Hybrid Learning Enhancement of RBF Network with Particle Swarm Optimization

Sultan Noman, Siti Mariyam Shamsuddin, and Aboul Ella Hassanien

Abstract. This study proposes RBF Network hybrid learning with Particle Swarm Optimization (PSO) for better convergence, error rates and classification results. In conventional RBF Network structure, different layers perform different tasks. Hence, it is useful to split the optimization process of hidden layer and output layer of the network accordingly. RBF Network hybrid learning involves two phases. The first phase is a structure identification, in which unsupervised learning is exploited to determine the RBF centers and widths. This is done by executing different algorithms such as k-mean clustering and standard derivation respectively. The second phase is parameters estimation, in which supervised learning is implemented to establish the connections weights between the hidden layer and the output layer. This is done by performing different algorithms such as Least Mean Squares (LMS) and gradient based methods. The incorporation of PSO in RBF Network hybrid learning is accomplished by optimizing the centers, the widths and the weights of RBF Network. The results for training, testing and validation of five datasets (XOR, Balloon, Cancer, Iris and Ionosphere) illustrates the effectiveness of PSO in enhancing RBF Network learning compared to conventional Backpropogation.

Keywords: Hybrid learning, Radial basis function network, K-means, Least mean squares, Backpropogation, Particle swarm optimization, Unsupervised and supervised learning.

1 Introduction

Radial Basis Function (RBF) Networks form a class of Artificial Neural Networks (ANNs), which has certain advantages over other types of ANNs, such as better

Sultan Noman and Siti Mariyam Shamsuddin
Soft Computing Group,
Faculty of Computer Science and Information System, UTM, Malaysia
e-mail: sultannoman@yahoo.com, mariyam@utm.my

Aboul Ella Hassanien
Information Technology Department,
Faculty of Computer Science and Information System, Cairo University
e-mail: aboitcairo@gmail.com

A.-E. Hassanien et al. (Eds.): Foundations of Comput. Intel. Vol. 1, SCI 201, pp. 381–397.
springerlink.com

approximation capabilities, simpler network structures and faster learning algorithms. The RBF Network is a three layer feed forward fully connected network, which uses RBFs as the only nonlinearity in the hidden layer neurons. The output layer has no nonlinearity and the connections of the output layer are only weighted, the connections from the input to the hidden layer are not weighted [1].

Due to their better approximation capabilities, simpler network structures and faster learning algorithms, RBF Networks have been widely applied in many science and engineering fields. It is three layers feedback network, where each hidden unit implements a radial activation function and each output unit implements a weighted sum of hidden units' outputs. Its training procedure is usually divided into two stages. The first stage includes determination of centers and widths of the hidden layer which are obtained from clustering algorithms such as K-means, vector quantization, decision trees, and self-organizing feature maps. The second stage involves weights establishment by connecting the hidden layer with the output layer. This is determined by Singular Value Decomposition (SVD) or Least Mean Squares (LMS) algorithms [2]. Clustering algorithms have been successfully used in training RBF Networks such as Optimal Partition Algorithm (OPA) to determine the centers and widths of RBFs. In most traditional algorithms, such as the K-means, the number of cluster centers need to be predetermined, which restricts the real applications of the algorithms. In addition, Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Self-Organizing Maps (SOM) are also been considered in clustering process [4].

In this study, PSO is explored to enhance RBF learning mechanism. The paper is structured as follows. Section 2, related work about RBF Network training is introduced. Section 3 presents RBF Network model and parameter selection problem. In section 4 describes PSO algorithm. BP-RBF Network model is given in Section 5. Section 6 describes our proposed approach. Sections 7 and 8 give the experiments setup, results and validation results of the proposed model on datasets respectively. The comparison between PSO-RBF Network and BP-RBF Network is presented in section 9 and finally, the paper is concluded in Section 10.

2 Related Work

Although there are many studies in RBF Network training, but research on training of RBF Network with PSO is still fresh. This section presents some existing work of training RBF Network based on Evolutionary Algorithms (EAs) such as PSO especially based on unsupervised learning only (Clustering).

In [11], they have proposed a PSO learning algorithm to automate the design of RBF Networks, to solve pattern classification problems. Thus, PSO-RBF finds the size of the network and the parameters that configure each neuron: center and width of its basis function. Supervised mean subtractive clustering algorithm has been proposed [13] to evolve RBF Networks and the evolved RBF acts as fitness evaluation function of PSO algorithm for feature selection. The method performs feature selection and RBF training simultaneously. PSO algorithm has been introduced [12] to train RBF Network related to automatic configuration of network architecture related to centers of RBF. Two training algorithm were compared. One was PSO algorithm. The other was newrb routine that was included in Matlab neural networks toolbox as standard training algorithm for RBF network.

