Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090251

Fault Diagnosis of Power Transformers using Kernel based Extreme Learning Machine with Particle Swarm Optimization

Liwei Zhang[*] *and Jinsha Yuan*

School of Electrical and Electronic Engineering, North China Electric Power University, 071000 Baoding, China

Received: 22 Jun. 2014, Revised: 20 Sep. 2014, Accepted: 23 Sep. 2014
Published online: 1 Mar. 2015

Abstract: To improve the fault diagnosis accuracy for power transformers, this paper presents a kernel based extreme learning machine (KELM) with particle swarm optimization (PSO). The parameters of KELM are optimized by using PSO, and then the optimized KELM is implemented for fault classification of power transformers. To verify its effectiveness, the proposed method was tested on nine benchmark classification data sets compared with KELM optimized by Grid algorithm. Fault diagnosis of power transformers based on KELM with PSO were compared with the other two ELMs, back-propagation neural network (BPNN) and support vector machines (SVM) on dissolved gas analysis (DGA) samples. Experimental results show that the proposed method is more stable, could achieve better generalization performance, and runs at much faster learning speed.

Keywords: power transformers, fault diagnosis, dissolved gas analysis, extreme learning machine, particle swarm optimization.

1 Introduction

Power transformers are considered as highly essential equipment of electric power transmission systems and often the most expensive devices in a substation. Failures of large power transformers can cause operational problems to the transmission system [1].

Dissolved gas analysis (DGA) is one of the most widely used tools to diagnose the condition of oil-immersed transformers in service. The ratios of certain dissolved gases in the insulating oil can be used for qualitative determination of fault types. Several criteria have been established to interpret results from laboratory analysis of dissolved gases, such as IEC 60599 [2] and IEEE Std C57.104-2008 [3]. However, analysis of these gases generated in mineral-oil-filled transformers is often complicated for fault interpretation, which is dependent on equipment variables.

With the development of artificial intelligence, various intelligent methods have been applied to improve the DGA reliability for oil-immersed transformers. Based on DGA technique, an expert system was proposed for transformer fault diagnosis and corresponding maintenance actions, and the test results showed it was effective [4]. By using fuzzy information approach, K. Tomsovic *et al.* [5] developed a framework that combined several transformer diagnostic methods to provide the "best" conclusion. Based on artificial neural network (ANN), Zhang et al. presented a two-step method for fault detection in oil-filled transformer, and the proposed approach achieved good diagnostic accuracy [6]. Wei-Song Lin et al. proposed a novel fault diagnosis method for power transformer based on Cerebellar Model Articulation controller (CMAC), and the new scheme was shown with high accuracy [7]. Yann-Chang Huang et al. presented a fault detection approach of oil-immersed power transformers based on genetic algorithm tuned wavelet networks (GAWNs) demonstrating remarkable diagnosis accuracy [8]. Weigen Chen et al. studied the efficiency of wavelet networks (WNs) for transformer fault detection using gases-in-oil samples, and the diagnostic accuracy and efficiency were proved better than these derived from BPNN [9]. A fault classifier for power transformer was proposed by Zheng *et al.* based on multi-class least square support vector machines (LS-SVM) [10]. Xiong Hao *et al.* developed an artificial immune algorithm for transformer fault detection [11].

[*] Corresponding author e-mail: zlwbd@126.com

© 2015 NSP
Natural Sciences Publishing Cor.

These novel diagnosis methods overcome the drawbacks of IEC method and improve the diagnosis accuracy.

However, conventional learning methods on neural networks such as back-propagation (BP) and SVM methods apparently face some drawbacks: (1) slow learning speed, (2) trivial human tuned parameters, and (3) trivial learning variants for different applications [12]. Extreme learning machine (ELM) is an emerging learning technique proposed for generalized single-hidden layer feed forward networks (SLFNs) [13]. ELM overcomes some major constraints faced by conventional learning methods and computational intelligence techniques. Similar to SVM, kernels can be applied in ELM as well [14,15]. Kernel based ELM (KELM) can be implemented in a single learning step, so it runs fast. Similar to other kernel based methods, the parameters of KELM are usually assigned empirically or obtained by trials [15]. Obviously, it is very time-consuming and the performance achieved with the chosen parameters is suboptimal.

Therefore, KELM with particle swarm optimization (PSO) is proposed for fault diagnosis of power transformers in this paper. This paper is organized as follows. Section 2 reviews original ELM, equality constrained-optimization-based ELM and KELM. Section 3 introduces the parameters selection for KELM. In Section 4, the proposed KELM with PSO are described in detail. Section 5 discusses the comparison results of the proposed method with other approaches. Conclusions are finally drawn in Section 6.

