A Self-Adaptive RBF Neural Network Classifier for Transformer Fault Analysis

Ke Meng, *Member, IEEE*, Zhao Yang Dong, *Senior Member, IEEE*, Dian Hui Wang, *Senior Member, IEEE*, and Kit Po Wong, *Fellow, IEEE*

***Abstract*—A new hybrid self-adaptive training approach-based radial basis function (RBF) neural network for power transformer fault diagnosis is presented in this paper. The proposed method is able to generate RBF neural network models based on fuzzy c-means (FCM) and quantum-inspired particle swarm optimization (QPSO), which can automatically configure network structure and obtain model parameters. With these methods, the number of neuron, centers and radii of hidden layer activated functions, as well as output connection weights can be automatically calculated. This learning method is proved to be effective by applying the RBF neural network in the classification of five benchmark testing data sets, and power transformer fault data set. The results clearly demonstrated the improved classification accuracy compared with other alternatives and showed that it can be used as a reliable tool for power transformer fault analysis.**

***Index Terms*—Computational methods, particle swarm optimization, power transformer fault diagnosis, radial basis function (RBF) neural network.**

I. Introduction

POWER system stability depends on the reliable operation of various individual components within the network. Power transformer is one of the necessary and significant units in the transmission and distribution levels of a power system; however, it is subjected to many different types of faults which may cause interruptions in power supply, consequently result in serious economic losses as well as social impacts. As a result, effective fault diagnosis approaches are warrant to detect and analyze the power transformer internal faults, and eliminate the associated impacts to the lowest possible level.

To understand the phenomena of transformer faults, different methods have been suggested and reported [1]–[9]. Dissolved gas analysis (DGA) method has been widely accepted and used in the internal faults diagnosis of power transformers for years [1]. Because once the internal faults occur, the rate of cellulose and oil degradation will increase significantly. These fault gases are produced by degradation of transformer oil and solid insulating materials. However, the key gas and ratio techniques are mainly built on the knowledge/experience gained from previous fault diagnosis, which might vary from utility to utility, and no general mathematical rule can be summarized. Fortunately, the artificial intelligence-based methods provide a perfect solution to the deficiency, which include expert systems [2], fuzzy logics [3], [4], artificial neural networks (ANNs) [5]–[8], and support vector machine (SVM) [9]. The expert systems and fuzzy logic approaches involve human expertise, and have many successful applications. However, the major challenge is how to acquire expert experience, and transform this prior human knowledge into decision rules and membership functions. Furthermore, the final accuracy largely depends on the completeness and representation of accumulated human experience/knowledge. ANNs and SVMs have attracted widespread attention due to the mature theory background as well as satisfactory analysis performance. ANNs for power transformer fault analysis have been shown as being able to give effective and reliable performance [5], [7]. The detailed studies of ANNs and SVMs for transformer fault diagnosis are carried out in the following sections.

Due to a number of advantages compared with other types of ANNs, including better approximation ability, simpler network structure, and faster learning speed, radial basis function (RBF) neural network is continuously increasing its popularity in many fields. RBF neural network was first proposed in the late 1980s [10]. Normally, it forms a special architecture, which consists of three layers, namely input, hidden, and output layer. Each hidden layer node adopts a radial activated function, and output nodes implement a weighted sum of hidden unit outputs. The structure of multi-input and single-output (MISO) RBF neural network is represented in Fig. 1. Theoretically, RBF neural networks can approximate any continuous function defined on a compact set to any prescribed degree of accuracy by sufficiently expanding the networks structure [11]. Currently, the majority of training schemes for RBF neural networks can be classified as one-phase learning or two-stage training.

1) ***One-phase learning***. With this scheme, the parameters of hidden layer kernel functions and the output connection weights are adjusted simultaneously with one objective function, which is minimization of network output errors.
2) ***Two-stage training***. Two layers of RBF neural network are trained separately; firstly the parameters of hidden layer kernel functions are determined in self-organizing manner

Manuscript received May 26, 2009; revised October 29, 2009. First published February 22, 2010; current version published July 21, 2010. Paper no. TPWRS-00379-2009.

K. Meng and Z. Y. Dong are with the Department of Electrical Engineering, The Hong Kong Polytechnic University, Hong Kong (e-mail: eekemeng@polyu.edu.hk; eezydong@polyu.edu.hk; zydong@ieee.org).

D. H. Wang is with the Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, VIC 3086, Australia (e-mail: dh.wang@latrobe.edu.au).

K. P. Wong is with the Department of Electrical Engineering, The Hong Kong Polytechnic University, Hong Kong, and also with the School of Electrical, Electronic, and Computer Engineering, University of Western Australia, Perth, Australia (e-mail: eekpwong@polyu.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TPWRS.2010.2040491

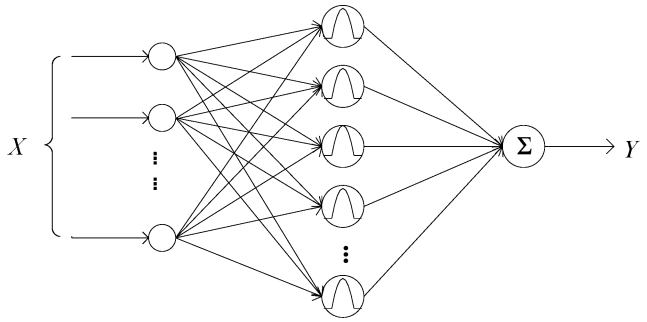

Fig. 1. Typical MISO RBF neural network structure.

or assigned randomly, followed by the output connection weights adjusted through various supervised techniques.

However, how to choose suitable network structures has always been the problem which limits their wider applications. A small network may never converge or take very long time to converge, while a large one may converge very fast but lack generalization ability. How to select the suitable structure for neural network is still largely built on a trial and error basis. Consequently, the key element for their further acceptance is to develop a scheme which is able to automatically generate optimally structured neural networks by any given sample, which ensures both strong learning and generalization capabilities. However, one-phase learning scheme requires strong ability for algorithm itself, due to the difficulties involved in high dimension optimization problems, therefore comparatively two-stage training method is preferable.

Normally, in a two-stage training approach for RBF neural networks, the kernel function centers are determined by clustering approach. The popular clustering techniques are the hard c-means (HCM) and fuzzy c-means (FCM), which belong to unsupervised clustering approaches on the basis of computing distance [12]. FCM is a data clustering technique in which each data point belongs to a cluster, providing a method that shows how to group the data that populate multidimensional space into a specific number of different clusters [13]. It starts with a set of initial guesses for cluster centers, which are intended to mark the mean location of each cluster. With these initial cluster centers, it assigns every data point a membership grade to each cluster. By iteratively updating the cluster centers as well as the membership grades for each data point, FCM moves the cluster centers to the right location within the data set. However, it is worth noting that the unsupervised clustering algorithms are not preferable in determining the number of hidden layer clusters, especially when dealing with complex data distribution. In this paper, based on FCM, a fuzzy clustering method is adopted.