A hybrid PSO (HPSO) was proposed [15] with simulated annealing and Chaos search technique to train RBF Network. The HPSO algorithm combined the strong ability of PSO, SA, and Chaos. An innovative Hybrid Recursive Particle Swarm Optimization (HRPSO) learning algorithm with normalized fuzzy c-mean (NFCM) clustering, PSO and Recursive Least Squares (RLS) has been presented [16] to generate RBF networks modeling system with small numbers of descriptive RBFs for fast approximating two complex and nonlinear functions. On other hand, a newly evolutionary search technique called Quantum-Behaved Particle Swarm Optimization, in training RBF Network has been used [17]. The proposed QPSO-Trained RBF Network was test on nonlinear system identification problem.

Unlike previous studies, this research shares consideration of parameters of RBF (unsupervised learning) which are centers and length of width or spread of RBFs with different algorithms such as K-means and K-nearest neighbors or standard deviations algorithms respectively. However, training of RBF Network need to enhance with PSO to optimize the centers and widths values which are obtained from the clustering algorithms and PSO also used to optimize the weights which connect between hidden layer and output layer (supervised learning). Also this paper has been presented to train, test and validate the PSO-RBF Network on the datasets.

3 Architecture of RBF Network

RBF Network is structured by embedding radial basis function a two-layer feed-forward neural network. Such a network is characterized by a set of inputs and a set of outputs. It is used in function approximation, time series prediction, and control, and the network architecture is constructed with three layers: input layer, hidden layer, and output layer. The input layer is made up of source nodes that connect the network to its environment. The second layer, the only hidden layer of the network, applies a non-linear transformation from the input space to a hidden space. The nodes in the hidden layer are associated with centers that determine the behavior structure of network. The response from the hidden unit is activated through RBF using Gaussian function or other functions. The output layer provides the response of the network to the activation pattern of the input layer that serves as a summation unit.

In a RBF model, the layer from input nodes to hidden neurons is unsupervised and the layer from hidden neurons to output nodes is supervised. The model is given by the following equation for the j^{th} output $y_j(i)$:

$$y_j(i) = \sum_{k=1}^{K} w_{jk} \Phi_k [x(i), c_k, \sigma_k] \qquad (1)$$
$$j = 1,2,.....,n \qquad i = 1,2,.....,N,$$

Where K is the number of RBFs used, and $c_k \in R^m$, $\sigma_k \in R^m$, are the center value vector and the width value vector of RBF, respectively. These vectors are defined as:

$$c_k = [c_{k1} \quad c_{k2} \quad ... \quad c_{km}]^T \in R^m \quad , \quad k = 1,....., K$$
$$\sigma_k = [\sigma_{k1} \quad \sigma_{k2} \quad ... \quad \sigma_{km}]^T \in R^m \quad , \quad k = 1,....., K, \tag{2}$$

Where, $\{w_{jk} \mid k = 1, 2,\ldots, K\}$ are the weights of RBFs connected with the j^{th} Output. Fig. 1 shows the structure of RBF Network.

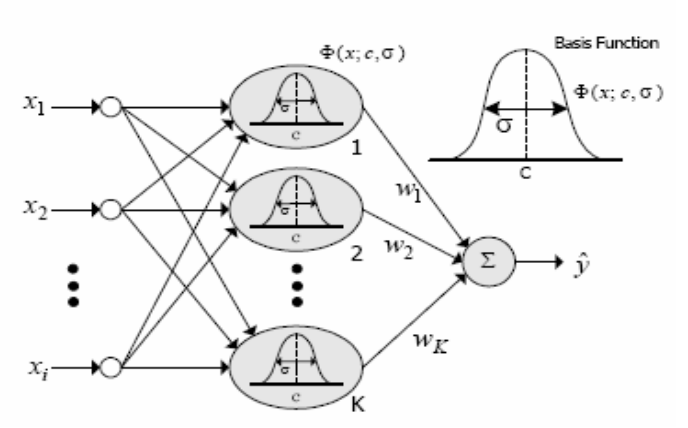

Fig. 1 Structure of RBF network

RBF Network can be implemented in a two-layered network. For a given set of centers, the first layer performs a fixed nonlinear transformation which maps the input space onto a new space. Each term $\Phi_k(.)$ forms the activation function in a unit of the hidden layer. The output layer then implements a linear combination of this new space.

$$\Phi_k(x\ , c_k, \sigma_k) = \prod_{i=1}^{m} \phi_{ki}(x_i, c_{ki}, \sigma_{ki}) \tag{3}$$

Moreover, the most popular choice for $\varphi(.)$ is the Gaussian form defined as

$$\phi_{ki}(x_i, c_{ki}, \sigma_{ki}) = \exp[-(x_i - c_{ki})^2 / 2\sigma_{ki}^2] \tag{4}$$

In this case, the j^{th} output in equation (1) becomes:

$$y_j(i) = \sum_{k=1}^{K} w_{jk} \exp\{-\sum_{i=1}^{m}[(x_i - c_{ki})^2 / 2\sigma_{ki}^2]]\} \tag{5}$$

In equation (5), σ_k indicates the width of the kth Gaussian RBF functions. One of the σ_k selection methods is shown as follows:

$$\sigma_k^2 = \frac{1}{m_k} \sum_{x \in \theta_k} \left\| x - c_k \right\| \tag{6}$$

Where θ_k is the kth cluster of training set and M_k is the number of sample data in the kth cluster.

