7

Firefly Meta-Heuristic Algorithm for Training the Radial Basis Function Network for Data Classification and Disease Diagnosis

Ming-Huwi Horng[1], Yun-Xiang Lee[2], Ming-Chi Lee[1] and Ren-Jean Liou[3]
[1]*Department of Computer Science and Information Engineering, National Pingtung Institute of Commerce*
[2]*Department of Computer Science and Information Engineering, National Cheng Kung University*
[3]*Department of Computer and Communication, National Pingtung Institute of Commerce*
Taiwan

1. Introduction

The radial basis function (RBF) network is a type of neural network that uses a radial basis function as its activation function (Ou, Oyang & Chen, 2005). Because of the better approximation capabilities, simpler network structure and faster learning speed, the RBF networks have attracted considerable attention in many science and engineering field. Horng (2010) used the RBF for multiple classifications of supraspinatus ultrasonic images. Korurek & Dogan (2010) used the RBF networks for ECG beat classifications. Wu, Warwick, Jonathan, Burgess, Pan & Aziz (2010) applied the RBF networks for prediction of Parkinson's disease tremor onset. Feng & Chou (2011) use the RBF network for prediction of the financial time series data. In spite of the fact that the RBF network can effectively be applied, however, the number of neurons in the hidden layer of RBF network always affects the network complexity and the generalizing capabilities of the network. If the number of neurons of the hidden layer is insufficient, the learning of RBF network fails to correct convergence, however, the neuron number is too high, the resulting over-learning situation may occur. Furthermore, the position of center of the each neuron of hidden layer and the spread parameter of its activation function also affect the network performance considerably. The determination of three parameters that are the number of neuron, the center position of each neuron and its spread parameter of activation function in the hidden layer is very important.

Several algorithms had been proposed to train the parameters of the RBF network for classification. The gradient descent (GD) algorithm (Karayiannis, 1999) is the most popular method for training the RBF network. It is a derivative based optimization algorithm that is used to search for the local minimum of a function. The algorithm takes steps proportional to negative of the gradient of function at the current situation. Many global optimization methods had been proposed to evolve the RBF networks. The genetic algorithm is a popular method for finding approximate solutions to optimization and search problems. Three genetic

operations that are selection, crossover and mutation, of the main aspects of GA evolve the optimal solution form an initial population. Barreto, Barbosa & Ebecken (2002) used the real-code genetic algorithm to decide the centers of hidden neurons, spread and bias parameters by minimizing the mean square error of the desired outputs and actual outputs. The particle swarm optimization is a swarm intelligence technique, first introduced by Kennedy & Eberhart (2007), inspired by the social behavior of bird flocks or fish schools. The computation of the PSO algorithm is dependent on the particle's local best solution (up to the point of evaluation) and the swarm's global best solution. Every particle has a fitness value, which is evaluated by the fitness function for optimization, and a velocity which directs the trajectory of the particle. Feng, (2006) designed the parameters of centers, the spread of each radial basis function and the connection weights as the particle, and then applied the PSO algorithm to search for the optimal solution for constructing the RBF network for classification. Kurban & Besdok, (2009) proposed an algorithm by using artificial bee colony algorithm to estimate the weights, spread, bias and center parameters based on the algorithm. This chapter concluded the ABC algorithm is superior to the GA, PSO and GD algorithms.

The firefly algorithm is a new swarm-based approach for optimization, in which the search algorithm is inspired by social behavior of fireflies and the phenomenon of bioluminescent communication. There are two important issues in the firefly algorithm that are the variation of light intensity and formulation of attractiveness. Yang (2008) that simplifies the attractiveness of a firefly is determined by its brightness which in turn is associated with the encoded objective function. The attractiveness is proportional to their brightness. Furthermore, every member x_i of the firefly swarm is characterized by its bright I_i which can be directly expressed as an inverse of a cost function for a minimization problem. Lukasik & Zak (2009) applied the firefly algorithm for continuous constrained optimization. Yang (2010) compared the firefly algorithm with the other meta-heuristic algorithms such as genetic and particle swarm optimization algorithms in the multimodal optimization. These works had the same conclusions that the algorithm applied the proposed firefly algorithm is superior to the two existing meta-heuristic algorithms.

In this chapter, a firefly algorithm of the training of the RBF network is introduced and the performance of the proposed firefly algorithm is compared with the conventional algorithms such as conventional GD, GA, PSO and ABC algorithms on classification problems from the UCI repository. Furthermore, the receiver operating characteristic analysis is used to evaluate the diagnosis performance of medical datasets. Some conclusions are made in the final section.

2. Radial basis function network

The neural network are non-linear statistical data modeling tools and can be used to model complex relationships between inputs and outputs or to find patterns in a dataset. The radial basis function network is a popular type of network that is very useful for pattern classification (Bishop, 1995). A radial basis function (RBF) network can be considered a special three-layered network shown in Fig 1.

The input nodes pass the input values x to the internal nodes that construct the hidden layer. Each unit of hidden layer implements a specific activation function called radial basis function. The nonlinear responses of hidden nodes are weighted in order to calculate the

