Boolean Function Classification Using Hybrid Ant Bee Colony Algorithm

Habib Shah, [1] Rozaida Ghazali,[2] Nazri Mohd Nawi[3] and Nawsher Khan[4]

Faculty of Computer Science and Information Technology
Universiti Tun Hussein Onn Malaysia (UTHM)
Parit Raja, 86400 Batu Pahat. Johor, Malaysia
[4]Universiti Malaysia Pahang, Malaysia
[1]Habibshah.uthm@gmail.com,
[2]rozaida@uthm.edu.my
[3]nazri@uthm.edu.my
[4]nawsherkhan@gmail.com

Abstract: Neural network (NN) tools are suitable for many tasks such as classification, clustering, scheduling and prediction. NN performance depends on the strategy of learning a phenomenon, the number of hidden nodes, activation function and, of course, the behavior of the data. There are many techniques used for training NN, while the social insect's techniques become more focused by researchers because of its natural behavioral processing. The Artificial Bee Colony (ABC) algorithm has produced an easy way for solving combinatorial, statistical problems and for training NN by different organized agents. The objective of training NN is to adjust the weights so that application of a set of inputs produces the desired set of outputs. Normally, NN is trained by a standard back-propagation (BP) algorithm; however, BP is too slow for many applications and trapping in a local minima problem. To recover the above gap, the hybrid technique was used for training NN here. The hybrid of natural behavior agent ant and bee techniques was used for training NN. The simulation result of a Hybrid Ant Bee Colony (HABC) was compared with, ABC, BP Levenberg-Mardquart (LM) and BP Gradient Descent (GD) learning algorithms. According to experimental results, the proposed HABC algorithm did improve the classification accuracy for the Boolean function, and prediction of volcano time-series data, which was used to train the MLP.

Keywords: swarm intelligence, ant colony optimization, artificial bee colony, hybrid ant bee colony algorithm, back propagation.

1. Introduction

Artificial neural networks (ANN), sometimes also referred to only as neural networks (NNs), are information processing systems that have certain computational corresponding properties to those that have been assumed for biological NNs. ANN shows the ability to learn from the environment in an interesting way and show significant abilities of learning, recall, generalization and adaptation in the wake of changing operating environments [1].

Neural networks are a study area of artificial intelligence (AI) where we, by inspiration from the human brain, find data structures and algorithms for learning, classification and prediction of data. Many jobs that humans complete naturally fast, such as the recognition of a familiar face, prove to be very complicated tasks for a computer when the usual programming methods are used [2]. By applying the NNs model, a program can determine by example, and create an internal arrangement of rules to classify different inputs for different targets such as classification, clustering, and prediction.

From the last decade, many results have been successfully obtained by using NNs in various research areas. NNs are nonlinear computer algorithms that learn with feedback, and can model the behavior of complicated nonlinear processes. An artificial neural network consists of a collection of interconnected processing units or nodes called neurons. These are input, output and hidden neurons. Input nodes receive input signals or values from an external source, output nodes convey the result of the neural network processing and hidden nodes make up the internal layer(s) between input and output node layers. The connections between nodes are weighted, that is, they have a value that represents the strength of the connection [3].

The accuracy of NNs depend on the selection of a few parameters such as optimal weight values, network structure, activation function, learning parameter, and input variables. An improper selection of these parameters can affect NN performance for given problems. The initialization of an optimum set of weights for a given problem improved the NNs accuracy. The selection of different sets of weight values, though outputs of the NNs are parallel to inputs. The well-known ability of NNs is to produce optimal outputs with a different random set of weight values [4].

Multilayer perceptron is a classic NN, which can provide the optimal solution by different training techniques [5]. NNs have been successfully applied to a variety of classifications and learning tasks. The success of error correction training algorithms such as BP has meant that supervised learning, where the correct outcome is known, has been the most used. In addition, although the structure of NNs is a significant contributing factor to its performance, the structure is generally heuristically chosen.

The BP is the most well-known method to train MLP by updating its weight values through feed and back-forward process via the network layers [6]. It has a high success rate in solving many complex problems, but still has some drawbacks, especially when the setting of the parameters and initial weight values are not done appropriately. Furthermore,

if the network topology is not carefully selected, the NN training can be trapped in local minima or might lead to slow convergence.

To overcome the difficulty of the BP training technique, a social insect based algorithm used for NN training such as swarm intelligence (SI), partial swarm optimization (PSO), ant colony algorithm (ACO) and the artificial bee colony (ABC) algorithm are used [7-10]. Recently, MLP trained by ABC algorithm was successfully applied to time series data prediction. IABC-MLP algorithm is used to optimize to the weight areas selection by decreasing the bee's agent before evaluation for NN training at time series data [11].

The scientists have created some hybrid techniques for training NNs such as the BP-ABC algorithm, marriage of PSO with SI, BP-ant colony algorithm and so on. These hybrid methods are robust and achieved in finding optimal weight values with low error and high accuracy. Here, the ABC and ACO algorithms are combined for finding optimal weight values with high accuracy for training Multilayer Perceptron (MLP) using classification and prediction tasks. The ABC algorithm obtains the probability techniques from ACO algorithm and the hybrid swarm-based algorithms successes in exploration and exploitations process.

This research shows the searching of optimal weights using the HABC algorithm and comparing the approach with a standard BP for the Boolean function classification and volcano time series data prediction. The proposed algorithm significantly improves the discovery of the best weight values area for MLP training.

The rest of the paper is organized as follows. A brief review to ANN is given in Section two. Section three discovers the objectives of the research. Training ANNs are given in section three. Section four covers the EA, SI, ACO, and ABC algorithm. The proposed algorithm is given in section five. Section six covers results and discussions, while the paper concludes in section seven.