The neuron number of the hidden layer, i.e., the cluster number of training set, must be determined before the parameter selection of RBF Network. If the neuron numbers of hidden layer has been decided, the performance of RBF depends on the selection of the network parameters. There are three types of parameters in RBF model with Gaussian basis functions: RBF centers (hidden layer neurons); Widths of RBFs (standard deviations in the case of a Gaussian RBF); and Output layer weights. There are two categories of training algorithms in RBF models: supervised and unsupervised learning.

4 Particle Swarm Optimization

Particle Swarm Optimization (PSO) algorithm, originally introduced by Kennedy and Eberhart in 1995 [5], simulates the knowledge evolvement of a social organism, in which each individual is treated as an infinitesimal particle in the n-dimensional space, with the position vector and velocity vector of particle i being represented as $X_i(t) = (X_{i1}(t), X_{i2}(t),\ldots, X_{in}(t))$ and $V_i(t) = (V_{i1}(t), V_{i2}(t),\ldots, V_{in}(t))$. The particles move according to the following equations:

$$V_{id}(t+1) = W \times V_{id}(t) + c_1 r_1 (P_{id}(t) - X_{id}(t)) + c_2 r_2 (P_{gd}(t) - X_{id}(t)) \tag{7}$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1)$$
$$i = 1,2,\ldots,M \; ; \; d = 1,2,\ldots,n \tag{8}$$

Where c_1 and c_2 are the acceleration coefficients, Vector $P_i = (P_{i1}, P_{i2},\ldots, P_{in})$ is the best previous position (the position giving the best fitness value) of particle i known as the personal best position (pbest); Vector $P_g = (P_{g1}, P_{g2},\ldots, P_{gn})$ is the position of the best particle among all the particles in the population and is known as the global best position (gbest). The parameters r_1 and r_2 are two random numbers distributed uniformly in (0, 1). Generally, the value of V_{id} is restricted in the interval $[-V_{max}, V_{max}]$. Inertia weight w was first introduced by Shi and Eberhart in order to accelerate the convergence speed of the algorithm [6].

5 BP-RBF Network

In our paper, the standard BP is selected as the simplest and most widely used algorithm to train feed-forward RBF Networks and considered for the full-training

paradigm; customizing it for half-training is straightforward and can be done simply by eliminating gradient calculations and weight-updating corresponding to the appropriate parameters.

The following is the procedure of BP- RBF Network algorithm:

1. Initialize network.
2. Forward pass: Insert the input and the desired output; compute the network outputs by proceeding forward through the network, layer by layer.
3. Backward pass: Calculate the error gradients versus the parameters, layer by layer, starting from the output layer and proceeding backwards: $\partial E/\partial w, \partial E/\partial c, \partial E/\partial \sigma^2$.
4. Update parameters: (weight, center and width of the RBF Network respectively)

a. $$w_{kj}(t+1) = w_{kj}(t) + \Delta w_{kj}(t+1) \tag{9}$$

$$\Delta w_{kj}(t+1) = \eta_3 \delta_k O_j \tag{10}$$

With

$$\delta_k = O_k(1-O_k)(t_k - O_k) \tag{11}$$

$$O_j = \exp(-(x-c_j)^2 / 2\sigma_j^2) \tag{12}$$

Where $w_{kj}(t)$ is the weight from node k to node j at time t, Δw_{kj} is the weight adjustment, η_3 is the learning rate, δ_k is error at node k, O_j is the actual network output at node j, O_k is the actual network output at node k and t_k is the target output value at node k.

b. $$c_{ji}(t+1) = c_{ji}(t) + \Delta c_{ji}(t+1) \tag{13}$$

$$\Delta c_{ji}(t+1) = \eta_2 \delta_k w_{kj} O_j / \sigma_j^2 (x_{ji} - c_{ji}) \tag{14}$$

Where $c_{ji}(t)$ is the centre from node j to node i at time t, Δc_{ji} is the center adjustment, η_2 is the learning rate, δ_k is error at node k, O_j is the actual network output at node j, σ_j is the width at node j, w_{kj} is the weight connected between node j and k and x_{ji} is the input node j to node i.

c. $$\sigma_j(t+1) = \sigma_j(t) + \Delta\sigma_j(t+1) \tag{15}$$

$$\Delta\sigma_j(t+1) = \eta_1 \delta_k w_{kj} O_j((x-c_j)^2 / 2\sigma_j^4) \tag{16}$$

Where $\sigma_j(t)$ is the width of node j at time t, $\Delta\sigma_j$ is the width adjustment, η_1 is the learning rate, δ_k is error at node k, O_j is the actual network output at node j, σ_j is the width at node j, w_{kj} is the weight connected between node j and k, x_{ji} is the input at node i and η_3, η_2, η_1 are learning rate factors in the range [0; 1].

5. Repeat the algorithm for all training inputs. If one epoch of training is finished, repeat the training for another epoch.