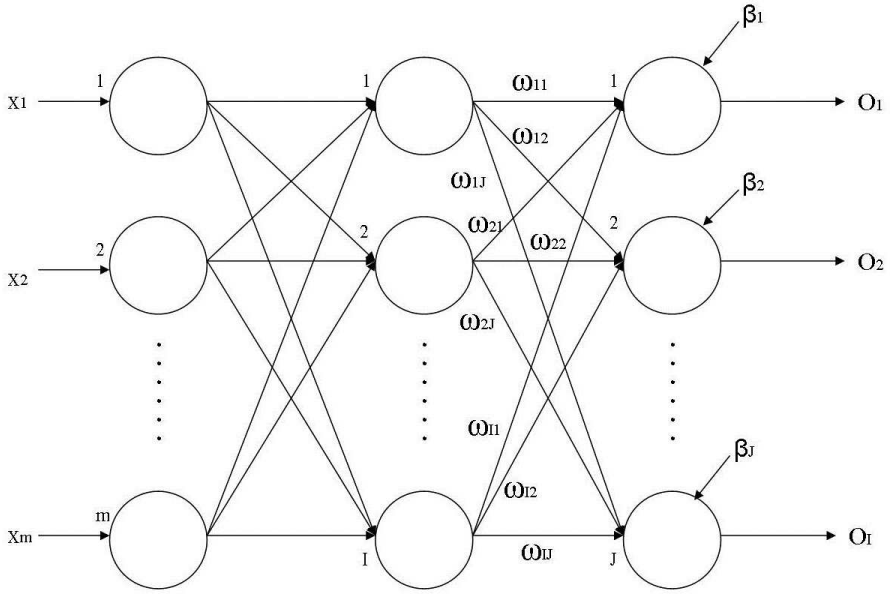


Fig. 1. The structure of radial basis function network

final outputs of network in the output layer. The input layer of this network has m units for m dimensional input vectors. The input units are fully connected to I hidden layer units, which are in turn fully connected to the J output layer units, where J is the number of output layer. Each neuron of the hidden layer has a parameter mean vector called center. Figure 1 shows the detailed structure of an RBF network. Each input data x with m dimensions, $x=(x_1, x_2, ..., x_m)$, are located in the input layer, which broadcast to hidden layer. The hidden layer has I neurons and each neuron compute the distance between the centers and the inputs. Each activation function of the neuron in hidden layer is chosen to be Gaussians and is characterized by their mean vectors c_i and its spread parameter α_i ($i=1,2,\ldots,I$). That is, the activation function $\phi(x)$ of the i^{th} hidden unit for an input vector **x** is given by:

$$\phi_i(x) = exp[-\alpha_i \cdot \|x-c_i\|^2] \tag{1}$$

The ϕ_i affects the smoothness of the mapping, thus, the output value of the neuron j of output layer $\overline{y_j}$ for training sample x, are given by $o(x)$ in (2).

$$\begin{aligned} o(x_i) &= (o_1, o_2, ..., o_J) \\ o_j &= \sum_{h=1}^{I} w_{hj}\phi_i(x_i) + \beta_j \end{aligned} \tag{2}$$

The weights, w_{ij} ($i=1,2,\ldots,I.$, $j=1,2,\ldots,J$), is the i-th node of output of hidden layer that transmitted to j-th node of the output layer, and β_j is the bias parameter of the j-th node of

output layer determined by the RBF network training procedure. In practice, the training procedure of RBF is to find the adequate parameters w_{ij} , α_i , β_i and c_i such that the error metrics such as the mean square error (MSE) is minimum.

$$MSE(\mathrm{w}, \alpha, \beta, \mathrm{c}) = \frac{1}{N}\sum_{k=1}^{N} \left\| d(x_k) - o(x_k) \right\|^2 \tag{3}$$

where $d(x_i)$ and $o(x_i)$ is denoted to the desired output vector and actual output vector for training sample x_i . In (3), the N is the number of the training samples.

3. Training algorithms: GD, GA, PSO, ABC and FA

This section gives brief descriptions of training algorithms of RBF network that include the gradient descent algorithm (GD), the genetic algorithm (GA), the particle swarm optimization (PSO) algorithm and the artificial colony bee (ABC) algorithm.

3.1 Gradient Descent (GD) algorithm

GD is the derivative based optimization algorithm (Karayiannis, 1999)that is used to search for the local minimum of a function. The algorithm takes steps proportional to negative of the gradient of function at the current situation with given the parameters α_i and assumed all β_i are equal to 0. In general, the output of a RBF network can be written in the following form.

$$\overline{O} = (o_1, o_2, ..., o_J) = \begin{bmatrix} w_{11} & w_{12} & . & . & w_{1J} \\ w_{21} & w_{22} & . & . & w_{2J} \\ . & . & . & . & . \\ . & . & . & . & . \\ w_{I1} & w_{I2} & . & . & w_{IJ} \end{bmatrix} \cdot \begin{bmatrix} exp[-\alpha_1 \cdot \left\| \mathrm{x}\text{-}c_1 \right\|^2] \\ exp[-\alpha_2 \cdot \left\| \mathrm{x}\text{-}c_2 \right\|^2] \\ \\ \\ exp[-\alpha_I \cdot \left\| \mathrm{x}\text{-}c_I \right\|^2] \end{bmatrix} \tag{4}$$

and

$$O = \mathrm{W} \cdot \mathrm{H} \tag{5}$$

where the weight matrix is represented as W and the ϕ matrix is the H matrix, respectively. The GD algorithm can be implemented to minimize the MSE term defined as the equation (3) based on the following equations.

$$w_{ij} = w_{ij} - \eta \frac{\partial MSE}{\partial w_{ij}} \tag{6}$$

$$c_i = c_i - \eta \frac{\partial MSE}{\partial c_i} \tag{7}$$

where the η is the parameter of learning rate.

3.2 Genetic Algorithm (GA)

Genetic algorithm (Goldberg, 1989) inspired by the evolutionary biology is a popular method for finding approximate solutions to optimization and search problems. In the genetic algorithm, a population of strings called chromosomes which encode candidate solutions to an optimization problem, evolves toward better solutions. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population based on their fitness, and modified by recombined and possibly randomly mutated to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached. The three genetic operations that are selection, crossover and mutation, of the main aspects of GA evolve the optimal solution form an initial population. Barreto, Barbosa & Ebecken (2002) used the real-code genetic algorithm to decide the centers of hidden neurons, spread and bias parameters by minimizing the MSE of the desired outputs and actual outputs.

3.3 Particle Swarm Optimization (PSO) algorithm

The particle swarm optimization (PSO) first introduced by Kennedy & Eberhart (1995), is a swarm optimization method that optimizes a problem by iteratively trying to improve candidate solutions called particles. The improvement of candidate particles with D dimension in the PSO algorithm is dependent on the particle's local best solution, $P_l^t = (p_{l1}^t, p_{l2}^t, ..., p_{lD}^t)$ (up to the point of evaluation) and the swarm's global best solution $p_g^t = (p_{g1}^t, p_{g2}^t, ..., p_{gD}^t)$ at the iteration t. Every particle has a fitness value, which is evaluated by the fitness function for optimization, and a velocity which directs the trajectory of the particle. The D-dimensional position for particle i can be at the iteration t represented as $x_i^t = (x_{i1}^t, x_{i2}^t, ..., x_{iD}^t)$. Like to the position, the velocity of particle i can be described as $v_i^t = (v_{i1}^t, v_{i2}^t, ..., v_{iD}^t)$. The movements of particles i at the $t+1$ iteration are followed as the Eq. [8] and [9].

$$v_{id}^t = v_{id}^{t-1} + c_1 r_1 (P_{id}^t - x_{id}^t) + c_2 r_2 (p_{gd}^t - x_{id}^t) \quad d = 1, 2, .., D \tag{8}$$

$$x_{id}^t = x_{id}^{t-1} + v_{id}^t \qquad d = 1, 2, .., D \tag{9}$$

where c_1 indicates the cognition learning factor; c_2 indicates the social learning factor, and r_1 and r_2 are random numbers between (0, 1). Feng (2006) designed the parameters of centers, the spread of each radial basis function and the connection weights as the particle, and then applied the PSO algorithm to search for the optimal solution for constructing the RBF network for classification.