When it comes to the calculation of output layer connection weights, existing approaches mainly fall into three categories: matrix inversion technique [14], gradient-based training method [15], and global optimization approaches [16], [17]. The matrix inversion method mainly deals with linear weights optimization and it often suffers from the "curse of dimensionality" problem when the network size is very large and sometimes the inverse matrix does not exist at all. The gradient-based training methods are quite fast provided that the gradient information of the error surface is known; however, most of them often easily fall into local optima. The global optimization techniques are mainly the evolutionary algorithms (EAs), such as genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO). They can be used to train neural networks due to their global optimization capabilities. Although these methods do not always guarantee discovering the globally optimal solutions in finite time, they often provide reasonably good solutions. GA ensures population evolves and solution changes continually, however, they often lack a strong capacity of producing the best offspring individuals and may experience slow convergence when approaching global optimum. DE is with no doubt a very powerful method, but the greedy updating scheme and intrinsic differential property may lead the evolution process be trapped by local optima [18], [19]. PSO converges very quickly, but has a slow fine-tuning ability of the solution. Once it gets stuck into local optima, it is very hard to jump off it. Generally speaking, each approach has its own advantages and drawbacks. Many attempts try to merge some of their individual implementations together into a hybrid algorithm to overcome individual disadvantages and to maximize their various advantages. Compared with other techniques, PSO is computationally inexpensive in terms of memory and speed. The most attractive features of PSO can be summarized as, simple concept, easy implementation, fast speed, and robust capability [20]. In this paper, several concepts from quantum computing and immunology are adopted to improve the search capability of conventional PSO, and then quantum-inspired particle swarm optimization (QPSO) [21] is introduced.

The paper is organized as follows: after introduction, FCM and QPSO will be reviewed for completeness, followed by the detailed steps of proposed approach. Then numerical simulations are presented and this methodology is tested with five benchmark classification data sets, and power transformer fault data set. Conclusions and further developments are given in the last section.

II. Configurations of RBF Neural Network

A. RBF Neural Network Structure

A normal RBF neural network is a fully connected network with three layers. The performance through the input layer to the output layer compete the task of classification by dividing the whole input space into several subspaces in the form of hyperellipsoid. Sometimes when the distribution of input data is very sophisticated, the hyperellipsoid alone cannot deliver satisfactory performance. As a consequence, a composite structure of RBF neural network is used, shown in Fig. 2, the combination of hyperellipsoids and hyperplanes is used for partitioning the input space; therefore, the input space can be flexibly divided to enhance its classification ability [22]. Clearly, the neural network learning procedure consists of two parts, nonlinear part and linear part, which will be discussed further in the following sections.

The given data set is $\{\boldsymbol{x}_i, y_i\}_{i=1}^{n}$, $\boldsymbol{x}_i \in \mathbb{R}^s$ is the input vector which denotes a pattern to be classified, and $y_i \in \{\pm 1\}$ is the associated desired output which denotes relevant class label. A typical network with $k+s$ hidden layer neurons, the output of

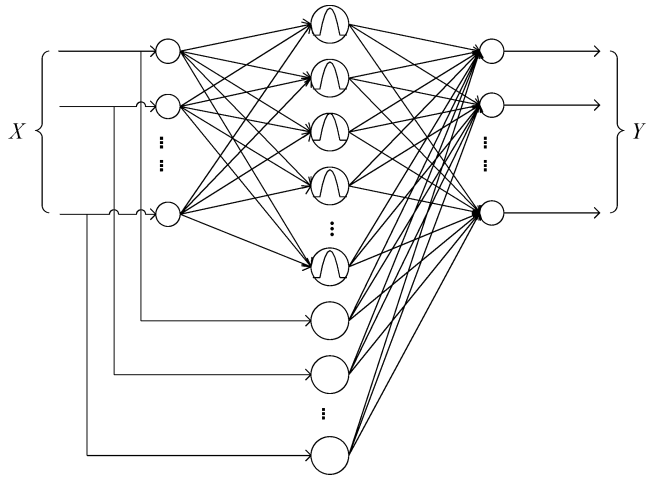

Fig. 2. Hybrid RBF neural network structure.

hidden layer jth Gauss kernel neuron for input vector $\boldsymbol{x}_i$ can be expressed as

$$g_j(\boldsymbol{x}_i) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{v}_j\|_2^2}{2\sigma_j^2}\right), \quad \begin{cases} i = 1, 2, \cdots, n \\ j = 1, 2, \cdots, k \end{cases} \tag{1}$$

where $\boldsymbol{v}_j$ are the kernel centers, and σ_j is the kernel width.

The output for input vector $\boldsymbol{x}_i$ can be calculated by

$$y(\boldsymbol{x}_i) = \sum_{j=1}^{k} \omega_j g_j(\boldsymbol{x}_i) + \sum_{j^*=1}^{s} \omega_{j^*} x_{j^* i} + e_i \tag{2}$$

where ω are the output connection weights, and e_i is the output error bias.

B. Fuzzy C-Means

The number of hidden layer neurons is a major problem for neural networks, also a matter for experimentation. For some clustering methods, including FCM, the number of clusters k needs to be given in advance. Until now there are two main options: validity measures and compatible clustering merging. In the former one, samples must be clustered several times, each time with a different number of clusters, $k \in [2, n]$. The latter one starts with a large number of clusters, then proceeding by gradually merging similar clusters to obtain fewer clusters. Here k should be large enough so that the nonlinearity of system could be captured accurately. However, cluster number essentially depends on nonlinear extent of given data sample. When lacking of enough prior knowledge, trial and error method is usually used to choose proper value step by step, therefore, the computation burden is undoubtedly aggravated.

In this paper, based on FCM, a fuzzy clustering approach is adopted, where we can specify the range of hidden layer neurons. Let $\boldsymbol{x}_i \in \mathbb{R}^s$ be the data of patterns represented in feature space. Start cluster number at $k = n/2$, then judge whether new center should be added according to network performance. If the result is not satisfactory, choosing a new cluster center $\boldsymbol{v}_{k+1}$ from the remaining samples which is different from the existed ones $[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k]$. Update membership matrix, start algorithm with new centers. Repeat the above steps until satisfactory result or maximum neuron $k < n$ is reached. Here this cluster range $[n/2, n)$ is an experience value.

The FCM algorithm performs clustering by solving

$$Min.\ J_m(\boldsymbol{u}, \boldsymbol{v}; \boldsymbol{x}) = \sum_{i=1}^{n}\sum_{j=1}^{k} u_{ji}^m \|\boldsymbol{x}_i - \boldsymbol{v}_j\|_2^2 \tag{3}$$

$$s.t. \begin{cases} \boldsymbol{u} = [u_{ji}], u_{ji} \in [0, 1] \\ \sum\limits_{j=1}^{k} u_{ji} = 1, \sum\limits_{i=1}^{n} u_{ji} > 0 \end{cases}, \begin{cases} i = 1, 2, \cdots, n \\ j = 1, 2, \cdots, k. \end{cases} \tag{4}$$

Step 1) For given data set, fix $k \in [n/2, n)$; admissible error $\varepsilon > 0$; initial cluster center $\boldsymbol{v}_0$; fuzzification constant m, which denotes the degree of fuzzification $(1 < m < \infty)$. If $m \to 1$, the membership degrees of any pattern to any cluster tend to be either 0 or 1, and approaches hard c-means; on the other hand; if $m \to \infty$, the membership degrees of any pattern to any cluster tend to be equal to $1/k$, thus producing the highest level of fuzziness. Although no theoretically optimal value has been determined, the most common choice is $m = 2$.

Step 2) Calculate $\boldsymbol{u}(t) = [u_{ji}(t)]$. $u_{ji}(t)$ is the membership value of vector $\boldsymbol{x}_i$ to clusters centre $\boldsymbol{v}_j$; $d_{ji} = \|\boldsymbol{x}_i - \boldsymbol{v}_j\|_2$ is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{v}_j$:

$$u_{ji}(t) = \frac{1}{\sum\limits_{r=1}^{k} \left\{ \left[\frac{d_{ji}(t-1)}{d_{ri}(t-1)} \right]^{\frac{2}{m-1}} \right\}}. \tag{5}$$

Step 3) Calculate $\boldsymbol{v}(t)$. And $\boldsymbol{v} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k]$ is the array of clusters centers; for $\forall j$

$$\boldsymbol{v}_j(t) = \frac{\sum\limits_{i=1}^{n} [u_{ji}(t)]^m \boldsymbol{x}_i}{\sum\limits_{i=1}^{n} [u_{ji}(t)]^m}. \tag{6}$$

Step 4) If maximum of iterations is reached or stop criterion $\|\boldsymbol{v}(t) - \boldsymbol{v}(t-1)\| \le \varepsilon$ is met, then stop; otherwise, jump to step 2.