BP-RBF Network doesn't need the momentum term as it is common for the MLP. It does not help in training of the RBF Network [14].

6 PSO-RBF Network

PSO has been applied to improve RBF Network in various aspects such as network connections (centers, weights), network architecture and learning algorithm. The main process in this study is to employ PSO-based training algorithm on center, width and weight of RBF network, and investigate the efficiency of PSO in enhancing RBF training. Every single solution of PSO called (a particle) flies over the solution space in search for the optimal solution. The particles are evaluated

```
For each particle do
    initialize particle position and velocity
End for

While stopping criteria are not fulfilled do
      For each particle do
          Calculate fitness value (MSE in RBF Network)
        If fitness value is better than best fitness value pBest in particle
           history  then
          Set current position as pBest
        End if
      End for
      Choose as gBest the particle with best fitness value among all particles
in
      current iteration
      For each particle do
          Calculate particle velocity based on eq. (7)
          Update particle position(center, width and weight) based on eq. (8)
      End for
End while
```

Fig. 2 PSO-RBF Network Algorithm

using a fitness function to seek the optimal solution. Particles (center, width) values are then initialized with values which are obtained from the k-means algorithm while particles (weight, bias) values are initialized randomly or from LMS algorithm. The particles are updated accordingly using the equation eq. (7) and eq. (8).

The procedure for implementing PSO global version (gbest) is shown in Figure 2. Optimization of RBF network parameters (the center and width of RBF and the weight, the bias) with PSO, the fitness value of each particle (member) is the value of the error function evaluated at the current position of the particle and position vector of the particle corresponds to the (center, width, weight and bias) matrix of the network. The pseudo code of the procedure is as follows:

7 Experiments

7.1 Experimental Setup

The experiments of this work included the standard PSO and BP for RBF Network training. For evaluating all of these algorithms we used five benchmark classification problems obtained from the machine learning repository [10].

Table 1 Execution parameters for PSO

Parameter	Value
Population Size	20
Iterations	10000
W	[0.9,0.4]
C_1	2.0
C_2	2.0

The parameters of the PSO algorithm were set as: weight w decreasing linearly between 0.9 and 0.4, learning rate $c_1 = c_2 = 2$ for all cases. The population size used by PSO was constant. The algorithm stopped when a predefined number of iterations have been reached. Values selected for parameters are shown in table 1.

Table 2 Parameters of the experiments

Parameter	Dataset				
	XOR	Balloon	Cancer	Iris	Ionosphere
Train data	5	12	349	120	251
Test data	3	4	175	30	100
Validation data	8	16	175	150	351
Input dimension	3	4	9	4	34
Output neuron	1	1	1	3	1
Network Structure	3-2-1	4-2-1	9-2-1	4-3-3	34-2-1

The XOR data set (3 features and 8 examples) is a logical operation on three operands that results in a logical value of true if and only if one of the operands but not both have a value of true. The Balloon data set (4 features and 16 examples) is used in cognitive psychology experiment. There are four data sets representing different conditions of an experiment. All have the same attributes. The cancer dataset (9 features and 699 examples) is related to the diagnosis of breast cancer in benign or malignant. The Iris dataset (4 features and 150 examples) is used for classifying all the information into three classes. Finally, the Ionosphere dataset (34 features and 351 examples) is radar data was collected by a system in Goose Bay; Labrador is used for classifying all the information into "Good" or "Bad" results.

The number of maximum iterations is set differently to bound the number of forward propagations to 4×10^4 and for comparison purposed. The maximum iterations in BP-RBFN is set to 2×10^4 (number of forward propagations = 2 × maximum number of iterations), while the maximum number of iterations in PSO-RBFN is set to 10000 (number of forward propagations = swarm size × maximum number of iterations) [9]. The stopping criteria are the maximum number of iterations that the algorithm has been reached or the minimum error.

The architecture of the RBF Network was fixed in one hidden layer (number of inputs of the problem - 2 hidden units - 1 output units) in XOR, Balloon, Cancer, Ionosphere and (number of inputs of the problem - 3 hidden units - 3 output units) in Iris dataset. The parameters of the experiments are described in Table 2.

7.2 Experimental Results

This section presents the results of the study on PSO-trained RBF Network and BP-trained RBF Network. The experiments are conducted by using five datasets: XOR, Balloon, Cancer, Iris and Ionosphere. The results for each dataset are compared and analysed based on the convergence, error and classification performance.

7.2.1 XOR Dataset

A connective in logic known as the "exclusive or" or exclusive disjunction is a logical operation on three operands. Two algorithms used to train and test of RBF Network. The stopping conditions of PSO-RBFN are set as minimum error of 0.005 or maximum iteration of 10000. On the other hand, the stopping conditions for BP-RBFN are set as the minimum error of 0.005 or the iterations have reached to 20000. The results for PSO-based RBFN and BP-based RBFN are illustrated in Table 3 and Figure 3. From Table 3, PSO-RBFN converges at 93 iterations compared to BP-RBFN with 5250 iterations for the whole learning process. Both algorithms are converged with given minimum error. For the classification, it shows that BP-RBFN is better than PSO-RBFN with 93.88% compared to 93.51%. However, PSO-RBFN converges faster compared to BP-RBFN. The classification rate for testing of XOR problem is not good due to smaller amount of data to be learned by the network.