3.4 Artificial Bee Colony (ABC) algorithm

The artificial bee colony (ABC) algorithm was proposed by the Kurban and Besdok, (2009) applied it to train the RBF network. In the ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. The employed bees bring loads of nectar from the food resource to the hive and may share the information about food source in the dancing area. These bees carry information about food sources and share them with a certain probability by dancing in a dancing area in the hive. The onlooker bees wait in the dances area for making a decision on the selection of a food source depending on the probability delivered by employed bees. The computation of probability is based on the amounts of the food source. The other kind of bee is scout bee that carries out random searches for new food sources. The employed bee of an abandoned food source becomes a scout and as soon as it finds a new food source it becomes employed again. In other words, the each search cycle of the ABC algorithm contains three steps. First, the employed bees are sent into their food sources and the amounts of nectar are evaluated. After sharing this information about the nectar, onlooker bees select the food source regions and evaluating the amount of nectar in the food sources. The scout bees and then chosen and sent out to find the new food sources.

In the ABC algorithm, the position of a food source z_i represents a possible solution to the optimization problems and the amount of nectar in a food source corresponds to the fitness $fit(z_i)$ of the corresponding solution z_i. In the training RBF network, a solution z_i is made up of the parameters of weights, spread, bias and vector centers of RBF network. The number of employed or onlooker bees is generally equal to the number of solutions in the population of solutions. Initially, the ABC algorithm randomly produced a distributed initial population P of SN solutions, where SN denoted the number of employed bees or onlooker bees. Each solution z_i ($i=1,2,\ldots,SN$) is a D-dimensional vector. Here D is the number of optimization parameters. In each execution cycle, C ($C=1, 2,\ldots, MCN$), the population of the solutions is subjected to the search processes of the employed, the onlooker and scout bees. An employed bee modifies the possible solution depending on the amount of nectar (fitness vale) of the new source (new solution) by using the Eq. (10).

$$z_{ij} = z_{ij} + rand(-1,1)(z_{ij} - z_{kj}) \tag{10}$$

Where $k \in \{1,2,.....,SN\}$ but $k \neq i$ and $j \in \{1,2,....,D\}$ are randomly selected indexes. $rand(a, b)$ is a random number between $[a, b]$.

If there is more nectar in new solution is than that in the precious one, the bee remembers the new position and forgets the old one, otherwise it retains the location of the previous one. When all employed bees have finished this search process, they deliver the nectar information and the position of the food sources to the onlooker bees, each of whom chooses a food source according to a probability proportional to the amount of nectar in that food source. The probability p_i of selecting a food source z_i is determined using the following Eq. (11).

$$p_i = \frac{fit(z_i)}{\sum_{i=1}^{SN} fit(z_i)} \tag{11}$$

In practical terms, any food source z_i,(i=1,2,…,SN) sequentially generates a random number between [0, 1] and if this number is less than p_i, an onlooker bee are sent to food source z_i and produces a new solution based on the equation (9). If the fitness of the new solution is more than the old one, the onlooker memorizes the new solution and shares this information with other onlooker bees. Otherwise, the new solution will be discarded. The process is repeated until all onlookers have been distributed to the food sources and produces the corresponding new solution.

If the position of food source can not be improved through the predetermined number of "*limit*' of bees, then the food resource z_i is abandoned and then the employed bee becomes a scout. Assume that the abandoned source is z_i and $j \in \{1,2,....,D\}$, then the scout discovers a new food source to be replaced with z_i. This operation can be defined as in (12).

$$z_{ij} = z_{min}^{j} + rand(0,1)(z_{max}^{j} - z_{min}^{j}) \tag{12}$$

where the z_{min}^{j} and z_{max}^{j} are the upper bound and upper bound of the j-th component of all solutions. If the new solution is better than the abandoned one, the scout will become an employed bee. The selection of employed bees, onlooker bees and scouts is repeated until the termination criteria have been satisfied.

3.5 Firefly Algorithm

Firefly algorithm (FA) was developed by Xin-She Yang at Cambridge University in 2008. In the firefly algorithm, there are three idealized rules: (1) all fireflies are unisex so that one firefly will be attracted to other fireflies regardless of their sex; (2) Attractiveness is proportional to their brightness, thus for any two flashing fireflies, the less brighter one will move towards the brighter one. If there is no brighter one than a particular firefly, it will move randomly. As firefly attractiveness one should select any monotonically decreasing function of the distance $r_{i,j} = d(x_j, x_i)$ to the chosen j-th firefly, e.g. the exponential function.

$$r_{i,j} = \left\| x_i - x_j \right\| \tag{13}$$

$$\beta \leftarrow \beta_0 e^{-\gamma r_{i,j}} \tag{14}$$

where the β_0 is the attractiveness at $r_{i,j} = 0$ and γ is the light absorption coefficient at the source.

The movement of a firefly i is attracted to another more attractive firefly j is determined by

$$x_{i,k} \leftarrow (1-\beta)x_{i,k} + \beta x_{j,k} + u_{i,k} \tag{15}$$

$$u_{i,k} = \sigma(rand1 - \frac{1}{2}) \tag{16}$$

The particular firefly x_i with maximum fitness will move randomly according to the following equation.

$$x_{i^{max},k} \leftarrow x_{i^{max},k} + u_{i^{max},k}, \text{ for k=1,2,…,c}$$

$$u_{i^{max},k} = \sigma(rand2 - \frac{1}{2}) \tag{17}$$

when $rand1$, $rand2$ are random vector whose each element obtained from the uniform distribution range from 0 to 1; (3). The brightness of a firefly is affected or determined by the landscape of the fitness function. For maximization problem, the brightness I of a firefly at a particular location x can be chosen as $I(x)$ that is proportional to the value of the fitness function.