2 Kernel based ELM (KELM)

2.1 Original ELM

Extreme Learning Machine (ELM) was originally developed for the single-hidden layer feedforward networks (SLFNs) and then extended to the "generalized" SLFNs. The hidden layer in ELM need not be tuned. ELM randomly chooses the input weights and the hidden neurons' biases and analytically determines the output weights of SLFNs. Input weights are the weights of the connections between input neurons and hidden neurons and output weights are the weights of the connections between hidden neurons and output neurons.

The output function of ELM for generalized SLFNs (take one output node case as an example) is

$$f_L(\boldsymbol{x}) = \sum_{i=1}^{L} \beta_i h_i(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta} \tag{1}$$

where $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_L]^{\mathrm{T}}$ is the vector of the output weights between the hidden layer of L nodes and the output node, and $\boldsymbol{h}(\boldsymbol{x}) = [h_1(\boldsymbol{x}), \cdots, h_L(\boldsymbol{x})]$ is the output (row) vector of the hidden layer with respect to the input $\boldsymbol{x}$. $\boldsymbol{h}(\boldsymbol{x})$ actually maps the data from the d-dimensional input space to the L-dimensional hidden-layer feature space (ELM feature space) $\mathbf{H}$, and thus, $\boldsymbol{h}(\boldsymbol{x})$ is indeed a feature mapping.

Given a set of training data $\{(\boldsymbol{x}_i, \boldsymbol{t}_i) | \boldsymbol{x}_i \in \mathbf{R}^d, \boldsymbol{t}_i \in \mathbf{R}^m, i = 1, \ldots, N\}$. Different from traditional learning algorithms, ELM tends to reach not only the smallest training error but also the smallest norm of output weights

$$\text{Minimize: } \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|^2 \text{ and } \|\boldsymbol{\beta}\| \tag{2}$$

where $\boldsymbol{H}$ is the hidden-layer output matrix

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}_1) \\ \vdots \\ \boldsymbol{h}(\boldsymbol{x}_N) \end{bmatrix} = \begin{bmatrix} h_1(\boldsymbol{x}_1) & \cdots & h_L(\boldsymbol{x}_1) \\ \vdots & \ddots & \vdots \\ h_1(\boldsymbol{x}_N) & \cdots & h_L(\boldsymbol{x}_N) \end{bmatrix}, \tag{3}$$

and $\boldsymbol{T}$ is the expected output matrix

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{t}_N^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} t_{11} & \cdots & t_{1m} \\ \vdots & \ddots & \vdots \\ t_{N1} & \cdots & t_{Nm} \end{bmatrix}. \tag{4}$$

The minimal norm least square method was used in the original implementation of ELM

$$\boldsymbol{\beta} = \boldsymbol{H}^{\dagger}\boldsymbol{T} \tag{5}$$

where $\boldsymbol{H}^{\dagger}$ is the *Moore–Penrose generalized inverse* of matrix $\boldsymbol{H}$.

The orthogonal projection method can be used to calculate the Moore–Penrose generalized inverse of $\boldsymbol{H}$ in two cases: when $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}$ is nonsingular and $\boldsymbol{H}^{\dagger} = (\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathrm{T}}$, or when $\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}$ is nonsingular and $\boldsymbol{H}^{\dagger} = \boldsymbol{H}^{\mathrm{T}}(\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}})^{-1}$.

2.2 Equality constrained-optimization-based ELM

According to the ridge regression theory, one can add a positive value to the diagonal of $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}$ or $\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}$; the resultant solution is more stable and tends to have better generalization performance.

For multiclass classifier with multi-outputs, classifiers with m-class have m output nodes. If the original class label is p, the expected output vector of the m output nodes is $\boldsymbol{t}_i = [0, \cdots, 0, \overset{p}{1}, 0, \cdots 0]^{\mathrm{T}}$. In this case, only the pth element of $\boldsymbol{t}_i = [t_{i1}, \cdots, t_{im}]^{\mathrm{T}}$ is one, while the rest of the elements are set to zero. For the constrained-optimization-based ELM with multi-output node, the classification problem can be formulated as

$$\begin{aligned} &\text{Minimize: } L_{P_{ELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{C}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|^2 \\ &\text{Subject to: } \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta} = \boldsymbol{t}_i^{\mathrm{T}} - \boldsymbol{\xi}_i^{\mathrm{T}} \quad i = 1, \cdots, N \end{aligned} \tag{6}$$

© 2015 NSP
Natural Sciences Publishing Cor.

where $\boldsymbol{\xi}_i = [\xi_{i1}, \cdots, \xi_{im}]^{\mathrm{T}}$ is the training error vector of the m output nodes with respect to the training sample $\boldsymbol{x}_i$, C is the cost parameter.