Table 3 Result of BP-RBFN and PSO-RBFN on XOR dataset

	BP-RBFN		PSO-RBFN	
	Train	Test	Train	Test
Learning Iteration	5250	1	93	1
Error Convergence	0.00500	0.32972	0.004998	0.28693
Classification (%)	93.88	64.72	93.51	64.03

Figure 3 illustrates that PSO-RBFN significantly reduces the error with minimum iterations compared to BP-RBFN.

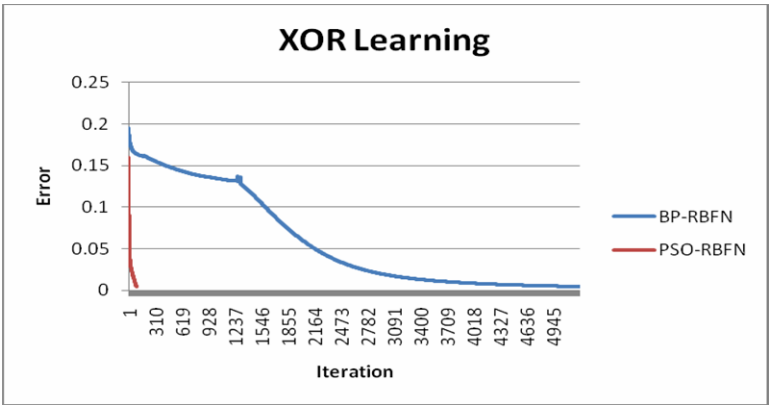

Fig. 3 Convergence rate of XOR dataset

7.2.2 Balloon Dataset

This data is used in cognitive psychology experiment. There are four data sets representing different conditions of an experiment. All have the same attributes.

It contains 4 attributes and 16 instances. The stopping conditions of PSO-RBFN are set to a minimum error of 0.005 or maximum iteration of 10000. Conversely, the stopping conditions for BP-RBFN are set to the minimum error of 0.005 or the iterations have reached to 20000. From Table 4, we conclude that PSO-RBFN converges faster compared to BP-RBFN for the whole learning process. However, both algorithms have converged to the given minimum error. For the classification, it shows that PSO-RBFN is better than BP-RBFN with 95.05% compared to 91.27%.

Figure 4 illustrates the learning process for both algorithms. In PSO-RBFN, 20 particles work together to find the lowest error (gbest) at each iteration and consistently reducing the error. While in BP-RBFN, it seems that the error is decreasing at each iteration, and the learning is discontinued at a specified condition.

Table 4 Result of BP-RBFN and PSO-RBFN on Balloon dataset

	BP-RBFN		PSO-RBFN	
	Train	Test	Train	Test
Learning Iteration	20000	1	3161	1
Error Convergence	0.01212	0.23767	0.0049934	0.16599
Classification (%)	91.27	75.41	95.05	78.95

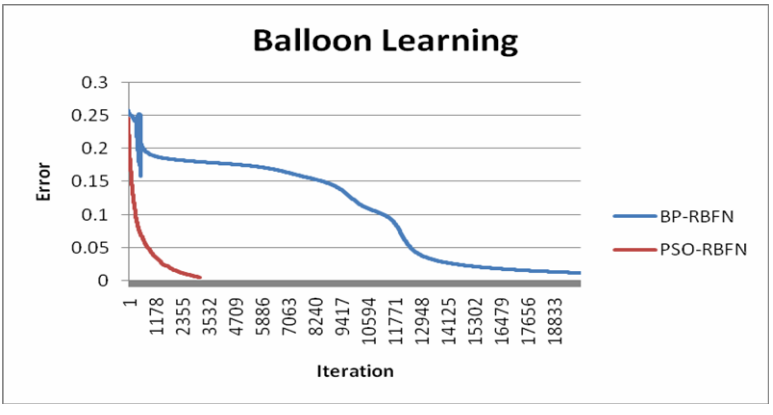

Fig. 4 Convergence of Balloon dataset

7.2.3 Cancer Dataset

The purpose of the breast cancer data set is to classify a tumour as either benign or malignant based on cell descriptions gathered by microscopic examination. It contains 9 attributes and 699 examples of which 485 are benign examples and 241 are malignant examples. The first 349 examples of the whole data set were used for training, the following 175 examples for validation, and the final 175 examples for testing [8]. The ending conditions of PSO-RBFN are set to minimum error of 0.005 or maximum iteration of 10000. Alternatively, the stopping conditions for BP-RBFN are set to a minimum error of 0.005 or maximum iteration of 20000 has been achieved.