Another critical component of kernel-based learning method is the choice of appropriate cluster radii. Experimentally, it is often treated as a constant according to the farthest distance between cluster centers and the number of clusters in a practical problem to be solved. In this paper, the cluster radii are calculated by QPSO, together with the output connection weights.

C. Quantum-Inspired Particle Swarm Optimization

QPSO has stronger search ability and quicker convergence speed since it not only introduces the concepts of quantum bit and rotation gate, but also the distinguished implementations of self-adaptive probability selection and chaotic sequences mutation. In QPSO, the state of a particle is depicted by quantum bit and angle, instead of particle position and velocity in the conventional PSO. These concepts are defined as below.

Quantum bit, the smallest unit in QPSO, is defined as a pair of numbers:

$$\begin{bmatrix}\alpha_{ji}(t)\\ \beta_{ji}(t)\end{bmatrix}, \begin{cases} j=1,2,\cdots,m \\ i=1,2,\cdots,n. \end{cases} \tag{7}$$

The modulus $|\alpha_{ji}(t)|^2$ and $|\beta_{ji}(t)|^2$ give the probabilities that the quantum bit exists in states "0" and "1", respectively, which should satisfy

$$|\alpha_{ji}(t)|^2 + |\beta_{ji}(t)|^2 = 1. \tag{8}$$

A string of quantum bits consist of a quantum bit individual, which can be defined as

$$\begin{aligned}\boldsymbol{q}_j(t) &= \begin{bmatrix}\alpha_{j1}(t),\cdots,\alpha_{ji}(t),\cdots,\alpha_{jn}(t)\\ \beta_{j1}(t),\cdots,\beta_{ji}(t),\cdots,\beta_{jn}(t)\end{bmatrix} \\ &= [q_{j1}(t),\cdots,q_{ji}(t),\cdots,q_{jn}(t)]. \end{aligned}\tag{9}$$

A quantum bit is able to represent a linear superposition of all possible solutions by its probabilistic representations [23]–[25].

Because of the normalization condition, the ***quantum angle*** can be represented as

$$\begin{cases} |q_{ji}(t)\rangle = \cos\theta_{ji}(t)|0\rangle + \sin\theta_{ji}(t)|1\rangle \\ \theta_{ji}(t) = \arctan\frac{\beta_{ji}(t)}{\alpha_{ji}(t)}. \end{cases} \tag{10}$$

Therefore, the quantum bit individual can be represented in the form of quantum angle:

$$\begin{aligned}\boldsymbol{q}_j(t) &= [q_{j1}(t),\cdots,q_{ji}(t),\cdots,q_{jn}(t)] \\ \downarrow \quad &\quad \downarrow \quad \downarrow \quad \downarrow \\ \boldsymbol{\theta}_j(t) &= [\theta_{j1}(t),\cdots,\theta_{ji}(t),\cdots,\theta_{jn}(t)]. \end{aligned}\tag{11}$$

The fundamental update mechanism of QPSO is evolving quantum bits and angles, by which the updated quantum bits should still satisfy the normalization condition. The quantum rotation gate update equation could be calculated by

$$\begin{aligned}\Delta\boldsymbol{\theta}_j(t+1) = \omega\cdot\Delta\boldsymbol{\theta}_j(t) + \varphi\cdot r_1\cdot[\boldsymbol{\theta}_{pb}(t) - \boldsymbol{\theta}_j(t)] \\ +\eta\cdot r_2\cdot[\boldsymbol{\theta}_{gb}(t) - \boldsymbol{\theta}_j(t)]\end{aligned}\tag{12}$$

where

φ,η	cognitive and social components;
r_1, r_2	random numbers in (0,1);
ω	inertia weight;
$\Delta\boldsymbol{\theta}_j$	angles change;
$\boldsymbol{\theta}_j$	current angles;
$\boldsymbol{\theta}_{pb}$	local best angles;
$\boldsymbol{\theta}_{gb}$	global best angles.

$$\begin{bmatrix}\alpha_{ji}(t+1)\\ \beta_{ji}(t+1)\end{bmatrix} = \begin{bmatrix}\cos\Delta\theta_{ji}(t+1) & -\sin\Delta\theta_{ji}(t+1)\\ \sin\Delta\theta_{ji}(t+1) & \cos\Delta\theta_{ji}(t+1)\end{bmatrix} \times \begin{bmatrix}\alpha_{ji}(t)\\ \beta_{ji}(t)\end{bmatrix}. \tag{13}$$

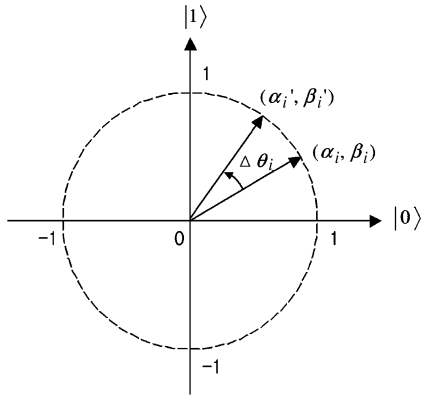

Fig. 3. Quantum rotation gate.

And quantum rotation gate can be illustrated in Fig. 3 [23].

Although the quantum bit and rotation gate representation has better characteristics of population diversity, the premature convergence problem could still appear. In order to address this shortcoming, the self-adaptive probability selection and chaotic sequences mutation are adopted. In the evolutionary process, we observe the global best individual, suppose it has not changed for iterations, then two quality discriminators will be introduced, affinity and concentration. The affinity value reflects the quality of quantum individual to the problem, and the concentration indicates the proportion of similar individuals in current population.

The ***individual affinity*** value can be defined as follows. We calculate the fitness value of every individual in current population and rearranged the population in terms of the fitness value in ascending sequence. The affinity is designed by using location index of quantum bit individual.

$$As\left[\boldsymbol{q}_j(t)\right] = r\cdot(1-r)^{j-1} \tag{14}$$

where r is the random number in (0,1).

The individuals should be returned to their original locations. The most attractive feature of this definition is that the affinity value is only relevant to the location index rather than the real fitness value.

The ***individual concentration*** can be defined as

$$Cs\left[\boldsymbol{q}_j(t)\right] = \frac{\sum_{a=1}^{m} Ks\left[\boldsymbol{q}_j(t),\boldsymbol{q}_a(t)\right]}{m} \tag{15}$$

$$Ks\left[\boldsymbol{q}_j(t),\boldsymbol{q}_a(t)\right] = \begin{cases} 1, & if\ \left\|\boldsymbol{q}_j(t),\boldsymbol{q}_a(t)\right\|_2 \le l \\ 0, & \text{otherwise.}\end{cases} \tag{16}$$

Then, a roulette selection is implemented based on the computed selection probabilities. This allocates every quantum bit individual a probability of being selected proportionally according to selection probabilities.