Based on the Karush–Kuhn–Tucker (KKT) theorem, to train ELM is equivalent to solving the following dual optimization problem:

$$L_{D_{ELM}} = \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{C}{2}\sum_{i=1}^{N}\|\boldsymbol{\xi}_i\|^2 - \sum_{i=1}^{N}\sum_{j=1}^{m}\alpha_{i,j}(\boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta}_j - t_{i,j} + \xi_{i,j}) \quad (7)$$

We can have the KKT corresponding optimality conditions as follows:

$$\frac{\partial L_{D_{ELM}}}{\partial \boldsymbol{\beta}_j} = 0 \rightarrow \boldsymbol{\beta}_j = \sum_{i=1}^{N}\alpha_{i,j}\boldsymbol{h}(\boldsymbol{x}_i)^{\mathrm{T}} \rightarrow \boldsymbol{\beta} = \boldsymbol{H}^{\mathrm{T}}\boldsymbol{\alpha} \quad (8)$$

$$\frac{\partial L_{D_{ELM}}}{\partial \boldsymbol{\xi}_i} = 0 \rightarrow \boldsymbol{\alpha}_i = C\boldsymbol{\xi}_i, \quad i = 1, \cdots, N \quad (9)$$

$$\frac{\partial L_{D_{ELM}}}{\partial \boldsymbol{\alpha}_i} = 0 \rightarrow \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta} - \boldsymbol{t}_i^{\mathrm{T}} + \boldsymbol{\xi}_i^{\mathrm{T}} = 0, \quad i = 1, \cdots, N \quad (10)$$

By substituting (8) and (9) into (10), the aforementioned equations can be equivalently written as

$$\left(\frac{\boldsymbol{I}}{\boldsymbol{C}} + \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\right)\boldsymbol{\alpha} = \boldsymbol{T} \quad (11)$$

From (8) and (11), we have

$$\boldsymbol{\beta} = \boldsymbol{H}^{\mathrm{T}}\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\right)^{-1}\boldsymbol{T} \quad (12)$$

The output function of ELM classifier is

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{\beta} = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{H}^{\mathrm{T}}\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\right)^{-1}\boldsymbol{T} \quad (13)$$

2.3 Kernel based ELM (KELM)

If a feature mapping $h(\boldsymbol{x})$ is unknown to users, one can apply Mercer's conditions on ELM. We can define a kernel matrix for ELM as follows:

$$\boldsymbol{\Omega}_{ELM} = \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}} : \Omega_{ELMi,j} = h(\boldsymbol{x}_i)h(\boldsymbol{x}_j) = K(\boldsymbol{x}_i, \boldsymbol{x}_j) \quad (14)$$

Then, the output function of ELM classifier (13) can be written compactly as

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{H}^{\mathrm{T}}\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\right)\boldsymbol{T} = \begin{bmatrix} K(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K(\boldsymbol{x}, \boldsymbol{x}_N) \end{bmatrix}\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{\Omega}_{ELM}\right)^{-1}\boldsymbol{T} \quad (15)$$

After ELM was trained, the given testing sample $\boldsymbol{x}$ was taken as the input of the classifier. The index of the output node with the highest output value is considered as the predicted class label of the given testing sample. Let $f_i(\boldsymbol{x})$ denote the output function of the ith output node, the predicted class label of sample $\boldsymbol{x}$ is

$$label(\boldsymbol{x}) = \arg\max_{i \in \{1, \cdots, m\}} f_i(\boldsymbol{x}) \quad (16)$$

3 User-specified parameters

In this study, the popular Gaussian kernel function $K(\boldsymbol{u}, \boldsymbol{v}) = \exp(-\gamma\|\boldsymbol{u} - \boldsymbol{v}\|^2)$ is used as the kernel function in KELM. In order to achieve good generalization performance, the cost parameter C and kernel parameter γ of KELM need to be chosen appropriately. Similar to SVM and LS-SVM, the values of C and γ are assigned empirically or obtained by trying a wide range of C and γ. As suggested in [15], 50 different values of C and 50 different values of γ are used for each data set, resulting in a total of 2500 pairs of (C, γ). The 50 different values of C and γ are $\{2^{-24}, 2^{-23}, \ldots, 2^{24}, 2^{25}\}$. This parameters optimization method is called Grid algorithm.

