)*

| *Algorithm* | *Parameters settings* | Mean (FE) | Time elapsed (S) | | |
|---|---|---|---|---|---|
| | | | Best | Worst | Mean |
| PSO | W=0.729, c1,c2=1.49445 | 1418.3 | 1.22 | 7.70 | 3.07 |
| PSO | W=0.729, $c_1$=1.1, $c_2$=1.2 | **1143** | **1.32** | **6.32** | **2.48** |
| PSO | W=0.7, $c_1$=1.1, $c_2$=1.2 | 1453.3 | 0.69 | 12.60 | 2.95 |
| HPABC | W=0.729, $c_1$,$c_2$=1.49445 | 928 | 0.11 | 6.05 | 1.84 |
| HPABC | $W$=0.729, $c_1$=1.1, $c_2$=1.2 | **725.6** | **0.11** | **4.69** | **1.45** |
| HPABC | W=0.7, $c_1$=1.1, $c_2$=1.2 | 839.00 | 0.27 | 6.51 | 2.15 |

In Table 8, we also used the same parameter settings of the PSO for the HPABC algorithm in order to find the best settings for the HPABC. The best parameter settings for PSO and HPABC algorithms were when w=0.729 as recommended in Sharma et al. (2011), and ($c_1$=1.1 and $c_2$=1.2) based on our experimental results of ANN training on the Cancer dataset.

In conclusion, the best parameter settings of the three algorithms (ABC, PSO and HPABC) to be used for the evaluation purpose as shown in Table 9.

Table 9

*ABC, PSO and HPABC Algorithms with their Parameter Settings*

| Algorithm | Parameters settings |
|---|---|
| ABC | Colony Size = 10, limit= 300 |
| PSO | Swarm Size =10, w=0.729, c1=1.1, c2=1.2 |
| HPABC | Colony Size = 10, limit= 300, w=0.729, c1=1.1, c2=1.2 |

## EVALUATIONS AND RESULTS

In this paper we presented the HPABC evaluation process to determine whether it could achieve better results in the ANN training problem. For this evaluation, four benchmark classification datasets were used to evaluate the process.

In order to get informative and meaningful results that can clearly identify the algorithms' performance, different evaluation parameters were used, namely, Best, Worst and Mean time- elapsed parameters, SSE for training and testing accuracy measurement and Mean (FE) parameter for algorithm robustness and performance evaluation. Moreover, the mean Classification Accuracy (Mean (CA)) parameter was added in order to evaluate the ANN training performance after being trained using the compared algorithms. However in this work, Mean (CA) calculation was based on the SSE value of each Test input (30% of the benchmark dataset) in the trained ANN. If the SSE of that individual was ($SSE_x$ <= 0.00010, the stopping threshold) equal to or less than the total SSE of all the trained individuals (70% of the benchmark dataset), then it was considered as a successful classification. Then the total number of successful classifications was divided by the number of instances which indicated the overall accuracy of that trained ANN. The Mean (CA) of the total runs (30) of each of the evaluated algorithms was presented as the ANN classification accuracy performance. The evaluation results are shown in Table 10, and the best results obtained for reach dataset are in bold.

In Table 10, it is clear that the proposed HPABC algorithm has an impressive performance in the application of the ANN training process. This clearly illustrates the significance of hybridizing the advantages of the two well-known algorithms (ABC and PSO).

Table 10

*The Evaluation Results of the HPABC with Original ABC and PSO Algorithms*

| Dataset | Algorithm | Training | | | | Test | |
|---|---|---|---|---|---|---|---|
| | | Mean (FE) | Time elapsed (S) | | | Mean *SSE* | Mean (CA) |
| | | | Best | Worst | Mean | | |
| *Iris* | ABC | 70993.33 | 1.14 | 61.20 | 13.67 | 0.00010 | 98.96 |
| | PSO | 56770.66 | 0.05 | 86.37 | 12.17 | 0.0022 | 94.81 |
| | **HPABC** | **29839.06** | **0.32** | **23.40** | **6.26** | **6.666E-05** | **99.33** |
| *Cancer* | ABC | 4863.73 | 3.94 | 15.47 | 8.75 | 2.22E-05 | 99.77 |
| | PSO | 2255 | 0.71 | 96.05 | 4.34 | 0.00016 | 99.19 |
| | **HPABC** | **616.36** | **0.057** | **10.79** | **1.15** | **9.522E-06** | **99.90** |
| *Diabetes* | ABC | 4512.4 | 2.60 | 12.28 | 6.20 | 3.318E-05 | **99.66** |
| | PSO | 13734.66 | 0.73 | 96.15 | 25.60 | 0.00371 | 98.44 |
| | **HPABC** | **1272.6** | **0.033** | **14.74** | **2.15** | **3.607E-05** | 99.63 |
| *Glass* | ABC | **392359.53** | **142.37** | **338.45** | **328.23** | 0.0028 | 73.22 |
| | PSO | 400011 | 419.91 | 570.02 | 445.97 | 0.015 | 56.97 |
| | **HPABC** | 400005.06 | 422.06 | 469.20 | 427.94 | **0.0023** | **77.74** |