The ***selection probabilities*** can be expressed as

$$Ps\left[\boldsymbol{q}_j(t)\right] = \frac{As\left[\boldsymbol{q}_j(t)\right]/Cs\left[\boldsymbol{q}_j(t)\right]}{\sum_{j=1}^{m}\left\{As\left[\boldsymbol{q}_j(t)\right]/Cs\left[\boldsymbol{q}_j(t)\right]\right\}}. \tag{17}$$

Therefore, the quantum bit individuals can be selected according to individuals selection probabilities, guaranteeing that the individuals having high affinity values could be selected; and the one that has high concentration value could be rejected.

After that, the chaotic sequences mutation is implemented. A widely used system evidencing chaotic behavior is the logistic map, which can be expressed as follows:

$$g(t+1) = 4g(t) \cdot [1 - g(t)]. \tag{18}$$

The mutation implementation can be defined as

$$\boldsymbol{q}'(t) = \boldsymbol{q}(t) \cdot \left[1 \pm r \cdot \left(1 - \frac{t}{T}\right) \cdot \boldsymbol{g}(t)\right]. \tag{19}$$

Notice that there is a user-defined control variable r, mutation control constant. Selection of this value largely depends on practical problem. Here according to our experience, the range [0.1, 0.5] is a suitable option.

D. Steps of the Proposed Algorithm

The proposed learning scheme for RBF neural networks can be described as follows.

Step 1) Each data set is randomly and proportionally divided into three parts (training/validation/testing), which are selected to evaluate the proposed network performance.

Step 2) Cluster range $k \in [n/2, n)$, initial cluster center $\boldsymbol{v}_0$, admissible error for FCM $\varepsilon > 0$; maximum iteration $T = 2000$; cognitive and social components $\varphi = 1.65$ and $\eta = 1.81$; inertia weight $\omega = 0.72$.

Step 3) Iteratively calculate and update the membership matrix and cluster center matrix in FCM. If the admissible error is reached or maximum iteration is met, the processes of clustering stop.

Step 4) Calculate output connection weights and cluster radii by QPSO, and save the best feasible solutions with training and validation data sets.

In order to compare the performance, the following criteria is adopted, namely root mean squared error (RMSE):

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y}_i)^2} \tag{20}$$

where y_i is the true value, and $\bar{y}_i$ is the network output.

If RMSE cannot reach given accuracy, then let $k = k+1$, go to step 5; otherwise stop the algorithm and jump to step 6.

Step 5) According to the existing membership matrix $\boldsymbol{u}$, find new cluster center vector $\boldsymbol{v}_{k+1}$ by computing

$$Min. \sum_{1 \le i,j \le k, i \ne j} (u_{ni} - u_{nj}). \tag{21}$$

Based on the new centers $[\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k, \boldsymbol{v}_{k+1}]$, jumps to step 3.

Step 6) The model that has the minimum error is selected as the best model; calculate network output results of testing data.

The general steps are shown in Fig. 4.

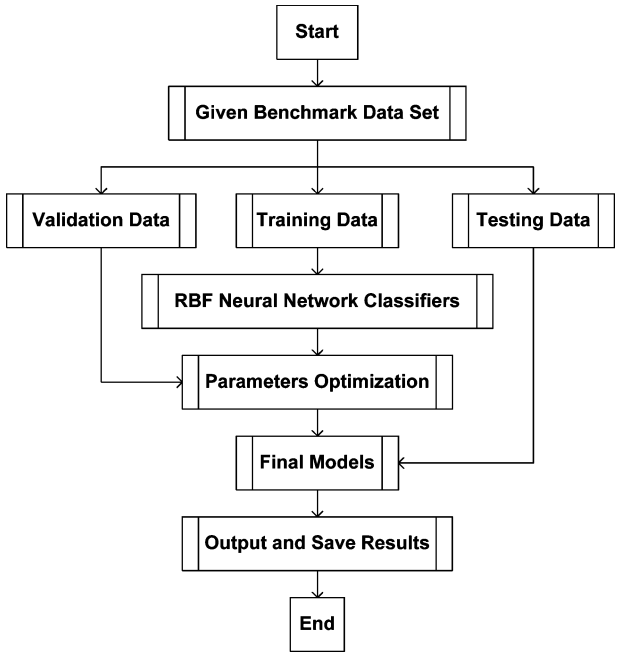

Fig. 4. Flowchart for the proposed approach.

III. Benchmark Classification Data Sets

In order to test the performance of the proposed method, five benchmark classification data sets are introduced here, namely fisher iris data set, teaching assistant evaluation data set, thyroid data set, haberman's survival data set, and glass identification data set [26].

1) Given data sets are randomly divided into three parts. The training part is used in network learning, the validation part is adopted to prevent over-fitting, and the testing part is applied to test generalization ability.
2) In order to carry out comparisons, SVM [27], RVM [28], NEWRB RBF (Con.) [29], QPSO RBF (Hyb.), FCM-QPSO RBF (Con.), and FCM-SDE RBF (Hyb.) are also used in the case studies in the following sections; see Table I.
3) Note that each approach is typically associated with a few model parameters that need to be provided before training for best performance. The parameters of the first three approaches are optimized by DE [18].
4) In the test, the last four methods are run to get the optimal hidden node number firstly, and then the remaining 49 trials are all carried with this structure.
5) For 50 trials in each case study, all the output results are rounded to the nearest integers.
6) All the programs are run on a 2.13-GHz, Intel Core 2, with 2G RAM PC.

A. Fisher Iris Data Set

The constructed RBF neural network is checked by the fisher iris data set [30], which includes 150 data with four input features and three classes.

In this case study, 75/25/50 vectors are selected randomly for training/validation/testing, respectively. Based on the data ob-

TABLE I
SELECTED APPROACHES FOR COMPARISONS

#	Methods	RBF Neural Network Structure	
		Conventional	Hybrid
1	SVM	---	---
2	RVM	---	---
3	NEWRB RBF	✓	---
4	QPSO RBF	---	✓
5	FCM-SDE RBF	---	✓
6	FCM-QPSO RBF	✓	---
7	FCM-QPSO RBF	---	✓

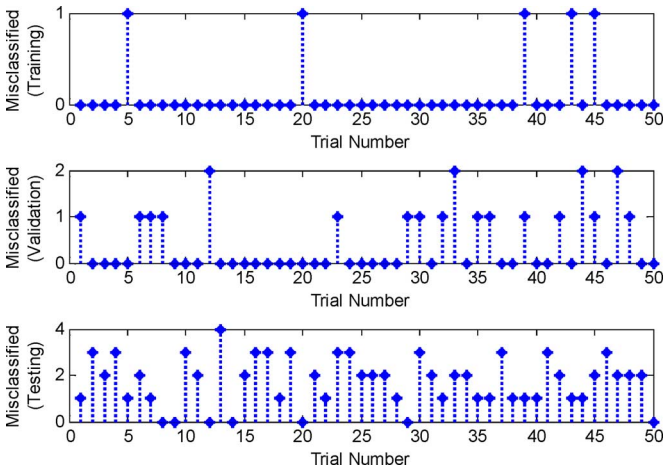

Fig. 5. Results of FCM-QPSO (Hyb.)—fisher iris data set.

TABLE II
RESULTS COMPARISON—FISHER IRIS DATA SET

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	97.94	98.64	98.78	97.83	99.64	99.49	**99.73**
Std. Dev.	0.009	0.010	0.008	0.007	0.006	0.006	**0.005**
Testing Rate (%)	96.44	**97.80**	96.64	89.44	95.84	93.80	96.56
Std. Dev.	0.024	**0.019**	0.022	0.043	0.022	0.024	0.021
Vectors / Neurons	74.90	22.50	**17**	59+4	45+4	62	43+4
Trial Time (s)	4.40	3.14	**0.06**	165.7	14.19	14.65	12.96

tained through 50 trials, the comparisons of classification performance by selected techniques are represented in Fig. 5 and Table II.