Similar to SVM and LS-SVM, the generalization performance of KELM with Gaussian kernel are sensitive to the combination of (C, γ) as well. The best generalization performance of ELM with Gaussian kernel is usually achieved in a very narrow range of such combinations. Thus, the best combination of (C, γ) of KELM with Gaussian kernel needs to be chosen for each data set.

Take Diabetes data set for example, the performance sensitivity of KELM with Gaussian kernel on the user-specified parameters (C, γ) is shown in Fig. 1. The simulations were carried out in MATLAB 7.0.1 environment running in Core 2 Duo 1.8GHZ CPU with 2GB RAM. Diabetes data set is from the University of California, Irvine (UCI) machine learning repository, concluding 2 classes and 768 instances. In this case, 576 instances were selected randomly from Diabetes data set as training data and the rest 192 instances were taken as testing data. As mentioned above, 50 different values of C and 50 different values of γ were used in this simulation. It can be seen from Fig. 1 that the performance of KELM with Gaussian kernel on Diabetes data set is sensitive to the user-specified parameters (C, γ) and the highest testing accuracy is obtained in a very narrow range of the combination of (C, γ). The time used for searching the best combination of these two parameters is 732.04 seconds; one of the best combinations of (C, γ) is $(2^0, 2^1)$ and the corresponding testing accuracy is 83.85%.

4 Optimal KELM with PSO

From practical point of view, it may be time consuming and tedious for users to choose appropriate kernel

© 2015 NSP
Natural Sciences Publishing Cor.

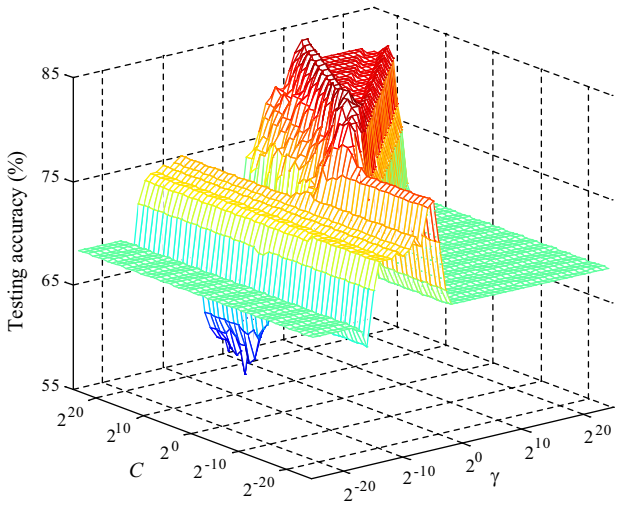

Fig. 1: Performances of KELM with Gaussian kernel on Diabetes data set.

parameters (C, γ) by using the method mentioned in Section 3. What is more, the discrete values of C and γ might result in suboptimal testing accuracy although a wide range of C and γ have been tried (which will be discussed later in Section 5.1). In order to reduce time costs and achieve optimal generalization performance, the parameters in KELM with Gaussian kernel were optimized by using particle swarm optimization (PSO) in this paper.

4.1 Particle swarm optimization

Particle swarm optimization (PSO) is a population-based stochastic optimization technique developed by Eberhart and Kennedy [16]. PSO simulates the social behavior of organisms, such as birds in a flock or fishes in a school, and can be described as an automatically evolving system.

PSO works by initializing a flock of birds randomly over the searching space, where every bird is called as a "particle". These particles fly with a certain velocity and find the global best position after several iterations. During each iteration, each particle adjusts its velocity vector according to its momentum and the influence of its best position (P_b) as well as the best position of its neighbors (P_g), and then a new position that the particle is to fly is obtained. Supposing the dimension of searching space is D, the total number of particles is n, the position of the ith particle can be expressed as vector $\boldsymbol{X}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$; the best position of the ith particle searching until now is denoted as $\boldsymbol{P}_{ib} = (p_{i1}, p_{i2}, \ldots, p_{iD})$, and the best position of all particles searching until now is denoted as vector $\boldsymbol{P}_g = (p_{g1}, p_{g2}, \ldots, p_{gD})$; the velocity of the ith particle is represented as vector $\boldsymbol{V}_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. Then the original PSO is described as