2. Artificial Neural Networks

Inspired by the structure of the brain, NNs consist of a set of highly interconnected entities, called nodes or units. Each unit is designed to mimic its biological counterpart, the neuron. Each accepts a weighted set of inputs and responds with an output. Multilayer perceptron (MLP) is organized in layers of neuron and implements a feed forward processing chain. A multilayer network has two or more layers of units, with the output from one layer serving as input to the next. The layers with no external output connections are referred to as hidden layers. MLP was introduced in 1957 to solve different combinatorial problems [5]. MLP, which is also known as feed forward neural networks was first introduced for the non-linear XOR, and was then successfully applied to different combinatorial problems. MLP is mostly used for information processing and pattern recognition in the prediction of seismic activities. In this section, MLP's characteristics and interaction with the seismic signals are explained. MLP works as a universal approximation in which the input signal propagates in a forward direction. It is highly used and tested with different problems such as in time series prediction and function approximation [8-11]. The MLP is categorized into three layers: the input layer, the hidden layer and the output layer, where each layer in this order gives the input to the next. The extra layers give the structure needed to recognize non-linearly separable classes.

Figure 1 shows the architecture of MLP with hidden layers, an output layer, and input layer.

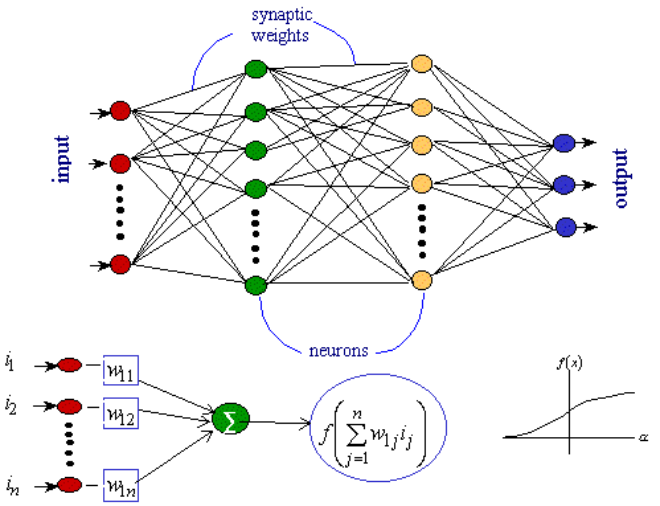

Fig 1: Multi Layer Perceptron Neural Network

$$Y_i = f_i\left(\sum_{j=1}^{n} w_{ij} x_j + b_i\right) \quad (1)$$

where y_i is the output of the node, x_i is the jth input to the node, w_{ij} is the connection weight between the input node and output node, θ_i is the threshold (or bias) of the node, and f_i is the node transfer function. Usually, the node transfer function is a nonlinear function such as a signed function, a Gaussian function, and others. The network error function E will be minimized as

$$E(w(t)) = \frac{1}{n}\sum_{j=1}^{n}\sum_{k=1}^{k} (d_k - O_t) \quad (2)$$

where E ($\boldsymbol{w}$ (t)) is the error at the tth iteration; $w(t)$ is the weights in the connections at the tth iteration; dk is the desired output node; ok is the actual value of the kth output node; K is the number of output nodes; and n is the number of patterns. T is the optimization target to minimize the objective function by optimizing the network weights $\boldsymbol{w}(t)$.

2.1 Training Artificial Neural Networks

The purpose of NN training is to produce suitable output patterns for corresponding input patterns. It is achieved by an iterative learning process that updates the NN weights. The learning can be categorized as supervised, unsupervised, and reinforcement or a hybrid of supervised and unsupervised. Supervised learning occurs when the correct output pattern is known and used during training [12-13]. The NNs can be trained by adjusting the weights of the inputs with supervised learning and unsupervised learning techniques. In this learning technique, the patterns to be recognized are known in advance, and a training set of input values is already classified with the desired output. Before commencing, the

weights are initialized with random values. Each training set is then presented for the perceptron in turn. For every input set, the output from the perceptron is compared to the desired output. If the output is not correct, then adjusting the weights on the currently active inputs toward the desired result using backward and feed forward processing will be processed by BP until there is less error or final epoch. The different learning rules form the basis of various training techniques and their applicability is dependent on the NN architecture and the learning category being used. MLP has different learning algorithms for searching optimal weight values to minimize the error term between the output of the NNs and the actual desired output value like BP [14-17]. The error term is calculated by comparing the net output to the desired output and is then fed back through the network causing the synaptic weights to be changed in an effort to minimize error. The process is repeated until the error reaches a minimum value. The most well-known supervised training algorithm for MLP is BP, which is based on the error-correction learning rule. The forward and backward passes use the same weights, but in the opposite direction. The update for a synapse depends on variables at the neurons to which it is attached. In other words, the learning rules are local, once the forward and backward passes are complete. Also, the BP algorithm is a gradient descent on the squared error cost function between the desired and actual outputs. In general, it takes O (N) operations to compute the value of the cost function, where N is the number of synaptic weights. Naively, it should take O (N2) operations to compute the N components of the gradient. In fact, the BP algorithm finds the gradient in O (N) steps, which is much shorter. So, BP is a gradient descent algorithm that updates connection weights by computing the benefit of the update in terms of minimizing output error. It requires the use of a sum squared error calculation generally given by the following equation.

$$E=\sum_{t=1}^{T}\sum_{t=1}^{n}(Y_i(t)-Z_i(t)) \qquad (3)$$

where ***T*** is the number of training patterns, n is the number of output nodes, ***Yi(t)*** and **Z(t)** are the actual and expected outputs of node **i** for pattern **t**. It has a high success rate in solving many complex problems, but still has some drawbacks especially when the setting of the parameters and initial weight values are not done appropriately. Also, if the network topology is not carefully selected, the NN training can be trapped in local minima or might lead to slow convergence.