4. Training RBF network using firefly algorithm

The individuals of the fireflies include the parameters of weights (w), spread parameters (α), center vector (c) and the bias parameters (β). The mean vector c_i of the i-th neuron of hidden layers is defined by $c_i = (c_{i1}, c_{i2}, ..., c_{im})$, therefore, the parametric vector t_i of each of fireflies with $IJ + I + mI + J$ parameters is expressed as:

$$t_i = (w_{11}^i,\ w_{12}^i,...,w_{IJ}^i,\ \alpha_1^i,\alpha_2^i,....,\ \alpha_I^i,\ \mathrm{c}_{11}^{\mathrm{i}},\ \mathrm{c}_{12}^{\mathrm{i}},...,\ \mathrm{c}_{1\mathrm{m}}^{\mathrm{i}},.....,c_{I1}^i,\ \mathrm{c}_{I2}^{\mathrm{i}},...,\mathrm{c}_{Im}^{\mathrm{i}},\beta_1^i,\beta_2^i,,...\beta_m^i,....,\beta_J^i)$$

In fact, each of fireflies can represent a specific RBF network for classification. In our proposed FF-based training algorithm, the optimum vectors t_i of firefly of specific trained RBF network can maximize the fitness function defined in the Eq. (18).

$$f(t_i) = \frac{1}{1 + MSE} = \frac{1}{1 + \frac{1}{N}\sum_{k=1}^{N} \left\| d(x_k) - o(x_k) \right\|^2} \tag{18}$$

where $d(x_i)$ and $o(x_i)$ are denoted to the desired output vector and actual output vector for training sample x_i of RBF network designed by parametric vector t_i. The N is the number of the training samples. Figure 2 shows the pseudo codes of this proposed algorithm and the steps of the proposed algorithm are detailed described as follows.

Step 1. (Generate the initial solutions and given parameters)

In this step, the initial population of m solutions are generating with dimension $IJ + I + mI + J$, denoted by the matrix D.

$$D = [t_1, t_2, ..., t_n]$$

$$t_i = (w_{11}^i,\ w_{12}^i,..,w_{IJ}^i,\ \alpha_1^i,\alpha_2^i,..,\ \alpha_I^i,\ \mathrm{c}_{11}^{\mathrm{i}},\ \mathrm{c}_{12}^{\mathrm{i}},..,\ \mathrm{c}_{1\mathrm{m}}^{\mathrm{i}},..,c_{I1}^i,\ \mathrm{c}_{I2}^{\mathrm{i}},...,\mathrm{c}_{Im}^{\mathrm{i}},\beta_1^i,\beta_2^i,.,\beta_m^i,.,\beta_J^i) \tag{19}$$

where the values of weights (w) and centers (c) are assigned between -1 and 1, and the values of the spread and bias parameters α and β range from 0 to 1. Furthermore, the step will assign the parameters of firefly algorithm, that are σ, β_0, the maximum cycle number (MCL) and γ. Let number of cycle l to be 0.

Step 2. Firefly movement

In step 2, each solution t_i computes its fitness value $f(t_i)$ as the corresponding the brightness of firefly. For each solution t_i, this step randomly selects another one solution t_j with the more bright and then moves toward to t_j by using the following equations.

$$r_{i,j} = \left\| t_i - t_j \right\| = \sqrt{\sum_{k=1}^{IJ+I+mI+J} (t_{i,k} - t_{j.k})^2} \tag{20}$$

$$\beta = \beta_0 e^{-\gamma r_{i,j}} \tag{21}$$

$$t_{i,k} = (1-\beta) t_{i,k} + \beta t_{j,k} + u_{j,k}, \; k=1,2,\ldots, IJ + I + mI + J \tag{22}$$

where $u_{j,k} \sim U(0,1)$ is a randomly number ranged form 0 to 1 and the $t_{i,k}$ is the *k*-th element of the solution t_i.

Step 3. (Select the current best solution)

The step 3 selects the best one from the all solutions and defines as x_i^{max}, that is,

$$\begin{aligned} i^{max} &= \arg\max_i f(t_i); \\ x_i^{max} &= \arg\max_{x_i} f(t_i); \end{aligned} \tag{23}$$

Step 4. (Check the termination criterion)

If the cycle number *l* is equal to the *MCL* then the algorithm is finished and output the best solution x_i^{max}. Otherwise, *l* increases by one and randomly walks the best solution x_i^{max} then go to Step 2. The best solution x_i^{max} will randomly walk its position based the following equation.

$$t_{i^{max},k} \leftarrow t_{i^{max},k} + u_{i^{max},k}, \; k = 1,2,\ldots, IJ + I + mI + J \tag{24}$$

where $u_{i^{max},k} \sim U(0,1)$ is a randomly number ranged from 0 to 1.

5. Experimental results and discussion

The platform used to develop the five training algorithm included the gradient descent (GD), genetic algorithm (GA), particle swarm optimization (PSO), artificial bee colony algorithm (ABC) and the firefly algorithm (FF) is a person computer with following features: Intel Pentium IV 3.0 GHZ CPU, 2GB RAM, a Windows XP operating system and the Visual C++ 6.0 development environment. In experiments, learning parameter of GD is selected as $\eta = 0.01$. The used parameters of GA, PSO, ABC and FF algorithms are given at Tables 1, 2, 3 and 4, respectively. In order to obtain the classification results without partiality, the

following data set are used: Iris, Wine, Glass, Heart SPECTF and Breast cancer (WBDC) listed in Table 5, taken from the UCI machine repository (Asuncion, 2007).

In order to avoid the feature values in greater numeric ranges from dominating those in smaller numeric range, the scaling of feature is used, that is the range of each feature value can be linearly scaled to range [-1, 1]. Furthermore, the 4-fold method is employed in experiments, thus, the dataset is split into 4 parts, with each part of the data sharing the same proportion of each class of data. Three data parts is applied in the training process, while the remaining one is used in the testing process. The program is run 4 times to enable each slice of data to take a turn as the testing data. The percentage of correct classification of this experiment is computed by summing the individual accuracy rate for each run of testing, and then dividing the total by 4.

Firefly Algorithm

Input:

$f(x)$, $x=[t_1,t_2,....,t_m]$, t_i is the i-th firefly (solution) in the solution space with x the fitness function $f(t_i)$, $t_i=[t_{i,1},t_{i,2},....,t_{i,IJ+I+mI+J}]$, any of fireflies is a $IJ+I+mI+J$-dimensional vector, and the given parameters m, α, β_0, iteration number l and γ.