The results for the Iris dataset had significantly improved when using the HPABC algorithm; the classification accuracy was better, the mean function evaluation number Mean (FE) was 57.97% better than the ABC algorithm and 47.44% better than the SO algorithm. The Mean (SSE) of testing result was closer to zero which was also better than that collected from both the ABC and PSO algorithms. The Mean time elapsed on the Iris dataset in order for the ANN to reach 0.00010 SSE was highly improved by using the HPABC algorithm, where the Mean elapsed time for training was 54.21% less than ABC and 48.56% less than PSO.

On the other hand, the pattern classification results of the Cancer dataset had also improved using HPABC compared to the ABC and PSO, where the function evaluation number that the algorithm needed to reach 0.00010 SSE was 616.36 only which was very significant compared to the results of the ABC and PSO algorithms, where the results were about 87.33% better than ABC and about 72.67% better than PSO. The time elapsed for HPABC to train the ANN in order to reach the SSE stopping value was also significantly better compared to the other original algorithms, where the Best was 0.057s only, Worst was 10.79s and Mean was 1.15s, which were about 86.86% faster

than the ABC and about 73.5% faster than the PSO in terms of the Mean time values comparison. In addition the HPABC algorithm had the best mean classification accuracy value as well.

For the Diabetes dataset, the HPABC results were also significantly better than that of the original ABC and PSO algorithms in terms of mean function evaluation, (best, worst, mean) and time elapsed. The function evaluation result of the HPABC was 71.8% better than ABC and 90.73% better than PSO, and the mean time elapsed was 65.32% better than ABC and 91.6% better than PSO. The mean classification accuracy was very close to that of the ABC and better than that of PSO.

The ANN training process on the Glass dataset, also improved by using the HPABC algorithm in terms of the classification accuracy (CA) value, while for the other representative evaluation parameters the results were close to that of PSO. The ABC Mean (FE) and Time *Elapsed* results were slightly better than HPABC. However, for the *Glass* dataset the results of the Mean (FE), Time *Elapsed* and *Mean SSE* illustrate the difficulty of the ANN training process on that dataset of 6 classes with 214 records only.

In general, from the previously illustrated results the superior performance of the HPABC can be categorized into two main components: the overall time performance and the fast convergence speed with local optimal trapping avoidance. The HPABC algorithm's process time was improved due to the elimination of the Employed bee phase of the original ABC algorithm. This phase was performing the same job as the Onlooker bee phase but with no probability guidance step (randomly). Therefore, this phase was wasting the algorithm's time with no significant improvement. On the other hand, the convergence speed of the HPABC was better according to the mean function evaluation number values because the particle move of the PSO algorithm was adopted. This step was performed using the velocity calculation in Eq. (2.2 and 2.3) which enables the bees in the HPABC to benefit from their own experience though the personal best paramete *gbest* and also share their experience though the global best parameter *gbest*. These two parameters along with the velocity parameter helped the bees in the HPABC to rapidly converge into the best solution. This is similar to that of the PSO algorithm but with the penalty of being trapped at the local best (local optimal) solution, while in HPABC this was avoided by retaining the scout bee phase of the original ABC. The scout bee phase prevents the algorithm from being trapped at the local minima by providing a new random solution to the algorithm's

search space after a certain count "*limit*" of unsuccessful improving attempts. The HPABC was also compared against other hybrid algorithms such as the Enhanced Mussels Wandering Optimization E-MWO algorithm and Harmony Search with Best-to-Worst ratio (HS-BtW) as well as with the original MWO, GA and the gradient-based algorithm BP, that was used in Abusnaina et al. (2014). The results of these algorithms can be compared with HPABC because the ANNT was performed on the same four datasets used in this work. Furthermore, all the algorithms were population based (except for BP). The comparison was done based on the overall recognition accuracy (mean classification accuracy) and mean training time. The results are presented in Table 11.