Table 5 Result of BP-RBFN and PSO-RBFN on Cancer dataset

	BP-RBFN		PSO-RBFN	
	Train	Test	Train	Test
Learning Iteration	20000	1	10000	1
Error Convergence	0.03417	0.27333	0.0181167	0.27464
Classification (%)	92.80	70.37	97.65	71.77

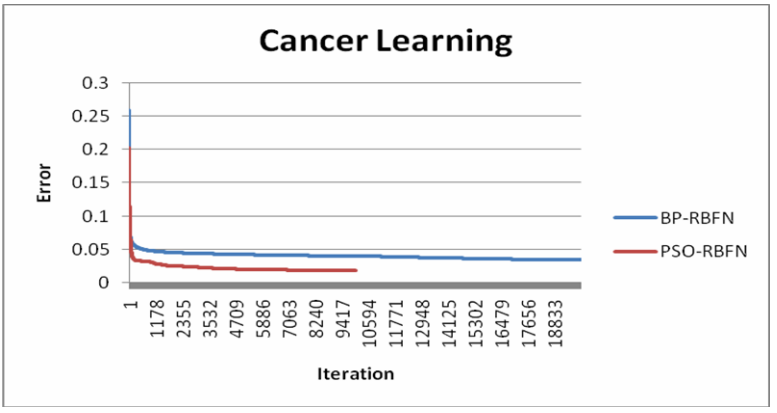

Fig. 5 Convergence of Cancer dataset

In Cancer learning process, from Table 5 shows PSO-RBFN takes 10000 iterations compared to 20000 iterations in BP-RBFN to converge. In this experiment, PSO-RBFN is managed to converge at iteration 10000, while BP-RBFN converges at a maximum iteration of 20000. Table 5 illustrates that PSO-RBFN is better than BP-RBFN with an accuracy of 97.65% and 92.80%. Figure 5 shows PSO-RBFN significantly reduce the error with small number of iterations compared to BP-RBFN.

7.2.4 Iris Dataset

The Iris dataset is used for classifying all the information into three classes which are iris setosa, iris versicolor, and iris virginica. The classification is based on its four input patterns which are sepal length, sepal width, petal length and petal width. Each class refers to type of iris plant contain 50 instances. For Iris dataset, the minimum error of PSO-RBFN is set to 0.05 or maximum iteration of 10000. While, the minimum error for BP-RBFN is set to 0.05 or the network has reached maximum iteration of 20000. Table 6 shows that BP-RBFN is better than PSO-RBFN with an accuracy of 95.66% compared to 95.48%. However, PSO-RBFN converges faster at 3774 iterations compared to 10162 iterations in BP-RBFN.

For Iris learning, both algorithms converge using the maximum number of pre-specified iteration. PSO-RBFN takes 3774 iterations to converge at a minimum error of 0.0499949 while minimum error for BP-RBFN is 0.05000 with 10162

Table 6 Result of BP-RBFN and PSO-RBFN on Iris dataset

	BP-RBFN		PSO-RBFN	
	Train	Test	Train	Test
Learning Iteration	10162	1	3774	1
Error Convergence	0.05000	0.04205	0.0499949	0.03999
Classification (%)	95.66	95.78	95.48	95.64

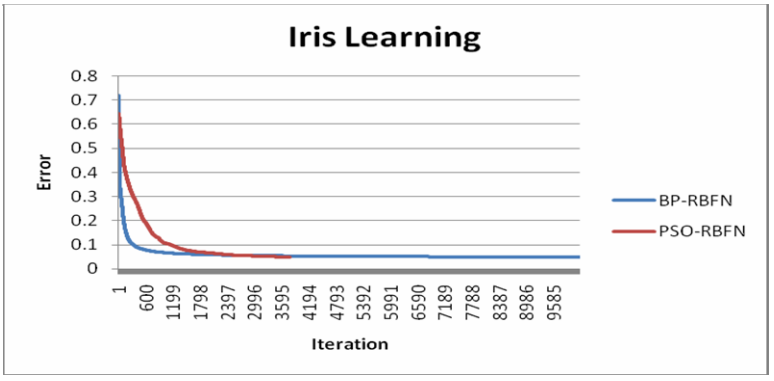

Fig. 6 Convergence of Iris dataset

iterations. Figure 6 shows that PSO-RBFN reduces the error with minimum iterations compared to BP-RBFN.

7.2.5 Ionosphere Dataset

This radar data was collected by a system in Goose Bay, Labrador. This system consists of a phased array of 16 high-frequency antennas with a total transmitted power on the order of 6.4 kilowatts. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not; their signals pass through the ionosphere. For Ionosphere problems, the stopping conditions for BP-RBFN is minimum error of 0.05 or maximum iteration of 20000. The minimum error of PSO-RBFN is 0.05 or maximum iteration of 10000. The experimental results for PSO-based RBFN and BP-based RBFN are shown in Table 7 and Figure 7.