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1 [p_{id}(t) - x_{id}(t)] + c_2 r_2 [p_{gd}(t) - x_{id}(t)] \quad (17)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad i = 1, 2, \ldots, n, \quad d = 1, 2, \ldots, D \quad (18)$$

Where c_1, c_2 are the acceleration constants with positive values; r_1 and r_2 are random number between 0 and 1. In addition to the c_1 and c_2 parameters, the implementation of the original algorithm also requires to place a limit on the velocity ($v_{\max}$). The inertia weight ω is used to balance the capabilities of global exploration and local exploration, which has a high value at the beginning and gradually lower later on. The following equation is used to determine ω

$$\omega = \omega_{\min} + (\omega_{\max} - \omega_{\min})(t-1)(T_e - 1) \quad (19)$$

Where $\omega_{\max}$ is the initial inertia weight, $\omega_{\min}$ is the final inertia weight, t is the current iteration and T_e is the epoch parameter when inertial weight at final value.

4.2 Parameters selection of KELM using PSO

In PSO for parameters optimization, the dimension of searching space is $D = 2$ corresponding to the two parameters (C, γ) of KELM with Gaussian kernel, and the position of each particle represents the parameter values of (C, γ) in Gaussian kernel. The aim of PSO for parameters optimization is to obtain the best generalization performance of KELM; therefore the testing accuracy can be taken as the fitness function of PSO.

The specific steps of PSO for KELM parameters optimization are described as follows.

Step1: Data preprocessing. All the attributes (except expected targets) of the classification dataset are normalized into the range $[-1, 1]$ and then the classification dataset is randomly divided into training and testing data in proportion.

Step2: Initialize the swarm size, maximum of iterations and velocities. Generate randomly an initial velocity for each particle.

Step3: Evaluate each particle's fitness value according to the testing accuracy of KELM and set the best position from the particle with the maximal fitness in the swarm.

Step4: Update the velocity and position for each candidate particle by means of (17), (18) and (19) in each iteration.

Step5: Check the termination criterion. If the maximum number of iterations is not yet reached, return to Step 3. Otherwise go to the next step.

Step6: Output the best combination of (C, γ) of KELM corresponding to the maximal fitness value.

The flowchart of this procedure is illustrated in Fig. 2.

5 Experiment results and discussion

In this section, all simulations on each data sets are carried out in MATLAB 7.6 environment running in Core

© 2015 NSP
Natural Sciences Publishing Cor.

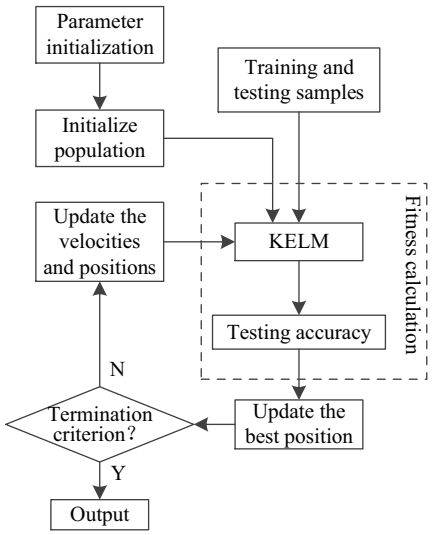

Fig. 2: Computational procedure of PSO for optimizing KELM.

2 Duo 1.8GHZ CPU with 2GB RAM. For KELM with PSO in all experiments, the range of cost parameter C and kernel parameter γ were also $[2^{-24}, 2^{25}]$ as mentioned in Section 3; population size was set to 24; maximum number of iterations (epochs) to train was set to 1000; acceleration constants c_1 and c_2 were set to 2; max particle velocity $v_{\max}$ was set to 2^{10}; initial inertia weight $\omega_{\max}$ was set to 0.9 and final inertia weight $\omega_{\min}$ was set to 0.4; epoch parameter T_e when inertial weight is at final value was set to 750. The training and testing data of all datasets are fixed for all trials of simulations.

5.1 Performance comparison on benchmark classification data sets

For comparison with Grid algorithm, KELM with PSO was also tested on Diabetes data set. The training and testing data were the same as Grid algorithm mentioned in Section 3. The fitness curve of PSO for KELM parameters optimization is shown in Fig. 3.