3. Classification and Prediction Mission

Classification of data concerned with the use of computers to create a structure that learns how to automatically choose to which of a predefined set of classes a given object belongs. At present, the benchmark that appears most often in the literature is the exclusive-or problem, often called "XOR." Boolean function classification is the most important issue until today how to decide the 0 and 1 or on or off separate classes by different techniques. Exclusive OR (**XOR**) difficulty is a straightforward non-linear classification problem. These are non-linear benchmark classification tasks consisting of 2^n patterns with N inputs and 1 output [12, 18]. Each input is either a 0 or a 1 and the correct output is a 1 if the number of inputs set to 1 is odd; otherwise, it is 0. The 2-bit parity problem is also the XOR problem. The 4-bit parity problem has 4 inputs and 16 input patterns. A correct output of 1 is required for an input pattern: 1 0 1 1. The 5-bit parity problem has 5 inputs and 32 input patterns. Consider a two-dimensional surface with points (0, 1) and (1, 0) of one type and (0, 0) and (1, 1) of some other type. It is not possible to separate these two types of points using a single line on the plane.

Definition 1 (Function) If **A** and **B** are sets, a function from **A** to **B** is a rule that tells us how to find a unique **b** ∈ **B** for each **a** ∈ A. We write ***f (a) = b*** and say that f maps **a** to **b**. We also say the value of f at **a** is **b**.

We write f: **A** → **B** to indicate that ***f*** is a function from **A** to **B**. We call set A the domain of f and set **B** the range or, equivalently, codomain of ***f***. To specify a function completely you must give its domain, range and rule.

A Boolean function is a function f from the Cartesian product ×**n** {0, 1} to {0, 1}. Alternatively, we write **f: ×n{0, 1} → {0, 1}.** The set ×**n** {0, 1}, by definition, the set of all n-tuples **(x_1, · · ·, xn**) where each xi, either 0 or 1, is called the domain of f. The set {0, 1} is called the codomain (or, sometimes, range) of f. The Cartesian product ×**n** {0, 1} is also written {0, 1} n. This corresponds to writing the product of n copies of y as ***Yn***. Example 1 (tabular representation of Boolean functions) One way to represent a function, the domain of which is finite, is with a table. Each element x of the domain has a row of the table listing the domain element x and the corresponding function value f(x). For example, the two truth tables are.

Problem 1: XOR6

Table 1: XOR table of four operators

p	q	f
0	0	0
0	1	1
1	0	1
1	1	0

Problem 2: 3-bit parity problem

Inputs (0 0 0;0 0 1;0 1 0;0 1 1; 1 0 0; 1 0 1; 1 1 0; 1 1 1)

Targets (0;1;1;0;1;0;0;1)

The problem is taking the modulus 2 of summation of three inputs. In other words, if the number of binary inputs is odd, the output is 1, otherwise it is 0.

Problem 3: 4-bit encoder/decoder

Inputs (0 0 0 1;0 0 1 0;0 1 0 0;1 0 0 0)

Targets (0 0 0 1;0 0 1 0;0 1 0 0;1 0 0 0)

4-bit encoder/decoder is quite close to real world pattern classification tasks, where small changes in the input pattern cause small changes in the output pattern [19].

Without Boolean function, the time series data prediction task is also most important and the best feature of NNs. Here we will use HABC for volcano time series data. Volcanoes are the most impressive natural phenomenon among seismic

events, and people are captivated by their nature beauty as well as by their forceful eruptions. They exhibit a wide variety of eruption styles ranging from effusive eruptions typically resulting in lava flows or lava fountains, over medium-sized explosive eruptions, to large eruptions with eruption columns of several tens of kilometers in height. Aside from earthquakes, floods and storms, volcanic eruptions present the largest natural hazards, and compared to earthquakes, floods and storms. They can even influence the earth's climate [20]. Volcanic hazards are as diversified as the eruptions themselves. They can be a direct result of the volcanic activity, e.g. lava flows, ash fall, pyroclastic flows and gases, or they can be triggered by a combination of volcanic and non-volcanic processes, i.e. lahars (rain triggered mud and debris flows), landslides and tsunamis.

In this respect, neural networks provide a quick and flexible approach for data integration and model development. To date, there have been a number of research advancements taken place in the area of neural network applications. Among them are automatic classifications of seismic signals at the Mt. Vesuvius volcano in Italy and at Soufriere Hills volcano in Montserrat [21-22]. Typically, in the context of volcanic seismology, neural networks have been preferred rather than other classical statistical pattern recognition methods. The popularity of neural network models to solve pattern recognition problems has been primarily because of their seemingly low dependence on domain-specific knowledge and because of the availability of efficient learning algorithms.

NNs are information processing paradigms that are inspired by the ways in which the human brain processes information. NNs are very useful when the underlying problem is not clearly understood [14]. Their applications do not require *a priori* knowledge about the system being modeled. Furthermore, they save on data storage requirements, since it is not required to keep all past data in the memory.