Table 7 Result of BP-RBFN and PSO-RBFN on Ionosphere dataset

	BP-RBFN		PSO-RBFN	
	Train	Test	Train	Test
Learning Iteration	20000	1	5888	1
Error Convergence	0.18884	0.23633	0.0499999	0.01592
Classification (%)	62.27	62.71	87.24	90.70

In Ionosphere learning process, Table 7 shows PSO-RBFN takes 5888 iterations compared to 20000 iterations in BP-RBFN to converge. In this experiment, PSO-RBFN is managed to converge using minimum error at iteration of 5888, while BP-RBFN trapped at the local minima and converges at a maximum iteration of 20000. For the correct classification percentage, it shows that PSO-RBFN result is better than BP-RBFN with 87.24% compared to 62.27%. Figure 7 shows PSO-RBFN significantly reduce the error with small number of iterations compared to BP-RBFN. The results for this data are not promising for BP-RBFN since

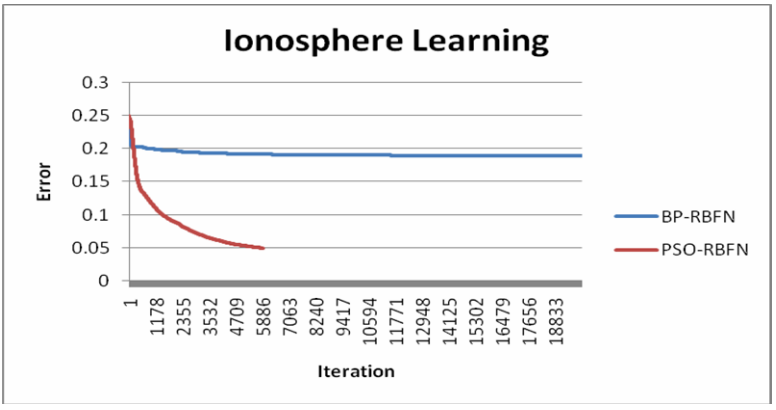

Fig. 7 Convergence of Ionosphere dataset

it depends on the data and repeatedly traps in local minima. The local minima problem in BP-RBFN algorithm is usually caused by disharmony adjustments between centers and weights of RBF Network. To solve this problem, the error function has been modified as suggested [17].

8 Validation Results

In artificial neural network methodology, data samples are divided into three sets; training, validation and testing in order to obtain a network which is capable of generalizing and performing well with new cases. There is no precise rule on the optimum size of the three sets of data, although authors agree that the training set must be the largest. Validations are motivated by two fundamental problems either in model selection or in performance estimation.

Table 8 Validation Result of BP-RBFN and PSO-RBFN on all dataset

Dataset	BP-RBFN		PSO-RBFN	
	Train	Test	Train	Test
XOR	0.11332	0.32864	0.00494071	0.47354
Balloon	0.06004	0.33155	0.00499450	0.27348
Cancer	0.03046	0.04233	0.00541208	0.02733
Iris	0.05000	0.06227	0.0499760	0.05792
Ionosphere	0.20743	0.22588	0.0499953	0.06325

To create a N-fold partition of the dataset we simplifies that for each of N experiments, use N-1 folds for training and the remaining one for testing and the true error is estimated as the average error rate. The results demonstrate the evaluation of our algorithms with respect to the convergence rate on the training and testing

dataset. These results for BP-RBFN and PSO-RBFN on all dataset are shown in Table 8.

On the whole data, the experiments showed that PSO gives high performance for RBF Network training. PSO-RBFN reduces the error with minimum iterations compared to BP-RBFN.

9 Comparison between PSO-RBF Network and BP-RBF Network

This analysis is carried out to compare the results between PSO-RBFN and BP-RBFN. The learning patterns for both algorithms in both experiments are compared using a five datasets. The classification results for all datasets are shown in Figure 8.

For Balloon, Cancer and Ionosphere dataset, the results show that PSO-RBFN is better in terms of convergence rate and correct classification. PSO-RBFN converges in a short time with high classification rates. For XOR and Iris dataset, both algorithms converge to the solution within specified minimum error; it shows that at this time, BP-RBFN classifications are better than PSO-RBFN. But in terms of convergence rate, it shows that PSO-RBFN is better than BP-RBFN, and PSO-RBFN significantly reduces the error with minimum iterations.

For overall performance, the experiments show that PSO-RBFN produces feasible results in terms of convergence rate and classification accuracy.

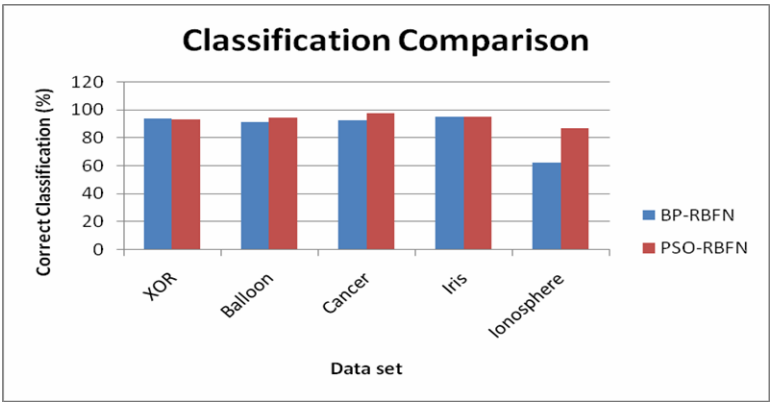

Fig. 8 Comparison of Classification Accuracy of PSO-RBFN and BP-RBFN

10 Conclusion

This paper proposes PSO based Hybrid Learning of RBF Network to optimize the centers, widths and weights of network. Based on the results, it is clear that PSO-RBFN is better than BP-RBFN in term of convergence and error rate and PSO-RBFN reached optimum because it reduces the error with minimum iteration