B. Teaching Assistant Evaluation Data Set

The data consists of the performance evaluations of teaching assistant assignments at the Statistics Department of the University of Wisconsin-Madison [31], which includes 151 data with five input features and three classes.

In this case study, 75/25/51 vectors are selected randomly for training/validation/testing, respectively. Based on the data obtained through 50 trials, the comparisons of classification performance by different approaches are represented in Fig. 6 and Table III.

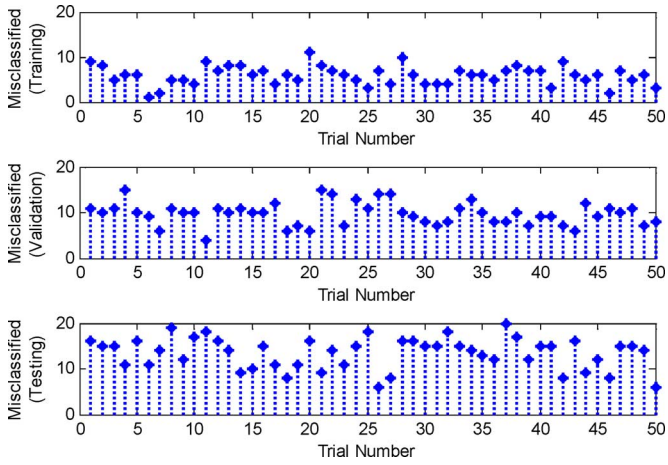

Fig. 6. Results of FCM-QPSO (Hyb.)—TA evaluation data set.

TABLE III
RESULTS COMPARISON—TA EVALUATION DATA SET

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	72.50	68.12	74.98	86.22	91.82	91.05	**92.19**
Std. Dev.	0.033	0.060	0.038	0.055	0.033	0.039	**0.030**
Testing Rate (%)	52.12	49.84	44.27	50.43	69.02	66.31	**73.69**
Std. Dev.	0.067	0.072	**0.061**	0.074	0.072	0.063	0.066
Vectors / Neurons	96.00	**29.60**	60	65+5	52+5	69	48+5
Trial Time (s)	4.04	3.32	**0.32**	214.1	16.13	16.54	14.90

C. Thyroid Data Set

The thyroid data set [32] is used to check the classification ability of the RBF neural network. The data set is used to try to predict whether a patient's thyroid to the class euthyroidism, hypothyroidism, or hyperthyroidism, which includes 215 data with five input features and three classes.

In this case study, 130/30/55 vectors are selected randomly for training/validation/testing, respectively. Based on the data obtained through 50 trials, the comparisons of the classification performance by different methods are represented in Fig. 7 and Table IV.

D. Haberman's Survival Data Set

The Haberman's survival data set [33] contains cases from a study conducted at the University of Chicago's Billings hospital on the survival of patients undergone surgery for breast cancer, which includes 306 data with three input features and two classes.

In this case study, 200/36/70 vectors are selected randomly for training/validation/testing, respectively. Based on the data obtained through 50 trials, the comparisons of the classification

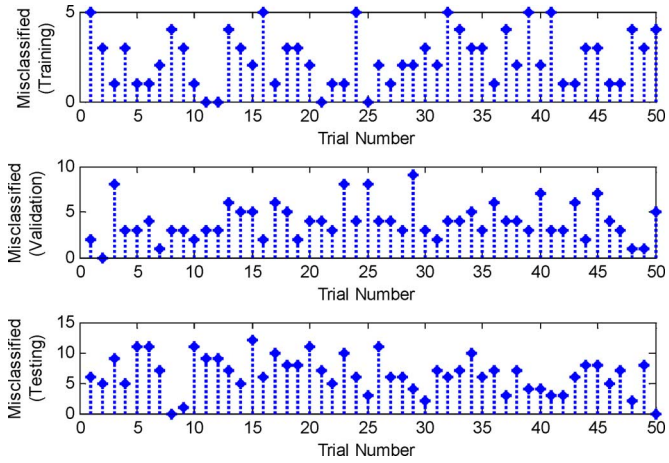

Fig. 7. Results of FCM-QPSO (Hyb.)—thyroid data set.

TABLE IV
RESULTS COMPARISON—THYROID DATA SET

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	92.42	96.04	97.05	96.00	97.66	96.76	**98.01**
Std. Dev.	0.011	0.015	0.011	0.025	**0.010**	0.020	0.012
Testing Rate (%)	87.85	87.27	82.69	78.20	87.81	81.20	**88.29**
Std. Dev.	0.040	**0.035**	0.045	0.068	0.056	0.052	0.054
Vectors / Neurons	125.9	**24.10**	42	116+5	102+5	120	97+5
Trial Time (s)	16.23	3.10	**0.17**	541.1	39.99	38.45	36.57

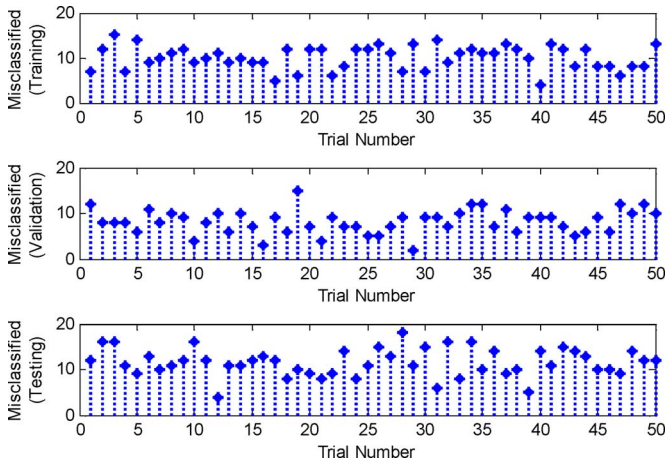

Fig. 8. Results of FCM-QPSO (Hyb.)—Haberman's survival data set.

performance by different methods are represented in Fig. 8 and Table V.

E. Glass Identification Data Set

The study of classification of types of glass was motivated by criminological investigation. This data set totally includes 214 data with nine input features and seven classes [34].

In this case study, 130/30/54 vectors are selected randomly for training/validation/testing, respectively. Based on the data

TABLE V
RESULTS COMPARISON—HABERMAN'S SURVIVAL DATA SET

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	75.65	79.27	89.77	91.12	95.50	95.88	**96.14**
Std. Dev.	0.014	0.014	0.015	0.018	0.013	0.017	**0.012**
Testing Rate (%)	73.74	73.25	57.00	72.46	79.74	79.71	**83.49**
Std. Dev.	0.055	0.045	0.057	0.065	0.049	0.043	**0.043**
Vectors / Neurons	161.7	**19.40**	131	148+3	140+3	158	136+3
Trial Time (s)	49.35	3.15	**1.09**	1562	84.87	85.48	79.30

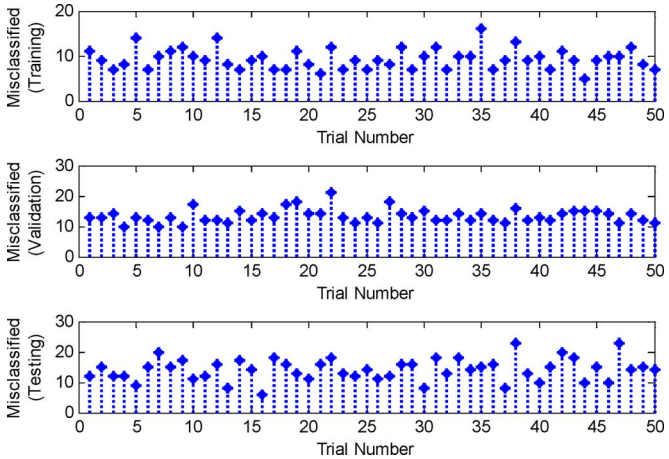

Fig. 9. Results of FCM-QPSO (Hyb.)—glass identification data set.

obtained through 50 trials, the comparisons of the classification performance by different methods are represented in Fig. 9 and Table VI.