4. Evolutionary Algorithms (EA)

Evolutionary algorithms are a set of approaches based on social insects, biological evolution and population-based or biological behavior of members in the environment [17, 23]. Swarm Intelligence (SI) is an issue of EA where the dynamics of the group become the individual reason for its continued existence. A common feature of population-based algorithms is that the population consisting of feasible solutions to the difficulty is customized by applying some agents on the solutions depending on the information on their robustness. Therefore, the population is encouraged toward improved solution areas of the solution space. Population-based optimization algorithms are categorized into two sections, namely, evolutionary algorithms (EA) and SI-based algorithms [17, 23]. In EA, the major plan underlying this combination is to take the weight matrices of the ANN as individuals, to change the weights by some operations such as crossover and mutation, and to use the error produced by the ANN as the fitness measure that guides selection. The efficiency and agent-based technique made researchers focus on and develop new social insect approaches for solving different problems.

The scientists have a persistent ABC algorithm because it is characterized by a honeybee behavior pattern. The ABC, which is successfully applied for several problems by extention by scientific researchers, becomes the Global Artificial Bee Colony (GABC) algorithm [24], an Improved Artificial Bee Colony (IABC) algorithm [25], the Global Hybrid Ant Bee Colony (GHABC) algorithm [26], a Hybrid Artificial Bee Colony (HABC) algorithm [27], the Hybrid Ant Bee Colony (HABC) algorithm and other types, which are the recent improvements for different mathematic, statistical and engineering problems.

4.1 Swarm Intelligence

In the last two decades, swarm intelligence (SI) has been the focus of much research because of its unique behavior inherent from social insects [28]. Bonabeau has defined the SI as "any attempt to design an algorithm or distributed problem-solving devices inspired by the collective behavior of social insect colonies and other animal societies" [29]. He mainly focused on the behavior of social insects alone such as termites, bees, wasps, and different ant species. However, the swarm can be considered any collection of interacting agents or individuals. Ants are individual agents of ACO [8]. An immune system can be considered a group of cells and molecules just as a crowd is a swarm of people [30]. PSO and ABC are popular population-based stochastic optimization algorithms adapted for the optimization of non-linear functions in multidimensional space [31]. SI-based algorithms ABC, Harmony Search and GABCS have the advantages of global optimization and easy recognition [32-33].

4.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population-based stochastic optimization method inspired by social behavior of birds flocking or fish schooling developed in 1995 [32]. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. Each particle keeps track of its coordinates in the problem spaces, which are associated with the best solution (fitness) it has achieved so far. This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called lbest. When a particle takes all of the population as its topological neighbors, the best value is a global best and is called best. The PSO method consists of, at each time step, updating the velocity of each particle toward its pbest and lbest locations. Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward pbest and lbest locations.

4.3 Artificial Bee Colony Algorithm

The Artificial Bee Colony algorithm (ABC) was proposed for optimization, classification, and NNs problem solutions based on the intelligent foraging behavior of a honeybee swarm [18]. Therefore, ABC is more successful and most robust on multimodal functions included in the set with respect to DE, PSO, and GA [6]. The ABC algorithm provides a solution in organizing form by dividing the bee objects into different tasks such as employed bees, onlooker bees, and scout bees. These three bees/tasks determine the objects of problems by sharing information to others bees. The common duties of these artificial bees are as follows:

Employed bees: Employed bees use multidirectional search space to find a food source with initialization of the area. They obtain information and all possibilities to find a food source and solution space. Sharing of information with onlooker bees is performed by employee bees. An employed bee produces a modification on the source position in her memory and discovers a new food source position. If the nectar amount of the new source is higher than that of the previous source, the employed bee memorizes the new source position and forgets the old one.

Onlooker bees: Onlooker bees evaluate the nectar amount obtained by employed bees and choose a food source depending on the probability values calculated using the fitness values. For this purpose, a fitness-based selection technique can be used. Onlooker bees watch the dance of hive bees and select the best food source according to the probability proportional to the quality of that food source.

Scout bees: Scout bees select the food source randomly without experience. If the nectar amount of a food source is higher than that of the previous source in their memory, they memorize the new position and forget the previous position. Whenever employed bees obtain a food source and use the food source very well, they again become scout bees to find new food sources by memorizing the best path. The detailed pseudo code of ABC algorithm is shown as follows:

1: Initialize the population of solutions Xi where i=1…..SN
2: Evaluate the population
3: Cycle=1
4: Repeat from step 2 to step 13
5: Produce new solutions (food source positions) $\mathbf{V}_{i,j}$ in the neighborhood of $x_{i,j}$ for the employed bees using the formula

$$\mathbf{V}_{i,j} = \mathbf{x}_{i,j} + \Phi_{ij}(\mathbf{x}_{i,j} - \mathbf{x}_{k,j}) \qquad (4)$$

where k is a solution in the neighborhood of i, Φ is a random number in the range [-1, 1] and evaluate them.
6: Apply the Greedy Selection process between processes
7: Calculate the probability values p_i for the solutions x_i by means of their fitness values by using the following formula:

$$\mathbf{p_i} = \frac{\mathbf{fit_i}}{\sum_{k=1}^{SN} \mathbf{fit_n}} \qquad (5)$$

The calculation of fitness values of solutions is defined as

$$\mathbf{fit_i} = \begin{cases} \dfrac{1}{1+f_i} & f_i >= 0 \\ 1+abs(f_i) & f_i < 0 \end{cases} \qquad (6)$$

Normalize p_i values into [0, 1]
8: Produce the new solutions (new positions) υ_i for the onlookers from the solutions x_i, selected depending on Pi, and evaluate them
9: Apply the Greedy Selection process for the onlookers between x_i and v_i
10: Determine the abandoned solution (source); if it exists, replace it with a new randomly produced solution x_i for the scout using the following equation

$$v_{ij} = x_{ij} + \Phi_{ij}(x_{ij} - x_{kj}) \qquad (7)$$

11: Memorize the best food source position (solution) achieved so far
12: cycle=cycle+1
13: **until** cycle = Maximum Cycle Number (MCN)