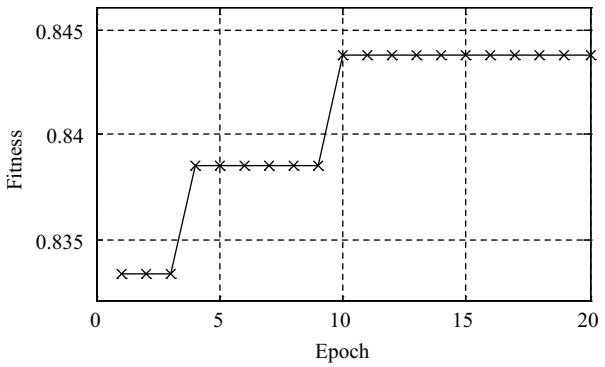

Fig. 3: Fitness curve of PSO for KELM parameters optimization.

It can be found from Fig. 3 that the best fitness is obtained after 10 iterations. The best testing accuracy was 84.38% and the corresponding parameters (C, γ) were $(2^{1.913}, 2^{0.8986})$. The best testing accuracy of KELM with Gaussian kernel optimized by PSO is higher than that derived by Grid algorithm mentioned in Section 3. The time consumed by PSO for KELM parameters optimization is 270.31 seconds, far less than 732.04 seconds spent by Grid algorithm. Compared with Grid algorithm, KELM with PSO achieves better generalization performance.

To verify the performance of KELM with PSO, simulations are also conducted on other eight benchmark classification datasets from the University of California, Irvine (UCI) machine learning repository. Specifications of the nine classification data sets (including Diabetes data set) are shown in Table 1.

Table 1: Specifications of classification datasets.

Datasets	#classes	#attributes	#instances	#train	#test
Diabetes	2	8	768	576	192
Balance	3	4	625	376	249
Glass	7	9	214	139	75
Iris	3	4	150	102	48
Wine	3	13	178	121	57
Liver	2	6	345	231	114
Vowel	11	10	990	528	462
Waveform	3	21	5000	3002	1998
Breast-can	2	30	569	301	268

The corresponding performance of KELM optimized by Grid algorithm and PSO on the nine classification problems is listed in Table 2, including the time consumed, parameters C and γ, and the best testing accuracy.

It can be found from Table 2 that the computational time of PSO is far less than that of Grid algorithm and the best testing accuracies obtained by PSO is even higher than that derived by Grid algorithm.

Finally, KELM with PSO achieves better performance and is less time-consuming compared with Grid algorithm.

5.2 Fault diagnosis of power transformers

In this study, 387 dissolved gas analysis (DGA) samples from real-world fault transformers are chose as experimental data. These samples were divided into two parts: 235 samples were taken as training data randomly and the rest 152 samples as testing data.

There are five gas concentrations in each instance corresponding to five dissolved gases: Hydrogen (H_2), ethylene (C_2H_4), methane (CH_4), ethane (C_2H_6), and acetylene (C_2H_2), which are the byproducts caused by internal faults in power transformers. The attributes of each instance are normalized as {H2/T, CH4/T, C2H6/T, C2H4/T, C2H2/T}, where T represents the total gas. The

© 2015 NSP
Natural Sciences Publishing Cor.

Table 2: Performance comparison of KELM optimized by different algorithms.

Datasets	Grid algorithm				PSO			
	time (s)	C	γ	Testing accuracy (%)	time (s)	C	γ	Testing accuracy (%)
Diabetes	732.05	2^{0}	2^{1}	83.85	**270.31**	$2^{1.913}$	$2^{0.8986}$	**84.38**
Balance-scale	297.19	2^{21}	2^{14}	91.97	**118.02**	$2^{20.63}$	$2^{13.02}$	91.97
Glass	105.53	2^{13}	2^{-3}	73.33	**62.469**	$2^{16.57}$	$2^{-1.556}$	**74.67**
Iris	65.75	2^{-24}	2^{-20}	100	**21.297**	$2^{17.72}$	$2^{-11.78}$	100
Wine	103.5	2^{13}	2^{-5}	100	**46.641**	$2^{12.03}$	$2^{-5.635}$	100
Liver	132.81	2^{17}	2^{9}	75.44	**45.828**	$2^{13.62}$	$2^{6.672}$	75.44
Vowel	766.94	2^{4}	2^{0}	96.97	**525.98**	$2^{18.8}$	$2^{-0.3934}$	**97.19**
Waveform	31123	2^{-1}	2^{9}	86.49	**7454.3**	$2^{-2.764}$	$2^{9.305}$	**86.54**
Breast-can	541.52	2^{10}	2^{16}	97.01	**209.72**	$2^{19.77}$	2^{25}	97.01

Table 3: Performance comparisons of different ELMs on DGA data set.