Table 11

*The Evaluation Results of the HPABC with Other Algorithms*

| Dataset | Evaluation | HPABC | E- MWO | MWO | HS-BtW | GA | BP |
|---|---|---|---|---|---|---|---|
| *Iris* | Mean CA% | **99.33** | 91.0 | 89.6 | 86.8 | 84.6 | 96.6 |
| | Mean Time(s) | 6.26 | 6.0 | **5.0** | 49.0 | 10.0 | 1132.0 |
| *Cancer* | Mean CA% | **99.90** | 97.1 | 97.3 | 98.2 | 97.4 | 96.1 |
| | Mean Time(s) | **1.15** | 26.0 | 30.0 | 30.0 | 1355.0 | 3909.0 |
| *Diabetes* | Mean CA% | **99.63** | 78.0 | 74.5 | 75.3 | 73.8 | 75.4 |
| | Mean Time(s) | **2.15** | 27.1 | 26.0 | 72.7 | 392.3 | 16311.0 |
| *Glass* | Mean CA% | **77.74** | 58.7 | 49.1 | 58.8 | 45.2 | 60.1 |
| | Mean Time(s) | 427.94 | 20.0 | **18.7** | 69.1 | 740.0 | 6104.0 |

The results shown in Table 11 clearly indicate the significant performance of the HPABC compared to other algorithms in terms of mean time elapsed and classification accuracy. The hybrid particle-move ABC was superior to the state-of-art GA and BP and other hybrid algorithms as well. It is also important to note that the results of the other algorithms in Table 11 are for the same datasets but with different training and testing split percentages, where in this work the datasets were split into 70% training and 30% testing while the dataset from the reference was split into 80% training and 20% testing. This indicates that the HPABC reached significant classification results with less training data compared with the other algorithms. In addition, the more the given data (information) for the training, the more the ANN will learn, and if less data was given for testing the ANN the mean classification accuracy percentage will increase.

## CONCLUSION

In this paper the HPABC algorithm which was previously tested on numerical functions was tested on another scientific problem, which was the Artificial Neural Networks Training (ANNT) for pattern classification process. The performance of the HPABC algorithm was evaluated on four types of benchmark pattern-classification datasets. The HPABC algorithm had proved its significant performance and achieved the goal of hybridization complementing the advantages of the ABC and PSO algorithms. In this work, the HPABC showed high performance in terms of speed and classification accuracy, where the mean function evaluation Mean (FE) values were at the lowest level unlike the ABC and PSO algorithms. The HPABC algorithm was also compared with the state-of-art and other hybrid algorithms, and the results obtained illustrated that the HPABC was a competitive algorithm. Finally, it can be concluded that the HPABC algorithm can be efficiently used for Artificial Neural Networks Training. However, it still has some difficulties with datasets of few records and a high number of classes (Glass dataset). Other types of pattern-classification datasets are suggested to be used for future work as well as other real- world optimization problems that are yet to be tested on the HPABC algorithm.

## ACKNOWLEDGMENT

This research publication was funded by Universiti Sains Malaysia under the Research University Cluster Grant Scheme (RUC) for "Reconstruction of the Neural Microcircuitry or Reward Controlled Learning in the Rat Hippocampus" [1001/PKOMP/ 8630022].

## REFERENCES

Abusnaina, A. A., Abdullah, R., & Kattan, A. (2014). Enhanced MWO training algorithm to improve classification accuracy of artificial neural networks. *Recent advances on soft computing and data mining* (pp. 183-194): Springer.

Alqattan, Z. N., & Abdullah, R. (2015). A hybrid artificial bee colony algorithm for numerical function optimization. *International Journal of Modern Physics C*, 1550109.

Bahamish, H. A. A., Abdullah, R., & Salam, R. A. (2009). *Protein tertiary structure prediction using artificial bee colony algorithm.* Paper presented at the Third Asia International Conference on Modelling & Simulation, AMS'09. .

Blum, C., & Socha, K. (2005). *Training feed-forward neural networks with ant colony optimization: An application to pattern classification.* Paper presented at the Fifth International Conference on Hybrid Intelligent Systems, HIS'05.