4.4 Ant Colony Optimization

Ant Colony Optimization (ACO) is a meta-heuristic procedure for the solution of a combinatorial optimization problem and discrete problem that has been inspired by the social insect foraging behavior of real ant decisions developed in the 1990s by M. Dorigo [19]. Real ants are capable of finding food sources through a short route through exploiting pheromone information, because ants leave pheromone on the ground, and real ants have a probabilistic preference for trajectory with a larger quantity of pheromone. The ants appear at a critical point in which they must choose to get food, whether to turn right or left. Initially, they have no information about which is the best way for getting the food source. Therefore, they choose randomly. It can be likely that, on average, half of the ants decide to turn right and the other half turn left randomly. After a short transitory period, the difference in the size of pheromone on the two paths is sufficiently large to influence the decision of new ants coming into the system. Now, in all probability, new ants will decide on the lower route, since at the decision point they recognize a greater sum of pheromone on the lower path. This, in turn, increases with a positive feedback result in the end, the number of ants choosing the lower, and shorter, route. At last, all ants will be using the shorter path for food. Thus, this process is considered a positive feedback loop, where the probability with which an ant chooses a path increases with the number of ants that previously chose the same trajectory. The ants on this path will complete the travel more times and thereby lay more pheromone over it.

Furthermore, ACO successfully applied for other tasks such as clustering, training ANN, classification and other statistical problems. When searching for food, ants initially explore the area surrounding their nest in a random manner. Real ants leave a chemical pheromone trail on the ground. As soon as an ant finds a food source, it evaluates the quantity and the quality of the food and carries some of it back to the

nest. The parameters considered here are those that affect directly or indirectly the computation of the probability in formula 7. The parameters considered here are those that affect directly or indirectly the computation of the probability in formula (7) as

$$p_k(r,s)=\begin{cases}\dfrac{[\tau(r,s)]\cdot[\eta(r,s)]^{\beta}}{\sum_{u\neq J_k(r)}[\tau(r,u)]\cdot[\eta(r,u)]^{\beta}} & if\ s\in J_k(r)\\ 0, & otherwise.\end{cases} \quad (8)$$

Where the following formula is used for the global updating rule: Once all ants have built their complete tours, pheromone is updated on all edges as follows:

$$\tau((r,s)\leftarrow(1-\rho)\cdot\tau(r,s)+\sum_{k=1}^{m}\Delta\tau_k(r,s) \quad (9)$$

where

$$\Delta\tau_k(r,s)=\begin{cases}\dfrac{1}{L_k} & \text{if } (r,s)\in \text{tour done by ant } k.\\ 0 & \text{otherwise,}\end{cases} \quad (10)$$

Where m = the number of ants, Q: a constant related to the quantity of a trail laid by ants as trail evaporation.

5. Hybrid Ant Bee Colony Algorithm

ABC and ACO were proposed for optimization, classification, and ANN problem solution based on the intelligent foraging behavior of honeybee and ant swarms [6, 18-19]. The ABC algorithm has a strong ability to find the global optimistic results optimal weight values by bee agents. It is a successfully trained ANN for classification of Boolean data, clustering and prediction of time-series data. HABC combines the ACO properties in the ABC algorithm, which may accelerate the evolving speed of ANNs and improve the classification precision of the well-trained networks. The easily understandable hybrid algorithm, using an ABC algorithm to search the optimal combination of all of the network parameters, and ACO was used for selection of the best food source to find the accurate value of each parameter. The HABC algorithm provides a solution in an organized form by dividing the agents into different tasks such as, employed bees, onlooker ants and scout bees.

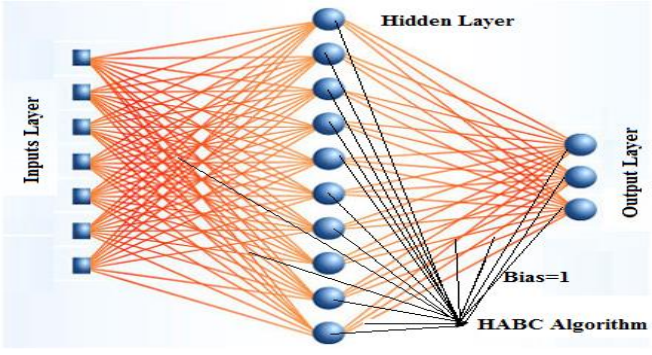

Figure 1 HABC training for MLP

The detailed pseudo code of HABC algorithm is shown as follows:

1: Load colony size and food number

2:

$$FN=\frac{SN}{2} \quad (11)$$

King Bee {

If $FN= SN \% 2 \neq 0$ (12)

Then

$$FN=\frac{SN+1}{2}$$

3: Initialize of solutions Xi

4: Evaluate the fitness fi of the population

5: cycle =1

6: Repeat

7: Produce a new solution vi by using equation

$$V_{i,j}=x_{i,j}+\Phi_{ij}(x_{i,j}-x_{k,j}) \quad (13)$$

Calculate the value τ_i

$$\tau_{(i)}=\begin{cases}\tau\geq 0 & for\ \dfrac{Q}{1+f_i}\quad else\\ \tau<0 & 1+abs(f_i)\end{cases} \quad (14)$$

Where

$$Q=\frac{1+FN}{\sum_{i=1}^{SN}(E+O)} \quad (15)$$

8: Apply greedy selection process*}*

Calculate the probability values $P(i)$ for the solutions (xi) by

$$P(i)=\frac{(0.09)*\tau(i)}{\sum_{1\leq j\leq m}^{SN}\tau(i)} \quad (16)$$

9: FOR each onlooker ant *{*

Select a solution *xi* depending on *p(i)*

Produce new solution vi

Calculate the value $\tau_{(i)}$ by eq (13)