Output:

The best solution x_i^{max} with the largest fitness value.

for $i=1$ to m do

$t_i \leftarrow generate\_InitialSolutions()$;

iter=0;

Repeat

$i^{max}=\arg\max_i f(t_i)$;

$t_i^{max}=\arg\max_{t_i} f(t_i)$;

for $i=1$ to m do

for $j=1$ to m do

if $f(x_j)<f(x_i)$ then

$\{r_{i,j} \leftarrow distance(x_i,x_j);\ \beta \leftarrow \beta_0 e^{-\gamma r_{i,j}}$;

$\mu_i \leftarrow generate\_random\_vector()$;

for $k=1$ to $IJ+I+mI+J$ do

$t_{i,k} \leftarrow (1-\beta)t_{i,k}+\beta t_{j,k}+u_{i,k}$;}

$\mu_i^{max} \leftarrow generate\_random\_vector()$;

for $k=1$ to $IJ+I+mI+J$ do

$\{t_{i^{max},k} \leftarrow t_{i^{max},k}+t_{i^{max},k}$;}

iter++;

Until (iter< l)

Fig. 2. The pseudo-code of the firefly algorithm for the training the RBF network

Parameter	Parameter value
Number of iteration	1000
Number of individuals	50
Selection type	Roulette
Mutation type	Uniform
Mutation ratio	0.05
Crossover type	Single point
Crossover ratio	0.8

Table 1. The used parameters of GA

Parameter	Value
Number of particles	50
Velocities randomly	[0.0, 1.0]
Number of iterations	1000
Cognitive coefficient C1	2.1
Cognitive coefficient C2	2.0

Table 2. The used parameters of PSO

Parameter	Value
Number of the initial solutions	50
Limit	100
MCN	1000

Table 3. The used parameters of ABC

Parameter	Value
Attractiveness β_0	1.0
Light absorption coefficient γ	1.0
Number of fireflies	50
Iteration number	1000
σ	0.1

Table 4. The used parameters of Firefly algorithm

Dataset	**Class Number**	**Attributes number**	**Number of patterns**
Iris	3	4	150
Wine	3	13	178
Glass	2	9	214
Heart SPECTF	2	22	267
Breast WDBC	2	30	569

Table 5. The used datasets in this study

Qasem & Shamsuddin (2011) uses three indices to evaluate the performance of trained RBF network using the different algorithms. The three performance indices are:

The percent of correct classification (PCC) is used as the measure for evaluating the trained RBF network.

$$PCC = \frac{\text{correct classification samples}}{\text{total samples}} \times 100 \tag{25}$$

The mean square error (MSE) on the data set is used to act as the performance index shown in (3) where $o(x_k)$ and $d(x_k)$ are the actual output and the desired output and N is the number of data paris in all dataset.

$$MSE = \frac{1}{N}\sum_{k=1}^{N}\left\|d(x_k) - o(x_k)\right\|^2 \tag{26}$$

The complexity index shows in (27) that is the sum of squared weights which is based on the concept of regularization and represents the smoothness of the RBF network.

$$Complexity = \frac{1}{2}\sum_{i=1}^{I}\sum_{j=1}^{J} w_{ij}^2 \tag{27}$$

5.1 Classification evaluation

One of the most important issues of designing the RBF network is the number of neurons in the hidden layer. Thus, we implement the RBF networks which have 1 neuron to 8 neurons for comparison, and each dataset is running 10 times based on 4-flod cross-validation. The average percentage and the corresponding standard derivation defined as the Eq. (25) of the designed RBF network by different algorithms are listed in Tables 6-10.

	The number of neuron of hidden layer						
	2	**3**	**4**	**5**	**6**	**7**	**8**
GD	75.33 ± 6.09	81.33 ± 5.89	84.67 ± 4.98	88.00 ± 5.12	89.33 ± 4.34	90.00 ± 4.21	89.33 ± 3.13
GA	84.66 ± 6.78	89.33 ± 6.88	90.00 ± 5.42	90.67 ± 4.65	92.00 ± 4.52	94.00 ± 3.45	91.67 ± 2.87
PSO	86.67 ± 4.23	92.21 ± 4.01	93.67 ± 3.32	94.00 ± 2.89	94.67 ± 2.34	95.45 ± 2.55	97.33 ± 1.78
ABC	87.33 ± 4.31	92.00 ± 3.97	94.67 ± 3.14	93.33 ± 2.69	94.67 ± 2.65	94.21 ± 2.47	96.14 ± 2.67
FF	**87.33 ± 2.13**	**93.33 ± 2.23**	**94.00 ± 2.98**	**94.00 ± 1.45**	**94.67 ± 1.23**	**96.14 ± 1.43**	**97.33 ± 1.02**

Table 6. The average PCC and standard deviation results of the Iris dataset using different algorithm.

	The number of neuron of hidden layer (sec)						
	2	**3**	**4**	**5**	**6**	**7**	**8**
GD	70.79 ± 6.53	74.16 ± 5.43	76.97 ± 6.32	79.21 ± 4.32	84.83 ± 3.89	88.96 ± 3.91	90.76 ± 3.23
GA	89.53 ± 5.23	88.65 ± 3.42	92.13 ± 3.56	90.45 ± 2.21	93.82 ± 2.34	95.35 ± 1.98	94.98 ± 1.64
PSO	90.52 ± 4.32	91.57 ± 3.29	92.13 ± 2.89	93.82 ± 2.45	94.38 ± 2.43	94.70 ± 1.98	95.35 ± 2.31
ABC	94.76 ± 4.06	95.45 ± 3.61	96.57 ± 3.41	95.10 ± 2.54	95.47 ± 3.14	96.70 ± 2.15	97.82 ± 2.51
FF	**94.76 ± 3.21**	**96.01 ± 2.87**	**97.45 ± 2.67**	**98.07 ± 2.23**	**97.82 ± 2.45**	**97.94 ± 1.86**	**98.07 ± 1.22**

Table 7. Statistical average PCC results of the Wine dataset using different algorithms

	The number of neuron of hidden layer						
	2	**3**	**4**	**5**	**6**	**7**	**8**
GD	64.49 ± 6.58	65.89 ± 7.14	68.69 ± 7.25	74.30 ± 5.21	78.51 ± 4.52	86.92 ± 3.25	93.39 ± 2.58
GA	69.62 ± 5.54	75.23 ± 4.25	85.05 ± 3.25	86.92 ± 3.98	89.25 ± 4.15	90.65 ± 3.24	92.25 ± 2.68
PSO	**89.16 ± 5.28**	**94.29 ± 4.68**	**92.25 ± 3.78**	**95.79 ± 4.12**	**95.79 ± 3.81**	**97.19 ± 2.45**	**98.48 ± 2.17**
ABC	92.25 ± 5.14	92.25 ± 4.21	92.25 ± 4.87	94.39 ± 4.51	95.79 ± 3.53	95.79 ± 2.26	98.48 ± 2.97
FF	92.25 ± 6.12	92.25 ± 3.91	94.39 ± 3.24	94.39 ± 4.18	94.39 ± 3.10	95.79 ± 2.19	97.19 ± 1.97

Table 8. Statistical average PCC results of the Glass dataset using different algorithms.