Camargo, L. C., Correa Tissot, H., & Ramirez Pozo, A. T. (2012). *Use of backpropagation and differential evolution algorithms to training MLPs.* Paper presented at the 31st International Conference of the Chilean Computer Science Society (SCCC).

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on, 1*(1), 53-66.

Eberhart, R. C., & Kennedy, J. (1995). *A new optimizer using particle swarm theory.* Paper presented at the Proceedings of the Sixth International Symposium on Micro Machine and Human Science.

Farshidpour, S., & Keynia, F. (2012). Using artificial bee colony Algorithm for MLP Training on software defect prediction. *Oriental Journal of Computer Science & Technology, 5*(2).

Ghanem, W., & Jantan, A. (2014). Using hybrid artificial bee colony algorithm and particle swarm optimization for training feed-forward neural networks. *Journal of Theoretical and Applied Information Technology, 67*(3), 664-674.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization: Technical report-tr06 Erciyes university, engineering faculty, computer engineering department.

Karaboga, D., Akay, B., & Ozturk, C. (2007). Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. *Modeling Decisions for Artificial Intelligence* (pp. 318-329) Proceedings. Springer.

Karaboga, D., & Basturk, B. (2007). Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. *Foundations of Fuzzy Logic and Soft Computing* (pp. 789-798) Proceedings. Springer.

Karaboga, D., & Gorkemli, B. (2011). *A combinatorial artificial bee colony algorithm for traveling salesman problem.* Paper presented at the International Symposium on Innovations in Intelligent Systems and Applications (INISTA).

Karaboga, D., & Ozturk, C. (2010). Fuzzy clustering with artificial bee colony algorithm. *Scientific Research and Essays, 5*(14), 1899-1902.

Kattan, A., & Abdullah, R. (2011). Training of feed-forward neural networks for pattern-classification applications using music inspired algorithm. *Training, 9*(11).

Kattan, A., & Abdullah, R. (2013). *Training feed-forward artificial neural networks for pattern-classification using the harmony search algorithm.* Paper presented at the The Second International Conference on Digital Enterprise and Information Systems (DEIS2013).

Li, X.-l., Shao, Z.-j., & Qian, J.-x. (2002). An optimizing method based on autonomous animats: Fish-swarm algorithm. *System Engineering Theory and Practice, 22*(11), 32-38.

Lichman, K. B. a. M. (2013). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

Mahmood, Z. N., Mahmuddin, M., & Mahmood, M. N. (2012). *Protein tertiary structure prediction based on main chain angle using a hybrid bees colony optimization algorithm.* Paper presented at the International Journal of Modern Physics: Conference Series.

Montana, D. J., & Davis, L. (1989). *Training feedforward neural networks using genetic algorithms.* Paper presented at the IJCAI.

Mustaffa, Z., Yusof, Y., & Kamaruddin, S. (2013). Enhanced Abc-Lssvm for energy fuel price prediction. *Journal of Information and Communication Technology, 12*, 73-101.

Nur, A. S. (2015). Near optimal convergence of back-propagation method using harmony search. *TELKOMNIKA Indonesian Journal of Electrical Engineering, 14*(1).

Ozturk, C., & Karaboga, D. (2011). *Hybrid artificial bee colony algorithm for neural network training.* Paper presented at the IEEE Congress on Evolutionary Computation (CEC).

Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *Control Systems, IEEE, 22*(3), 52-67.

Satpathy, R. (2017). Bioinspired algorithms in solving three-dimensional protein structure prediction problems *bio-inspired computing for information retrieval applications* (pp. 316-337): IGI Global.

Sharma, T. K., Pant, M., & Bhardwaj, T. (2011). *PSO ingrained artificial bee colony algorithm for solving continuous optimization problems*. Paper presented at the IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE).

SyarifahAdilah, M., Abdullah, R., & Venkat, I. (2012). *ABC algorithm as feature selection for biomarker discovery in mass spectrometry analysis.* Paper presented at the 4th Conference on Data Mining and Optimization (DMO).

Vazquez, R. A., & Garro, B. A. (2015). Training spiking neural models using artificial bee colony. *Computational Intelligence and Neuroscience, 2015*, 18.

Yaghini, M., Khoshraftar, M. M., & Fallahi, M. (2013). A hybrid algorithm for artificial neural network training. *Engineering Applications of Artificial Intelligence, 26*(1), 293-301.

Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting, 14*(1), 35-62.