Algorithms	parameters	values	CPU time (s)		Accuracy (%)	
			Training	Testing	Training	Testing $\pm$ Dev.
original ELM	L	95	0.0625	0.0313	91.91	74.51$\pm$ 1.74
KELM	(C,γ)	$(2^{6.4301}, 2^{-4.3483})$	**0.0097**	**0.0049**	**94.04**	**81.58 $\pm$ 0**
ELM with Sigmoid additive node	(C,L)	$(2^{24}, 700)$	0.5156	0.0313	92.34	76.04$\pm$ 0.89

six types of detectable faults in IEC Publication 60599 by using DGA are discharges of low energy (D1), discharges of high energy (D2), partial discharges (PD), thermal faults below 300°C (T1), thermal faults above 300°C (T2), and thermal faults above 700°C (T3).

The fitness curve of PSO for KELM parameters optimization is shown in Fig. 4.

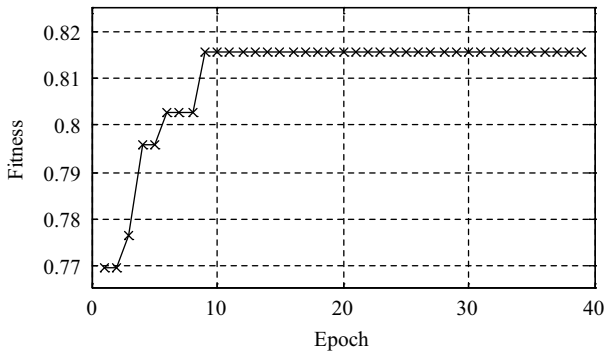

Fig. 4: Fitness curve of KELM with PSO on DGA data set.

It can be found from Fig. 4 that the best fitness is obtained after 9 iterations. The best testing accuracy was 81.58% and the corresponding parameters (C,γ) were $(2^{6.4301}, 2^{-4.3483})$. The computational time was 88.078 seconds. In comparison, the best testing accuracy obtained by Grid algorithm was 80.92%, $(C,\gamma) = (2^{5}, 2^{-5})$, and the computational time was 182.33 seconds.

The performance of KELM with PSO on DGA data set is compared with original ELM and ELM with Sigmoid additive node. The parameters (C,γ) of KELM were set to $(2^{6.4301}, 2^{-4.3483})$ determined by using PSO. In original ELM, the number of hidden nodes L was chosen as 95 through trials with L ranging from 10 to 1000; the hidden nodes used the sigmoid type of activation function. In ELM with Sigmoid additive node, the cost parameter C and the number L of hidden nodes were set as $(C,L) = (2^{24}, 700)$, optimized by using Grid algorithm with C ranging $\{2^{-24}, 2^{-23}, \ldots, 2^{24}, 2^{25}\}$ and L ranging from 10 to 1000. Testing accuracies of the three different ELMs on DGA data set with 50 trials are shown in Fig. 5. The performance comparisons of the three methods are listed in Table 3.

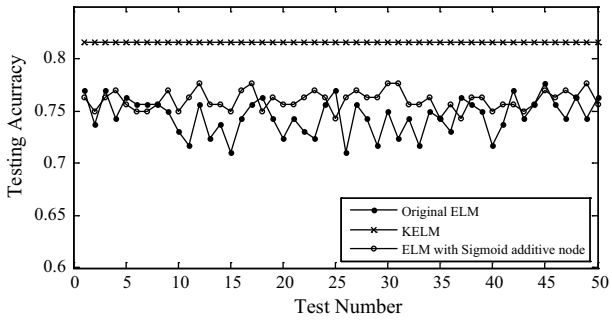

Fig. 5: Testing accuracies of different ELMs on DGA data set.

From Fig. 5, it can be seen that the testing accuracies of original ELM and ELM with Sigmoid additive node are changing in each trial, while the testing accuracy of KELM are constant in all trial and higher than the other two ELMs.

From Table 3, it can be found that KELM requires less training and testing time than the other two ELMs while with the highest training and testing accuracies.

© 2015 NSP
Natural Sciences Publishing Cor.

Table 4: Performance comparisons of different methods on DGA data set.