	The number of neuron of hidden layer						
	2	**3**	**4**	**5**	**6**	**7**	**8**
GD	61.42 ± 5.25	63.29 ± 5.65	68.91 ± 4.25	77.91 ± 3.95	78.65 ± 5.24	84.26 ± 3.24	88.37 ± 3.25
GA	60.67 ± 4.26	71.53 ± 4.64	79.40 ± 4.06	89.14 ± 4.58	88.39 ± 3.25	89.14 ± 2.85	92.13 ± 2.14
PSO	71.53 ± 4.52	72.23 ± 3.79	85.39 ± 3.14	86.52 ± 3.95	88.39 ± 2.52	88.76 ± 4.19	92.88 ± 2.53
ABC	74.16 ± 3.25	76.40 ± 3.21	85.39 ± 3.51	88.39 ± 3.28	89.14 ± 3.84	92.13 ± 2.91	95.18 ± 3.29
FF	**74.16 ± 3.69**	**79.40 ± 4.15**	**85.39 ± 4.09**	**88.39 ± 2.85**	**89.51 ± 3.12**	**95.18 ± 2.17**	**95.18 ± 1.56**

Table 9. Statistical average PCC results of the Heart SPECT dataset using different algorithms.

	The number of neuron of hidden layer						
	2	**3**	**4**	**5**	**6**	**7**	**8**
GD	75.92 ± 8.45	80.49 ± 6.78	85.59 ± 5.62	87.52 ± 5.67	88.05 ± 6.17	89.98 ± 4.78	91.21 ± 3.56
GA	84.44 ± 6.87	85.59 ± 5.97	93.50 ± 4.21	94.20 ± 4.54	93.85 ± 3.91	96.49 ± 3.21	98.36 ± 3.67
PSO	93.32 ± 5.34	93.59 ± 4.98	94.38 ± 3.76	95.08 ± 4.19	96.49 ± 3.27	97.19 ± 2.98	98.36 ± 2.65
ABC	93.84 ± 6.10	94.37 ± 4.12	95.78 ± 3.84	96.49 ± 3.61	95.85 ± 4.14	96.49 ± 3.19	98.49 ± 3.14
FF	**93.32 ± 4.78**	**93.59 ± 4.98**	**95.78 ± 3.23**	**96.13 ± 3.43**	**97.19 ± 2.87**	**98.49 ± 2.57**	**99.72 ± 1.87**

Table 10. Statistical average results of the WDBC dataset using different algorithms.

These tables reveal that GD is the worst because the gradient descent algorithm is a traditional derivative method which traps at local minima. Furthermore, unlike the other four algorithms, as the number of neurons increases, the correct classification rates of the network designed by GD algorithm increase accordingly. In other words, the usage of bio-inspired algorithms is more robust than traditional GD algorithms. The Table 6 and 7 are the classification results of the Iris and Wine datasets, which are three-class classification problems. In Table 6, we find the fact that the results of the deigned RBF networks using the PSO, ABC and FF are not significantly difference but are superior to the results using GA. In Table 7, the results of ABC and FF algorithms are better than the results of the GA and PSO algorithms. These results may reveal that the GA and PSO algorithms need more number of initials or more execution iterations for searching the optimal parameters of the radial basis function network. Tables 8-10 are the classification results of the Glass, Heart SPECTF and WDBC datasets, which are two-class classification problems. We also find that the results designed by the PSO, ABC and FF algorithms are better than the result of GA algorithm. The better results of each of the three tables are the usages of PSO, ABC and FF, but, the differences between them are not indistinct from these tables.

5.2 The analysis of complexity and mean square error

Generally speaking, the complexity of trained RBF network with a large number of hidden nodes is larger but its corresponding mean square error is smaller. In experiments, The Figs. 3-7 recorded the mean square error and complexity of each trained RBF network based the Eq. (23) and (24). These figures clearly appear the phenomenon that the GD is the worst because of the largest mean square error with the same complexity among all algorithms.

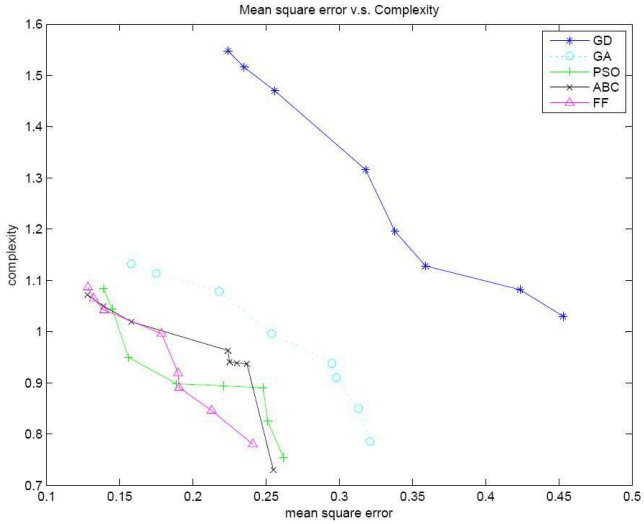


Fig. 3. The mean square error versus complexity of the classification of the Iris dataset.

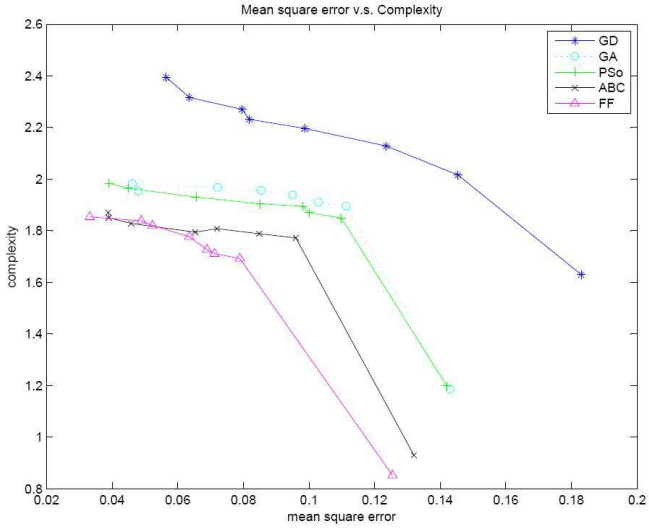


Fig. 4. The mean square error versus complexity of the Wine classification.

Figure 3 shows the relationship between the complexity and mean square error in training the RBF networks of Iris dataset. These figures appear that the results of PSO, ABC and FF are superior to the GD. The Fig. 4 show the results of ABC and FF algorithm are superior to the results of GD and PSO in the training the RBF networks for Wine dataset. The Fig. 5 show the best is the result designed by PSO algorithm. The Fig. 7 demonstrates the best are the usages of the FF, however, the results of GD, PSO, ABC and FF do not clearly differentiate form the results of Fig. 6.