Apply greedy selection process *}*

10: Continue ABC from step 10 to 14.

where xi represents a solution, the fitness solution o, vi indicates a neighbor solution of xi, and pi value of xi. Also, τ represents the fitness solution of trial i, which is not improved and j represents the improved solution. In the algorithm, the first half of the colony consists of employed bees, and the second half constitutes the onlooker bees. The Scout bee will decide the best values between onlookers and employed bees. In the HABC algorithm, the position of a food source represents a possible solution to the optimization problem, and the nectar total of a food source corresponds to the fitness solution of the associated solution by the king bee. The king bee initialized the colony size for employed and onlooker bees. The number of food sources equals half of the colony size and after division of employed and onlooker bees, they will start searching for and finding the optimal

food source. After initialization, the population of the positions (solutions) is subjected to repeated cycles, C = 1, 2... Maximum cycle number (MCN), of the search processes of the employed bees, the onlooker bees and scout bees. An employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) and tests the nectar total (accurate solution) of the new source (new solution). The king bee will gather employed bees and onlooker bees for the decision of a fitness solution. If the nectar sum of the new one is higher than that of the previous one, the employed bee memorizes the new position and forgets the old one. Otherwise, she keeps the position of the previous one in her memory. After all employed bees around complete the search process, they combine the nectar information of the food sources and their position information with the onlooker bees on the food's position. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar quantity. As in the case of the employed bee, she produces a modification on the position in her memory and checks the nectar quantity of the candidate source. Providing that its nectar is higher than that of the previous one, the bee memorizes the new position and forgets the old one. The onlooker bee chooses a food source depending on the probability value associated with that food source, *pi*, calculated by the eq (16).

King Bee: The king bee initialized the colony size for employed and onlooker bees. The food source will be divided in the same quantity. The king bee will update the food source for equal division to employed and onlooker bees. The number of food sources equals half of the colony size and after division on employed and onlooker bees, they will start searching for and finding the optimal food source.

Employed Bees: Employed bees use multidirectional search space for the food source with initialization of the area. They obtain information and all possibilities to find a food source and solution space. The sharing of information with onlooker bees is performed by employee bees. An employed bee produces a modification on the source position in her memory and discovers a new food source position. If the nectar quantity of the new source is higher than that of the previous source, the employed bee memorizes the new source position and forgets the old one.

Onlooker Bees: Onlooker bees evaluate the nectar quantity obtained by employed bees and choose a food source depending on the probability values calculated using the fitness values. For this purpose, a fitness-based selection technique can be used. Onlooker bees watch the dance of hive bees and select the best food source according to the probability proportional to the quality of that food source.

Scout bees: These agents are the last bees who select the food source randomly without experience. If the nectar amount of a food source is higher than that of the previous source in their memory, they memorize the new position and forget the previous position. Whenever employed ants get a food source and use the food source very well, they again become scout bees to find a new food source by memorizing the best path.

6. Simulation Results and Discussions

To evaluate the performance of the proposed HABC for training the MLP on XOR classification, 3-bit parity problem and 4-bit encoder/decoder problem are used, which are standard problems used in training neural networks. Simulation experiments were performed on a 1.66 GHz Core 2 Duo Intel Workstation with 1GB RAM using Matlab. The comparison of three learning algorithms: the standard BP (GD), BP (LM), GA and HABC are discussed based on the extensive simulations for Boolean function experiments. The learning rate for BP was set to 0.8. The weight values for MLP-HABC were initialized between [100, -100], while those of BP were initialized randomly between [−1, 1]. The stopping criterion used is when MSE ≤ 0.01 for BP, while HABC was stopped after the maximum cycle number. In this experiment, the roulette wheel selection scheme and single point crossover with the rate of 0.8, uniform mutations with the rate of 0.05 are employed for the genetic algorithm (GA). The generation gap is set to 0.9. The population size in GA was 50 for all problems. It is a complicated classification problem, which maps two, three and four binary inputs to a single binary output. To obtain reliable average results, five trials were performed during the training of the network. Each case and run started with a different number of parameters and with a random population of foods. The sigmoid function was used as the activation function for all network outputs. Finally, the mean squared error (MSE) for training was calculated for all algorithms. The network parameters setting; network topology, dimension, weight range, MCN, Objective Function Evaluation (OFE), and the number of maximum epochs in Table1.

Table 2: Setting of Network Parameters for MLP trained HABC and BP algorithms.

Network Parameter Problem	MLP Topology	Weight Range	MCN	OFE	Epoch
XOR 6	2-2-1	[-100,100]	500	75000	3500
3-Bit Parity	3-2-1	[-10,10]	500	100	3500
4-Bit Encoder/Decoder	3-2-1	[-10,10]	500	100	3000