F. Results Analysis

1) Note that each approach is typically associated with a few model parameters that need to be provided before training for best performance. Specifically, like the SVM and RVM, the Gaussian kernels and associated parameters should be provided; in addition, for SVM, the regularization parameter should be given. Compared to SVM, RVM does not need the tuning of a regularization parameter during training phase. For the normal neural network training schemes, the number of hidden layer neurons, and associated parameters need to be fixed in advance. We suggest that in order to get superior computing performance, these parameters should have problem-oriented values.
2) Obviously, as is shown in the above tables, we can conclude that the proposed method demonstrates superior performance in classification accuracy, as compared to other selected alternatives in majority of the cases. It not only achieves higher training accuracies, but also maintains stronger generalization ability. Moreover, with the hybrid structure, the required hidden layer nodes and the computation time are reduced accordingly.

TABLE VI
RESULTS COMPARISON—GLASS IDENTIFICATION DATA SET

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	88.74	87.90	90.16	88.68	92.21	90.89	**92.91**
Std. Dev.	**0.016**	0.038	0.024	0.034	0.019	0.019	0.019
Testing Rate (%)	54.81	50.59	47.96	59.82	68.33	69.62	**73.81**
Std. Dev.	0.062	0.063	0.062	0.065	0.064	**0.058**	0.067
Vectors / Neurons	143.1	**65.70**	84	106+9	92+9	114	86+9
Trial Time (s)	15.90	2.59	**0.58**	704.1	34.81	37.59	34.75

TABLE VII
POWER TRANSFORMER DATA SET

States	Training & Validation	Testing
Normal State	5 (4+1)	4
Thermal Heating	25 (20+5)	13
low-energy discharge	5 (4+1)	6
high-energy discharge	15 (12+3)	2
Overall	50 (40+10)	25

3) It can be seen that the first three approaches require much less time in the case studies, that is because we did not take into account of the time of parameters optimization. If the relevant process is added, the computational time will increase largely, especially for SVM.
4) All these results establish the ground of further application of the proposed methods for power transformer fault analysis.

IV. POWER TRANSFORMER FAULT ANALYSIS SIMULATION EXAMPLES

In power transformer fault identification, four types of states need to be identified, including normal state, thermal heating, low-energy discharge, and high-energy discharge. When abnormal phenomena such as overheating occur, the insulating transformer oil will degrade and produce many byproducts. The ratios of these combustible gases, H_2, CH_4, C_2H_2, C_2H_4, and C_2H_6, are closely related to the fault types. The pattern, degree, and trend of abnormality can be determined by monitoring the concentrations and growth of these combustible gases.

A. Power Transformer Data Set

The 75 sets of historical data of one power transformer are collected for case studies [9]. The data samples are then divided into two parts: training and validation sample (50 data set), and testing sample (25 data set). In the case studies, 40/10 data are selected from data sample for training/validation, respectively. The specific data set information is summarized in Table VII.

B. Single Classifier

Through principal component analysis (PCA) [35], we can clearly see the data distribution, shown in Fig. 10.

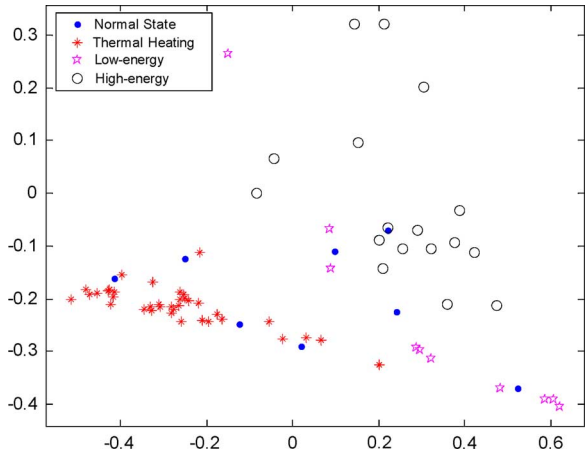

Fig. 10. Principal component projection map.

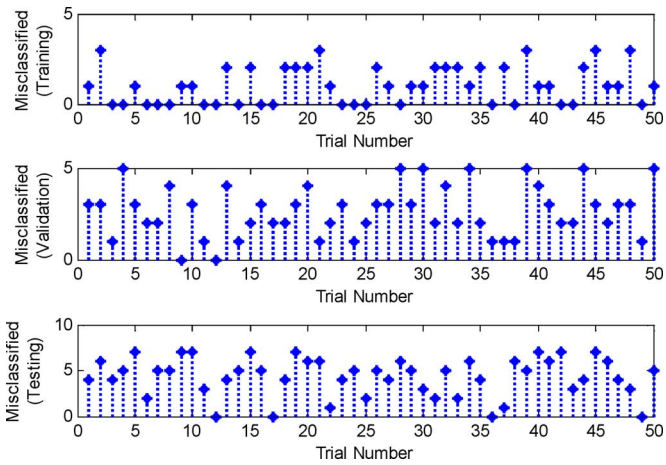

Fig. 11. Results of FCM-QPSO (Hyb.)—power transformer fault data.

From the principal component projection distribution map, it can be observed that the normal state data sparse scatter among the other data samples, which cannot easily be distinguished. And several low-energy data points distribute within the high-energy discharge data area, which may cause difficulties in classification. But the thermal heating data and the high-energy discharge data can readily be classified.

Based on the data obtained through 50 trials, the comparisons of the performance by different methods are represented in Fig. 11 and Table VIII.

In order to test the robustness of the proposed algorithm, 5% Gaussian distribution random noise was added to the original data sample. Based on the data obtained through 50 trials, the comparisons of the performance by different approaches are represented in Fig. 12 and Table IX.

The testing result shows that RBF neural network can still classify the testing samples effectively compared with the other methods. Although it does not reach the best results among all the approaches, it balances the training and testing accuracy.

C. Cascade Classifiers

In the case study, cascade classifiers is introduced, which was proposed in [9]. Based on the characteristics of different faults,

TABLE VIII
RESULTS COMPARISON—POWER TRANSFORMER FAULT DATA SET

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	86.00	**98.00**	90.00	90.80	95.38	93.95	96.30
Std. Dev.	0.000	**0.000**	0.000	0.030	0.028	0.029	0.029
Testing Rate (%)	82.00	76.00	60.00	72.40	79.84	76.56	**82.64**
Std. Dev.	**0.000**	0.000	0.000	0.095	0.085	0.088	0.082
Vectors / Neurons	40	**29**	30	32+5	29+5	35	29+5
Trial Time (s)	0.66	1.02	**0.07**	110.4	7.15	7.30	7.12

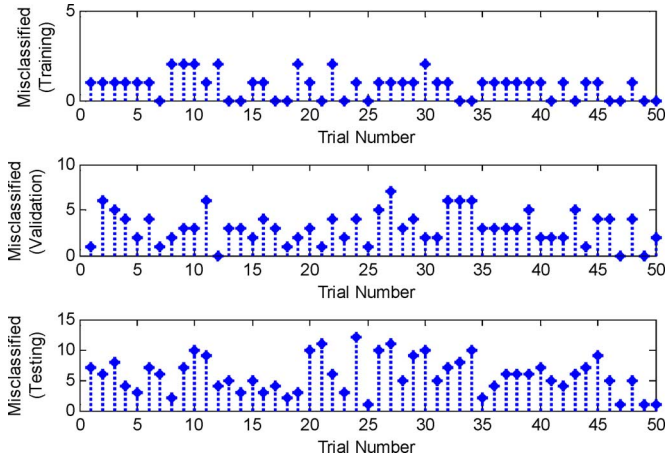

Fig. 12. Results of FCM-QPSO (Hyb.)—power transformer data with noise.