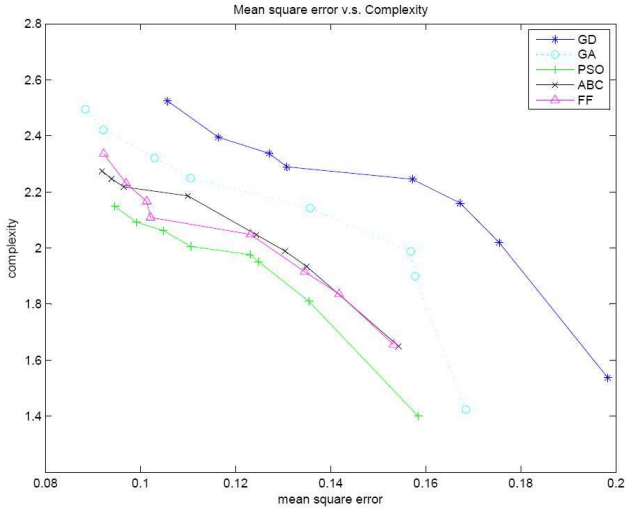


Fig. 5. The mean square error versus complexity of the Glass classification.

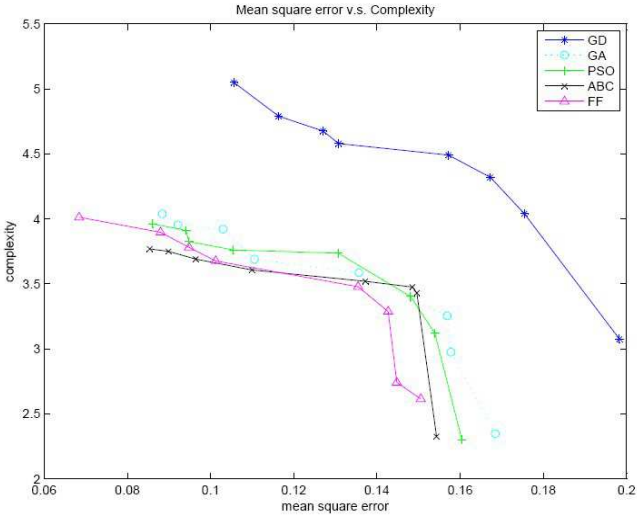


Fig. 6. The mean square error v.s. complexity of the Heart SPECT classification.

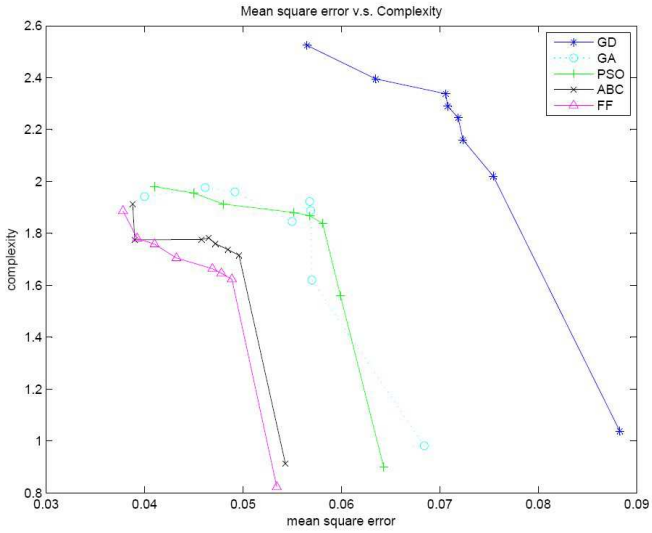


Fig. 7. The mean square error v.s. complexity of the WDBC classification.

5.3 Receiver operating characteristic analysis

The receiver operating characteristics analysis is a graphical curve is a tool for two-class classification problems that gives the evaluation of the predictive accuracy of a logistic model. The curve displays the relationship of the true positive rate (sensitivity) and the false positive rate (1-specificity) within a range of cutoffs. The sensitivity is a measure of accuracy for predicting events that is equal to the true positive/total actual positive; nevertheless, the specificity is a measure of accuracy for predicting nonevents that is equal to the true negative/total actual negative of a classifier. The area under curve (AUC) is an important index for evaluating the performance of classification. In general, the high AUC represents to good performance in the classification problems. The classifications of the two Heart SPECTF and Breast WDBC datasets listed Table 5 are two-class problems of the medical diagnosis that are suitable for this analysis. The SPECT dataset generated from describes diagnosing of cardiac single proton emission computed tomography images. The database of 267 SPECT image sets (patients) with 22 binary attributes was processed to extract features that summarize the original SPECT images and each of the patients is classified into two categories: normal (negative) and abnormal (positive). The Wisconsin Diagnostic Breast Cancer (WDBC) dataset was collected from Dr. William H. Wolberg of Wisconsin University. The dataset includes 567 data samples with 30 continuous attributes that are divided into 357 benign (negative) and 210 malignant (positive). In order to take one step ahead for analyzing the capability of classifications by using the five algorithms, the average of sensitivity and the average specificity of the receiver operating characteristic (ROC) analysis by using the SPECTF and WDBC datasets under the eight hidden nodes of trained RBF network are listed in the Table 11; and further, the corresponding AUC of ROC analysis with varied the bias parameters also listed in this table. In this table we find that the usage of ABC algorithm can have better capability in the classification of the SPECT dataset,

however, the FF algorithm is best in the classification of WDBC dataset. The average computation times of classifying the Heart SPECT dataset in 4-fold cross validation by using the GD, GA, PSO, ABC and FF are 0.21, 429.67, 103.76, 123.67 and 98.21 seconds, however, the average computation times of classifying the Breast dataset in 4-fold cross validation by using the GD, GA, PSO, ABC and FF are 0.24, 513.23, 161.84, 189.59 and 134.91 seconds

Algorithms	Heart SPECTF Database			Breast (WDBC) Database		
	Sensitivity	Specificity	AUC	Sensitivity	Specificity	AUC
GD	0.8868	0.8727	0.789	0.9151	0.8868	0.854
GA	0.9198	0.9273	0.896	0.9811	0.9860	0.944
PSO	0.9292	0.9010	0.902	0.9858	0.9832	0.961
ABC	**0.9528**	**0.9454**	**0.941**	0.9953	0.9832	0.975
FF	0.9321	0.9367	0.932	**1.0000**	**0.9944**	**0.984**