Simulation results:
For XOR6 problem

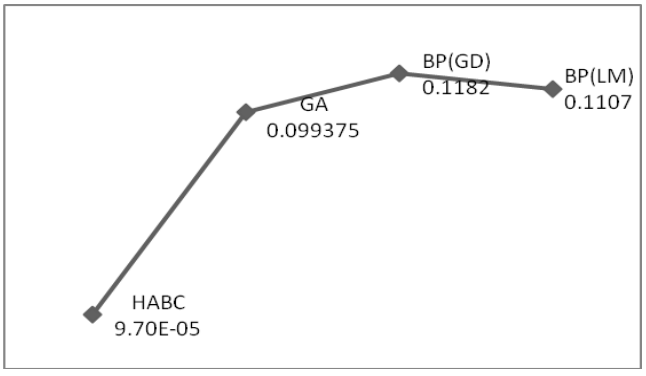

Figure 2. Average Mean Square Error by training BP, GA and HABC

For 3-bit parity problem

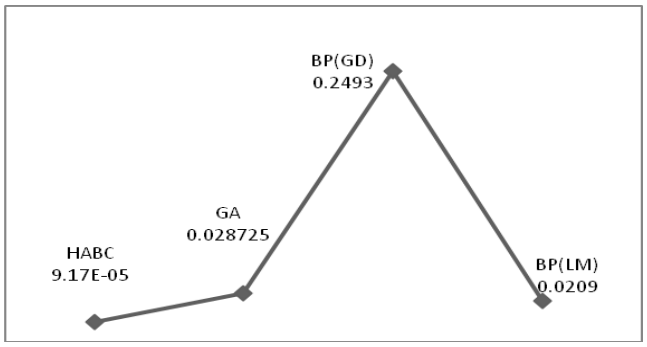

Figure 3. Average mean square error by training BP, GA and HABC

For 4-bit encoder/decoder problem,

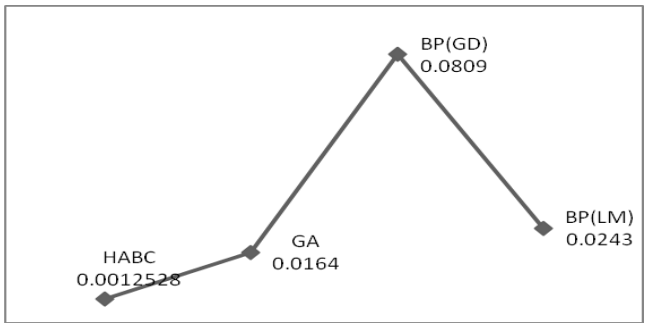

Figure 4. Average mean square error by training BP, GA and HABC

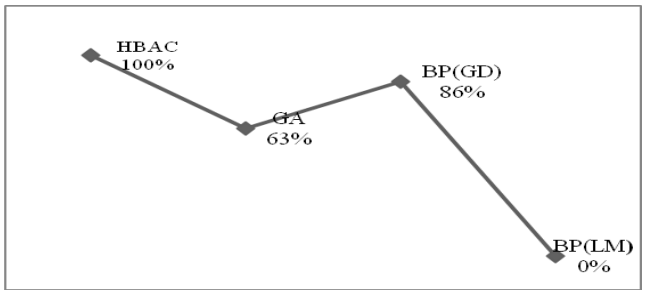

Figure 5. Success rate BP, GA and HABC for 3-bit parity problem

Here HABC algorithm has 100% success with MCN 400 to 500, while BP (GD) algorithm has 2%, 0% and 3% success with epoch 3000 to 3500 for XOR6, where BP (LM) algorithm has 73%, 86% and 6% success with epoch 3000 to 3500 and the GA algorithm has 86%, 63.333% and 0% success for XOR6, 3-bit parity and 4-bit encoder-decoder problems.

Table 3: Testing MSE for BP (GD, LM), GA and HABC

Algorithm	XOR 6	3-Bit Parity	4-Bit Encoder/Decoder
BP (GD)	0.0988314	0.110321	0.048921326
BP(LM)	0.0830923	0.0102821	0.013874632
GA	0.0812334	0.0117232	0.00188325
HABC	9.00E-05	8.17E-05	0.0011783

The optimal weights values were tested by the same problem for finding the efficiency of proposed learning algorithm. The testing MSE of HABC has minimum error compared to BP and GA algorithms.

The proposed HABC was capable of finding the desired network output in each run. Consequently, HABC outperforms other algorithms on classification problems considered in this work for the same evaluation number and can consistently find the optimal weight values for the networks. To find optimum outcomes for multimodal problems, the hybrid ant and bee strategy must combine the exploratory and exploitative components efficiently. Here HABC obtained the exploration process's process from ants while the exploitation processes were provided by the bee strategy. The simulation results showed the effectiveness and efficiency of the HABC algorithm. The use of HABC of XOR classification showed that the proposed algorithm provides promising tools for training ANNs. It can be seen in figures 2, 3, 4 and 5, that HABC algorithm classification follows the actual trend during the training phases. Generally, the HABC algorithm has shown 100% accuracy for Boolean data, which is significantly higher than the BP and GA algorithm. The time series data prediction is the feature work using the HABC algorithm.

7. Conclusion

The exploration and exploitation processes successfully combined by HABC. The HABC algorithm used for training the NNs model using the Boolean function classification problem. We hope it will be successfully applied to the time series data prediction method, which is our next research project. It has the powerful ability of searching optimal weight values in defining a colony. The performance of HABC is compared with the traditional BP algorithm; Where HABC shows significantly higher results than BP during experimentation.

Acknowledgement

The authors would like to thank University Tun Hussein Onn Malaysia for supporting this research under the Postgraduate Incentive Research Grant Vote No 0731l9.

References

[1] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biology*, vol. 5, pp. 115-133, 1943.

[2] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," in *Neurocomputing: foundations of research*, ed: MIT Press, 1988, pp. 15-27.

[3] N. Nawi, *et al.*, "The Development of Improved Back-Propagation Neural Networks Algorithm for Predicting Patients with Heart Disease Information Computing and Applications." vol. 6377, R. Zhu, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2010, pp. 317-324.

[4] N. M. Nawi, *et al.*, "An Improved Back Propagation Neural Network Algorithm on Classification Problems Database Theory and Application, Bio-Science and Bio-Technology." vol. 118, Y. Zhang, *et al.*, Eds., ed: Springer Berlin Heidelberg, 2010, pp. 177-188.