TABLE IX
RESULTS COMPARISON—POWER TRANSFORMER FAULT DATA SET WITH NOISE

#	1	2	3	4	5	6	7
Tra.+Va. Rate (%)	83.08	94.16	90.04	89.20	93.74	92.94	**96.09**
Std. Dev.	**0.011**	0.041	0.028	0.037	0.033	0.040	0.034
Testing Rate (%)	**80.96**	73.60	56.80	72.42	75.44	75.36	76.72
Std. Dev.	**0.030**	0.051	0.103	0.144	0.121	0.120	0.117
Vectors / Neurons	36.90	**25.00**	29	36+5	31+5	36	31+5
Trial Time (s)	0.54	1.02	**0.12**	116.8	7.23	7.64	7.16

three single models are developed to identify the four types of faults, respectively. The structure of these cascade classifiers is represented in Fig. 13.

From Fig. 14, it can be observed that the principal component data congregate around the neighborhood of the coordinate origin, which is the normal principal component projection region. Meanwhile, the other data deviate far from the normal sample and can readily be classified.

Based on the data obtained through 50 trials, the comparisons of the testing performance by different approaches are given in Table X.

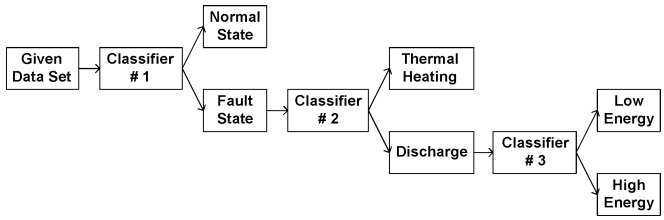

Fig. 13. Structure of cascade classifiers.

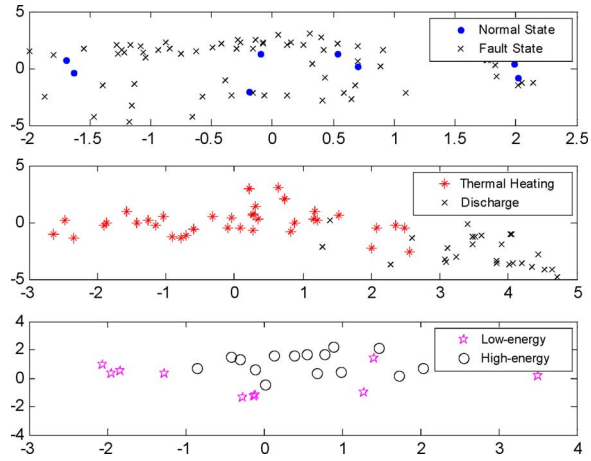

Fig. 14. Single case principal component projection map.

TABLE X
RESULTS COMPARISON—POWER TRANSFORMER FAULT ANALYSIS

#	1	2	3	4	5	6	7
Classifier # 1 (%)	88.56	89.28	89.36	90.00	94.08	93.92	**94.64**
Classifier # 2 (%)	99.24	99.42	99.52	99.33	99.62	99.62	**99.62**
Classifier # 3 (%)	82.50	82.25	82.75	83.50	86.25	86.25	**87.25**
Mean (%)	90.10	90.32	90.54	90.94	93.32	93.26	**93.84**

D. Results Analysis

As shown in the results, we can conclude that the proposed training method for RBF network overcomes the blindness in choosing suitable neural network structure and in determining the centers and radii of radial basis function. Contrary to the most standard RBF neural network learning methods, where the network structure is selected by a time consuming trial and error procedure, the RBF neural network structure can be automatically determined for any given data by this proposed algorithm.

Although the proposed neural network training algorithm needs more time in learning, obviously the RBF neural network optimized by this method ensures the overall generation ability. The application of this hybrid RBF neural network structure not only results in less training time but also better generalization ability. The promising classification ability on the benchmark data sets illustrated the efficiency of the proposed method.

However, each approach has its own merits and drawbacks. Although in the SVM and RVM, the number of support vectors (SV) or relevance vectors (RV), are determined automatically,

there are still some parameters are need to provided in advance. Furthermore, in order to get superior computing performance, these parameters should have problem-oriented values.

The drawback of the proposed scheme is the computational speed, because of long selection procedure for suitable number of hidden layer nodes. There are three options to improve the computational speed. 1) Try to reduce the range of hidden layer neurons; 2) Compute the kernel function width with effective mathematical methods, like the nelder-mead simplex method or maximum likelihood method; 3) Faster EAs can be designed to accelerate the optimization process.

V. Conclusion

This paper has presented a self-adaptive RBF neural network-based DGA methods for power transformer fault diagnosis. The proposed learning approach is able to generate RBF neural network models based on specially designed FCM and QPSO, which can auto-configure the structure of networks and obtain the model parameters. The availability of this method is proved by applying RBF neural network in classification of five famous benchmark data sets, and power transformer history faults dataset. The result suggests that the proposed training algorithms have good performance on data clustering, improving stability and generalization ability of RBF neural networks. The promising neural network performance on the validation data sets illustrates the improved accuracy of the proposed method when compared to the other alternatives and showed that it could be used as a reliable tool for transformer fault analysis.

References

[1] J. J. Kelly, "Transformer fault diagnosis by dissolved-gas analysis," *IEEE Trans. Ind. Appl.*, vol. IA-16, no. 6, pp. 777–782, Nov. 1980.

[2] C. E. Lin, J. M. Ling, and C. L. Huang, "An expert system for transformer fault diagnosis using dissolved gas analysis," *IEEE Trans. Power Del.*, vol. 8, no. 1, pp. 231–238, Jan. 1993.

[3] Q. Su, C. Mi, L. L. Lai, and P. Austin, "A fuzzy dissolved gas analysis method for the diagnosis of multiple incipient faults in a transformer," *IEEE Trans. Power Syst.*, vol. 15, no. 2, pp. 593–598, May 2000.

[4] H. T. Yang, C. C. Liao, and J. H. Chou, "Fuzzy learning vector quantization networks for power transformer condition assessment," *IEEE Trans. Dielectr. Insul.*, vol. 8, no. 1, pp. 143–149, Feb. 2001.

[5] Y. Zhang, X. Ding, Y. Liu, and P. J. Griffin, "An artificial neural network approach to transformer fault diagnosis," *IEEE Trans. Power Del.*, vol. 11, no. 4, pp. 1836–1841, Oct. 1996.

[6] Z. Y. Wang, Y. L. Liu, and P. J. Griffin, "A combined ANN and expert system tool for transformer fault diagnosis," *IEEE Trans. Power Del.*, vol. 13, no. 4, pp. 1224–1229, Oct. 1998.

[7] Y. C. Huang, "Evolving neural nets for fault diagnosis of power transformers," *IEEE Trans. Power Del.*, vol. 18, no. 3, pp. 843–848, Jul. 2003.

[8] Y. C. Huang, "A new data mining approach to dissolved gas analysis of oil-insulated power apparatus," *IEEE Trans. Power Del.*, vol. 18, no. 4, pp. 1257–1261, Oct. 2003.