Algorithms	parameters	values	CPU time (s)		Accuracy (%)	
			Training	Testing	Training ± Dev.	Testing ± Dev.
BPNN	L	28	3.0219	0.0281	78.13±2.34	69.84±2.66
KELM	(C,γ)	$(2^{6.4301}, 2^{-4.3483})$	**0.0097**	**0.0049**	**94.04**±0	**81.58±0**
SVM	(C,γ)	$(2^3, 2^6)$	0.0286	0.0057	91.91±0	79.61±0

In summary, from the above experimental results, it can be concluded that the KELM with PSO achieves better and more stable generalization performance in fault classification for power transformers.

5.3 Comparison with other fault diagnosis approaches for power transformers

Moreover, the performance of the KELM is compared with other widely used diagnosis methods for power transformers, such as back-propagation neural network (BPNN) and support vector machines (SVM).

The training and testing samples from DGA data set were the same as Section 5.2 mentioned above. The BPNN used was of single-hidden-layer, provided in the neural networks tools box of MATLAB; the transfer function was tangent sigmoid; the number of hidden layer nodes was chosen by trials. For SVM, the parameters (C,γ) were selected by using Grid algorithm. The performance comparisons are listed in Table 4.

From Table 4, it can be seen that the training time of KELM on DGA data set is far less than BPNN and SVM, and the training and testing accuracies are the highest. Obviously, the fault diagnosis approach based on KELM is stable and achieves better generalization performance than that based on BPNN and SVM.

6 Conclusions

In this paper, KELM with PSO has been presented for fault diagnosis of power transformers. The parameters of KELM are optimized by using PSO to improve the performance of KELM. Experimental results show that:

(1) Compared with Grid algorithm on nine benchmark classification data sets, KELM optimized by PSO achieves better performance and is less time-consuming.
(2) Compared with original ELM and ELM with Sigmoid additive node, KELM with PSO achieves better and more stable generalization performance in fault classification for power transformers.
(3) Compared with BPNN and SVM on DGA data set, KELM with PSO is able to obtain better diagnosis accuracy and runs faster.

Acknowledgement

The project was supported by the Fundamental Research Funds for the Central Universities (No.13MS69), North China Electric Power University, China.

References

[1] W. H. Tang, and Q. H. Wu, Springer-Verlag, London, 2011.
[2] IEC Publication 60599, 2007.
[3] IEEE Std C57.104-2008, 2009.
[4] C. F Lin, J. M. Ling, and C. L. Huang, IEEE Transactions on Power Delivery **8**, 231-238, 1993.
[5] K. Tomsovic, and M. Tapper, T. Ingvarsson, IEEE Transactions on Power Systems **8**, 1638-1646 (1993).
[6] Y. Zhang, X. Ding, and Y. Liu, IEEE Transactions on Power Delivery **11**, 1836-1841 (1996).
[7] Wei-Song Lin, Chin-Pao Hung, and Mang-Hui Wang, Proceedings of International Symposium on Neural Networks, **1**, 986-991 (2002).
[8] Yann-Chang Huang, IEEE Transactions on Power Delivery **18**, 1257-1261 (2003).
[9] Weigen Chen, Chong Pan, Yuxin Yun, and Yilu Liu, IEEE Transactions on Power Delivery **24**, 187-194 (2009).
[10] H. B. Zheng, R. J. Liao, S. Grzybowski, and L. J. Yang, Electric Power Applications **5**, 691-696 (2011).
[11] X. Hao, and S. Cai-Xin, IEEE Transactions on Power Delivery **22**, 930-935 (2007).
[12] G. -B. Huang, Q. -Y. Zhu, and C. -K. Siew, Neurocomputing **70**, 489-501 (2006).
[13] G. -B. Huang, Q. -Y. Zhu, and C. -K. Siew, Proceedings of the International Joint Conference on Neural Networks (IJCNN2004) **2**, 985-990 (2004).
[14] G. -B. Huang, and C. -K. Siew, International Journal of Information Technology, **11**, 16-24 (2005).
[15] G. -B. Huang, H. Zhou, X. Ding, and R. Zhang, IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics **42**, 513-529 (2012).
[16] Clerc, M., Kennedy, J., IEEE Trans. Evolut. Comput. **6**, 58-73 (2002).

© 2015 NSP
Natural Sciences Publishing Cor.

Liwei Zhang is currently a PhD Student in School of Electrical and Electronic Engineering of North China Electric Power University. His research interests include intelligent information processing and fault diagnosis of electrical equipment.

Jinsha Yuan received his PhD degree from North China Electric Power University in 1992. He is currently a professor of School of Electrical and Electronic Engineering of North China Electric Power University. His research interests include intelligent information processing and electromagnetic field theory and its application.

© 2015 NSP
Natural Sciences Publishing Cor.