[5] F. Rosenblatt, "The Perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, pp. 386-408, 1958.

[6] D. E. Rumelhart, *et al.*, *Parallel distributed processing: Psychological and biological models*: MIT Press, 1986.

[7] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 1995, pp. 1942-1948 vol.4.

[8] M. Dorigo and G. Di Caro, "Ant colony optimization: a new meta-heuristic," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, 1999, p. 1477 Vol. 2.

[9] D. Karaboga and A. Kalinli, "Training recurrent neural networks for dynamic system identification using parallel tabu search algorithm," in *Intelligent Control, 1997. Proceedings of the 1997 IEEE International Symposium on*, 1997, pp. 113-118.

[10] D. Karaboga, *et al.*, "Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks," in *Modeling Decisions for Artificial Intelligence*. vol. 4617, V. Torra, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2007, pp. 318-329.

[11] D. Karaboga and B. Akay, "A comparative study of Artificial Bee Colony algorithm," *Applied Mathematics and Computation*, vol. 214, pp. 108-132, 2009.

[12] M. Fionn, "Multilayer perceptrons for classification and regression," *Neurocomputing*, vol. 2, pp. 183-197, 1991.

[13] S. Ghosh-Dastidar and H. Adeli, "A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection," *Neural Networks*, vol. 22, pp. 1419-1431, 2009.

[14] B. Curry and D. E. Rumelhart, "MSnet: A Neural Network which Classifies Mass Spectra," *Tetrahedron Computer Methodology*, vol. 3, pp. 213-237, 1990.

[15] J. Leonard and M. A. Kramer, "Improvement of the backpropagation algorithm for training neural networks," *Computers & Chemical Engineering*, vol. 14, pp. 337-341, 1990.

[16] M. Gori and A. Tesi, "On the problem of local minima in backpropagation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 14, pp. 76-86, 1992.

[17] S. H. Leung, *et al.*, "A weight evolution algorithm for multi-layered network," in *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, 1994, pp. 892-896 vol.2.

[18] D. G. Stork and J. D. Allen, "How to solve the N-bit parity problem with two hidden units," *Neural Networks*, vol. 5, pp. 923-926, 1992.

[19] C. Gail A, "Neural network models for pattern recognition and associative memory," *Neural Networks*, vol. 2, pp. 243-257, 1989.

[20] A. A. Suratgar, *et al.*, "Magnitude of Earthquake Prediction Using Neural Network," in *Natural Computation, 2008. ICNC '08. Fourth International Conference on*, 2008, pp. 448-452.

[21] G. Muscato, *et al.*, "Volcanic Environments: Robots for Exploration and Measurement," *Robotics & Automation Magazine, IEEE*, vol. 19, pp. 40-49, 2012.

[22] E. Sansosti, *et al.*, "Dynamic deformation of Etna volcano observed by satellite radar interferometry," in *Geoscience and Remote Sensing Symposium Proceedings, 1998. IGARSS '98. 1998 IEEE International*, 1998, pp. 1370-1372 vol.3.

[23] S. Kiranyaz, *et al.*, "Evolutionary artificial neural networks by multi-dimensional particle swarm optimization," *Neural Networks*, vol. 22, pp. 1448-1462, 2009.

[24] G. Peng, *et al.*, "Global artificial bee colony search algorithm for numerical function optimization," in *Natural Computation (ICNC), 2011 Seventh International Conference on*, 2011, pp. 1280-1283.

[25] H. Shah and R. Ghazali, "Prediction of Earthquake Magnitude by an Improved ABC-MLP," in *Developments in E-systems Engineering (DeSE), 2011*, 2011, pp. 312-317.

[26] H. SHah, *et al.*, "Global Hybrid Ant Bee Colony Algorithm for Training Artificial Neural Networks,," presented at the International Conference on Computational Science and Applications, Brazil 2012.
H. Shah, *et al.*, "Hybrid Ant Bee Colony Algorithm for Volcano Temperature PredictionEmerging Trends and Applications in Information Communication Technologies." vol. 281, B. S. Chowdhry, *et al.*, Eds., ed: Springer Berlin Heidelberg, 2012, pp. 453-465.

[27] R. Akbari, *et al.*, "A powerful bee swarm optimization algorithm," in *Multitopic Conference, 2009. INMIC 2009. IEEE 13th International*, 2009, pp. 1-6.

[28] E. Bonabeau, *et al.*, "Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, NY,," 1999.

[29] G. Tao, "Artificial immune system based on normal model and immune learning," in *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, 2008, pp. 1320-1325.

[30] D. Karaboga and B. Gorkemli, "A combinatorial Artificial Bee Colony algorithm for traveling salesman problem," in *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on*, 2011, pp. 50-53.

[31] L. Hong-Bo, *et al.*, "Neural networks learning using vbest model particle swarm optimisation," in *Machine Learning and Cybernetics, 2004. Proceedings of 2004*

International Conference on, 2004, pp. 3157-3159 vol. 5.

[32] Das, S.; Mukhopadhyay, A.; Roy, A.; Abraham, A.; Panigrahi, B.K., "*Exploratory Power of the Harmony Search Algorithm: Analysis and Improvements for Global Numerical Optimization*," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* , vol.41, no.1, pp.89,106, Feb. 2011

[33] B. Debarati Kundu, Kaushik Suresh, Sayan Ghosh, Swagatam Das and Ajith Abraham, *Data Clustering Using Multi-objective Differential Evolution Algorithms, Fundamenta Informaticae Journal, IOS Press, Netherlands,* Volume 97, Number 4, pp. 381-403, 2009.