[9] G. Y. Lv, H. Z. Cheng, H. B. Zhai, and L. X. Dong, "Fault diagnosis of power transformer based on multilayer SVM classifier," *Elect. Power Syst. Res.*, vol. 74, no. 1, pp. 1–7, Apr. 2005.

[10] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, no. 9, pp. 1481–1497, Sep. 1990.

[11] J. Park and I. W. Sandberg, "Universal approximation using radial basis function networks," *Neural Comp.*, vol. 3, no. 2, pp. 246–257, Jun. 1991.

[12] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1974.

[13] J. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput. Geosci.*, vol. 10, no. 2–3, pp. 191–203, 1984.

[14] A. Staiano, R. Tagliaferri, and W. Pedrycz, "Improving RBF networks performance in regression tasks by means of a supervised fuzzy clustering," *Neurocomputing*, vol. 69, no. 13–15, pp. 1570–1581, Aug. 2006.

[15] E. S. Chng, S. Chen, and B. Mulgrew, "Gradient radial basis function networks for nonlinear and nonstationary time series prediction," *IEEE Trans. Neural Netw.*, vol. 7, no. 1, pp. 190–194, Jan. 1996.

[16] S. Chen, Y. Wu, and B. L. Luk, "Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1239–1243, Sep. 1999.

[17] C. M. Huang and F. L. Wang, "An RBF network with OLS and EPSO algorithms for real-time power dispatch," *IEEE Trans. Power Syst.*, vol. 22, no. 1, pp. 96–104, Feb. 2007.

[18] K. P. Wong and Z. Y. Dong, "Differential evolution, an alternative approach to evolutionary algorithm," in *Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems*, K. Y. Lee and M. El-Sharkawi, Eds. New York: Wiley, 2008, invited book chapter.

[19] G. Y. Yang, Z. Y. Dong, and K. P. Wong, "A modified differential evolution algorithm with fitness sharing for power system planning," *IEEE Trans. Power Syst.*, vol. 23, no. 2, pp. 514–522, May 2008.

[20] J. B. Park, K. S. Lee, J. R. Shin, and K. Y. Lee, "A particle swarm optimization for economic dispatch with non-smooth cost functions," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 34–42, Feb. 2005.

[21] K. Meng, H. G. Wang, Z. Y. Dong, and K. P. Wong, "Quantum-inspired particle swarm optimization for valve-point economic load dispatch," *IEEE Trans. Power Syst.*, to be published.

[22] D. Huang and T. W. S. Chow, "A people-counting system using a hybrid RBF neural network," *Neural Process. Lett.*, vol. 18, no. 2, pp. 97–113, Oct. 2003.

[23] K. H. Han and J. H. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," *IEEE Trans. Evol. Comput.*, vol. 6, no. 6, pp. 580–593, Dec. 2002.

[24] K. H. Han and J. H. Kim, "Quantum-inspired evolutionary algorithms with a new termination criterion, Hc Gate, and two-phase scheme," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 156–169, Apr. 2004.

[25] J. G. Vlachogiannis and K. Y. Lee, "Quantum-inspired evolutionary algorithm for real and reactive power dispatch," *IEEE Trans. Power Syst.*, vol. 23, no. 4, pp. 1627–1636, Nov. 2008.

[26] A. Asuncion and D. J. Newman, UCI Machine Learning Repository. Irvine, CA, Univ. California, Sch. Inf. and Comput. Sci., 2007. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[27] V. N. Vapnik, *The nature of statistical learning theory*. New York: Springer-Verlag, 1995.

[28] M. E. Tipping, "Sparse Bayesian learning and relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, no. 3, pp. 211–244, Sep. 2001.

[29] Matlab Neural Network Toolbox. [Online]. Available: http://www.mathworks.com/.

[30] R. A. Fisher, "The Use of multiple measurements in taxonomic problems," *Annu. Eugenics*, vol. 7, pt. II, pp. 179–188, 1936.

[31] H. R. Barbara and H. R. Frederick, "Concept learning and the recognition and classification of exemplars," *J. Verbal Learn. Verbal Behav.*, vol. 16, no. 3, pp. 321–338, Jun. 1977.

[32] D. Coomans, M. Broeckaert, M. Jonckheer, and D. L. Massart, "Comparison of multivariate discriminant techniques for clinical data—Application to the thyroid functional state," *Meth. Inf. Med.*, vol. 22, no. 2, pp. 93–101, Apr. 1983.

[33] S. J. Haberman, "Generalized residuals for log-linear models," in *Proc. 9th Int. Biomet. Conf.*, 1976, pp. 104–122.

[34] T. Denoeux, "A neural network classifier based on Dempster-Shafer theory," *IEEE Trans. Syst., Man, Cybern.*, vol. 30, no. 2, pt. A, pp. 131–150, Mar. 2000.

[35] J. E. Jackson, "Quality control methods for two related variables," *Ind. Qual. Control*, vol. 12, no. 7, pp. 4–8, 1956.

Ke Meng (M'10) received the M.E. degree from the East China University of Science and Technology, Shanghai, China, in 2007. He is pursuing the Ph.D. degree at the School of Information Technology and Electrical Engineering, The University of Queensland, St. Lucia, Australia.

He is now a research fellow at the Department of Electrical Engineering, The Hong Kong Polytechnic University, Hong Kong. His research interest includes intelligent algorithms, power system stability analysis, and control.

ZhaoYang Dong (M'99–SM'06) received the Ph.D. degree from The University of Sydney, Sydney, Australia, in 1999.

He is currently with the Hong Kong Polytechnic University, Hong Kong. Before coming to Hong Kong, he was a System Planning Manager with Transend Networks, Tasmania, Australia, and an Associate Professor with the University of Queensland, St. Lucia, Australia. His research interest includes power system planning, power system stability and control, electricity market, and computational intelligence and its application in power engineering.

Dian Hui Wang (M'02–SM'03) received the Ph.D. degree from Northeastern University, China, in March 1995.

From 1995 to 1997, he worked as a Postdoctoral Fellow at Nanyang Technological University, Singapore, then as a Research Associate and Research Fellow at The Hong Kong Polytechnic University, Hong Kong, until 2001. Since July 2001, he has been with the Department of Computer Science and Computer Engineering at La Trobe University, Melbourne, Australia, and currently as a Reader and Associate Professor. His current work includes intelligent systems for bioinformatics, information retrieval, and engineering applications.

Dr. Wang is serving as an Associate Editor for *Information Sciences*, *Neurocomputing*, and *International Journal of Applied Intelligence*.

Kit Po Wong (M'87–SM'90–F'02) received the M.Sc., Ph.D., and D.Eng. degrees from the University of Manchester, Institute of Science and Technology, Manchester, U.K., in 1972, 1974, and 2001, respectively.

He was with The University of Western Australia, Perth, Australia, from 1974 until 2004 and is now an Adjunct Professor there. Since 2002, he has been Chair Professor, and previously Head, of the Department of Electrical Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong. His current research interests include computation intelligence applications to power system analysis, planning and operations, as well as power market analysis.

Prof. Wong received three Sir John Madsen Medals (1981, 1982, and 1988) from the Institution of Engineers Australia, the 1999 Outstanding Engineer Award from IEEE Power Chapter Western Australia, and the 2000 IEEE Third Millennium Award. He was a Co-Technical Chairman of the IEEE ICMLC 2004 Conference and General Chairman of IEEE/CSEE PowerCon2000. He was an Editor-in-Chief of *IEE Proceedings in Generation, Transmission and Distribution* and Editor (Electrical) of the *Transactions of Hong Kong Institution of Engineers*. He is a Fellow of IET, HKIE, and IEAust.