and obtains the optimal parameters of RBF Network. In PSO-RBFN, network architecture and selection of network parameters for the dataset influence the convergence and the performance of network learning.

In this paper, both the algorithms need to be used the same network architecture. Choosing PSO-RBFN parameters also depend on the problem and dataset to be optimized. These parameters can be adjusted accordingly to achieve better optimization. However, to have better comparison, the same parameters for all five datasets have been used. For BP-RBFN, the learning rate which is critical for standard BP network is provided with a set of weight. This is to ensure the convergence time is faster with better results. Although Standard BP learning becomes faster based on those parameters, the overall process including parameters selection in BP-RBFN takes lengthy time compared to the process in PSO-RBFN.

Acknowledgment. This work is supported by Ministry of Higher Education (MOHE) under Fundamental Research Grant Scheme. Authors would like to thank Research Management Centre (RMC) Universiti Teknologi Malaysia, for the research activities and Soft Computing Research Group (SCRG) for the support and incisive comments in making this study a success.

References

1. Leonard, J.A., Kramer, M.A.: Radial basis function networks for classifying process faults. Control Systems Magazine 11(3), 31–38 (1991)
2. Liu, Y., Zheng, Q., Shi, Z., Chen, J.: Training radial basis function networks with particle swarms. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3173, pp. 317–322. Springer, Heidelberg (2004)
3. Chen, J., Qin, Z.: Training RBF Neural Networks with PSO and Improved Subtractive Clustering Algorithms. In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4233, pp. 1148–1155. Springer, Heidelberg (2006)
4. Cui, X.H., Polok, T.E.: Document Clustering Using Particle Swarm Optimization. In: Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005, pp. 185–191 (2005)
5. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, vol. IV, pp. 1942–1948 (1995)
6. Shi, Y., Eberhart, R.C.: A Modified Particle Swarm. In: Proceedings of 1998 IEEE International Conference on Evolutionary Computation, Piscataway, NJ, pp. 1945–1950 (1998)
7. Wang, X.G., Tang, Z., Tamura, H., Ishii, M.: A modified error function for the backpropagation algorithm. Neurocomputing 57, 477–488 (2004)
8. Zhang, C., Shao, H., Li, Y.: Particle Swarm Optimization for Evolving Artificial Neural Network. In: 2000 IEEE International Conference on Systems, Man, and Cybernetics, vol. 4, pp. 2487–2490 (2000)
9. Al-kazemi, B., Mohan, C.K.: Training Feedforward Neural Network Using Multiphase Particle Swarm Optimization. In: Proceeding of the 9th International Conference on Neural Information Processing, vol. 5, pp. 2615–2619 (2002)
10. Blake, C., Merz, C.J.: UCI Repository of Machine Learning Databases (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`

11. Qin, Z., Chen, J., Liu, Y., Lu, J.: Evolving RBF Neural Networks for Pattern Classification. In: Hao, Y., Liu, J., Wang, Y.-P., Cheung, Y.-m., Yin, H., Jiao, L., Ma, J., Jiao, Y.-C. (eds.) CIS 2005. LNCS, vol. 3801, pp. 957–964. Springer, Heidelberg (2005)
12. Liu, Y., Zheng, Q., Shi, Z., Chen, J.: Training Radial Basis Function Networks with Particle Swarms. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3173, pp. 317–322. Springer, Heidelberg (2004)
13. Chen, J., Qin, Z.: Training RBF neural networks with PSO and improved subtractive clustering algorithms. In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4233, pp. 1148–1155. Springer, Heidelberg (2006)
14. Vakil-Baghmishah, M.T., Pavesic, N.: Training RBF networks with selective backpropagation. Nerocomputing ScienceDirect. 62, 39–64 (2004)
15. Gao, H., Feng, B., Hou, Y., Zhu, L.: Training RBF neural network with hybrid particle swarm optimization. In: Wang, J., Yi, Z., **urada, J.M., Lu, B.**-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 577–583. Springer, Heidelberg (2006)
16. Sun, J., Xu, W., Liu, J.: Training RBF neural network via quantum-behaved particle swarm optimization. In: King, I., Wang, J., Chan, L.-W., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4233, pp. 1156–1163. Springer, Heidelberg (2006)
17. Wang, X.G., Tang, Z., Tamura, H., Ishii, M.: A modified error function for the backpropagation algorithm. Neurocomputing 57, 477–488 (2004)