Table 11. Area under curve (AUC) of ROC analysis of RBF network with eight hidden nodes. (The best results are highlighted in bold)

6. Conclusions

In this chapter, the firefly algorithm has been applied to train the radial basis function network for data classification and disease diagnosis. The training procedure involves selecting the optimal values of parameters that are the weights between layer and the output layer, the spread parameters, the center vectors of the radial functions of hidden nodes; and the bias parameters of the neurons of the output layer. The other four algorithms that are gradient descent (GD), genetic algorithm (GA), particle swarm optimization (PSO) and artificial bee colony algorithms are also implemented for comparisons. In experiments, the well-known classification problems such as the iris, Wine, Glass, heart SPECT and WDBC datasets, obtained from UCI repository had been used to evaluate the capability of classification among the five algorithms. Furthermore, the complexity and trained error also be discussed form experiments conducted in this chapter. The experimental results show that the usage of the firefly algorithm can obtain the satisfactory results over the GD and GA algorithm, but it is not apparent superiority to the PSO and ABC methods form exploring the experimental results of the classifications of UCI datasets. In order to go a step further for talking over the capability of classification among the five algorithms, the receiver operating characteristic (ROC) analysis are applied for this objective in classification of the heart SPECT and WDBC datasets. The experimental results also appear that the use of firefly algorithm has satisfactory in the high sensitivity, high specificity and bigger AUC in the corresponding ROC curves in WDBC dataset, however, the differences between ABC, PSO and firefly algorithms are not significant. The experimental results of this chapter reveal that the swarm intelligence algorithms, such as the particle swarm optimization, the artificial bee colony algorithm and the firefly algorithm are the good choices to search for the parameters of radial basis function neural network for classifications and disease diagnosis.

7. Acknowledgment

The authors would like to thank the National Science Council, ROC, under Grant No. NSC 100-2221-E-251-012 for support of this work.

8. References

Asuncion A. &, Newman D. (2007), UCI Machine Learning Repository. URL:http:// www. ics.uci.edu/~mlearn/MLRepository.html.

Barreto, Ada.M.S., Barbosa, H.J.C & Ebecken, N.F.F. (2002). Growing Compact RBF Networks Using A Genetic Algorithm, *In Proceedings of the VII Brazilian Symposium on Neural Networks*, pp. 61-66.

Bishop, C.M. (1995). Neural Networks for Pattern Recognition, Oxford, Clarendon Press.

Feng H.M. (2006), Self-generating RBFNs Using Evolutional PSO Learning, *Neurocomputing*, Vol. 70, pp. 241-251.

Feng, M.H. & Chou, H.C. (2011). Evolutional RBFNs Prediction Systems Generation in the Applications of Financial Time Seriies Data. Expert Systems with Applications, Vol. 38, 8285-8292.

Goldberg, David E. (1989). Genetic Algorithm in Search Optimization and Machine Learning, *Addison Wesley*.

Horng, M.H. (2010). Performance Evaluation of Multiple Classification of the Ultrasonic Supraspinatus Image by using ML, RBFNN and SVM Classifier, *Expert Systems with Applications*, 2010, Vol. 37, pp. 4146-4155.

Karayiannis, N.B. (1999). Reformulated Radial Basis Function Neural Network Trained by Gradient Descent. *IEEE Trnas. Neul Netw.*, Vol. 3, No, 10, pp. 657-671.

Kennedy J. & Eberhart, R. C. (1995). Particle Swarm Optimization, *In the Proceedings of the IEEE International Conference on Neural networks*. pp.1942-1948.

Korurek, M. & Dogan, B. (2010). ECG Beat Classification Using Particle Swarm Optimization and Radial Basis Function Network. *Expert Systems with Application*, Vol. 37, pp. 7563-7569.

Kurban T. & Besdok, E. (2009). A Comparison of RBF Neural Network Training Algorithms for Inertial Sensor based Terrain Classification, *Sensors*, Vol..9, pp. 6312-6329.

Lukasik, S. & Zak S. (2009). Firefly Algorithms for Continuous Constrained Optimization, *Computional Collective Intelligence, Semantic Web, Social Networks and Multigent Systems, Lecture Notes in Computer Sciences,* Vol. 5796, pp.97-106.

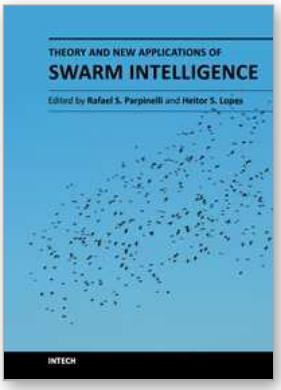
Ou, T. T., Oyang Y. J. & Chen C. Y. (2005). A Novel Radial Basis Function Network Classifier with Centers Set by Hierarchical Clustering. , *In Proceedings of the international joint conference on neural network, (IJCNN '05)*, Vol. 3, pp.1383-1388.

Qasem S.N. & Shamsuddin S.M. (2011). Radial Basis Function Network Based on Time Variant Multi-objective Particle Swarm Optimization for Medical Diseases Diagnosis. *Applied Soft Computing,* Vol. 11, pp. 1427-1438.

Wu, D., Warwick, K., Ma, Z., Gasson, M. N., Burgess, J. G., Pan, S. & Aziz, T. (2010). A Prediction of Parkinson's Disease Tremor Onset Using Radial Basis Function Neural Networks, *Expert Systems with Applications*, Vol .37, pp.2923-2928.

Yang X.S. (2010). Firefly Algorithm, Stochastic Test Functions and Design Optimization, *Int. J. Bio-inspired Computation*, No.2, Vol. 2, pp. 78-84.

Yang, X.S. (2008). Nature-inspired Metaheuristic Alogirthms. Frome; Luniver Press. ISBN 2008.



Theory and New Applications of Swarm Intelligence

Edited by Dr. Rafael Parpinelli

ISBN 978-953-51-0364-6
Hard cover, 194 pages
Publisher InTech
Published online 16, March, 2012
Published in print edition March, 2012

The field of research that studies the emergent collective intelligence of self-organized and decentralized simple agents is referred to as Swarm Intelligence. It is based on social behavior that can be observed in nature, such as flocks of birds, fish schools and bee hives, where a number of individuals with limited capabilities are able to come to intelligent solutions for complex problems. The computer science community have already learned about the importance of emergent behaviors for complex problem solving. Hence, this book presents some recent advances on Swarm Intelligence, specially on new swarm-based optimization methods and hybrid algorithms for several applications. The content of this book allows the reader to know more both theoretical and technical aspects and applications of Swarm Intelligence.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821