Applied Mathematics and Computation 218 (2012) 4973–4987

Contents lists available at SciVerse ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Multi-class support vector machine optimized by inter-cluster distance and self-adaptive deferential evolution

Xiaoyuan Zhang, Jianzhong Zhou *, Changqin Wang, Chaoshun Li, Lixiang Song

College of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan 430074, PR China

ARTICLE INFO

Keywords:
Support vector machine
Parameter optimization
Inter-cluster distance
Differential evolution
Fault diagnosis
Rolling element bearings

ABSTRACT

Support vector machine (SVM) is a popular tool for machine learning task. It has been successfully applied in many fields, but the parameter optimization for SVM is an ongoing research issue. In this paper, to tune the parameters of SVM, one form of inter-cluster distance in the feature space is calculated for all the SVM classifiers of multi-class problems. Inter-cluster distance in the feature space shows the degree the classes are separated. A larger inter-cluster distance value implies a pair of more separated classes. For each classifier, the optimal kernel parameter which results in the largest inter-cluster distance is found. Then, a new continuous search interval of kernel parameter which covers the optimal kernel parameter of each class pair is determined. Self-adaptive differential evolution algorithm is used to search the optimal parameter combination in the continuous intervals of kernel parameter and penalty parameter. At last, the proposed method is applied to several real word datasets as well as fault diagnosis for rolling element bearings. The results show that it is both effective and computationally efficient for parameter optimization of multi-class SVM.

1. Introduction

Support vector machine (SVM) is based on the structural risk minimization (SRM) principle [1], which makes it less prone to over-fitting. By maximizing the margin between two opposite classes, SVM can find the optimal separating hyper-plane that minimizes the upper bound of the generalization error, which enables SVM to have strong capability of fitting and generalization. By introducing the kernel tricks, SVM has the ability of dealing with infinite or nonlinear features in a high dimensional feature space. With the above attractive features, SVM is regarded as state-of-the-art classifier. It is generally acknowledged that SVM has a good performance in solving nonlinear and high dimensional pattern recognition problems with good generalization ability. Although SVM has so many advantages and has been successfully applied in many fields, such as biomedicine [2,3], text categorization [4,5] fault diagnosis [6,7] and so on, in practice, its parameters, the kernel parameters (for instance, width parameter g of RBF kernel function) and penalty parameter C, must be selected judiciously so that the performance of SVM can be brought into full play. Changing the kernel parameters is equivalent to selecting the feature spaces, and tuning C is corresponding to weighting the slack variables, the error terms. Consequently, the performance of SVM depends on its parameters largely. However, there is no systematic methodology or priori knowledge for determining the parameters of SVM.

A wide range of studies have been carried out on this topic. A simple and straightforward way is grid search (GS) [8]. This procedure requires a grid search over the parameter space. It trains SVMs with all desired combinations of parameters and

* Corresponding author.
E-mail address: jz.zhou@mail.hust.edu.cn (J. Zhou).

0096-3003/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2011.10.063

screens them according to the training accuracy. It makes the training process time-consuming, and when the number of parameters exceeds two it will become intractable. Another approach is to build estimates or bounds for the true generation error, and to use numerical optimization methods [9,10] to minimize some analytical criterions which are proxies of these estimates or bounds. The numerical methods are generally more efficient than GS, for their fast convergence rate. However, they are sensitive to the initial point. If the initial point is not proper, the numerical methods might not properly solve the parameter optimization problem. Additionally, these methods require that the kernel functions and the bounds of generalization error have to be differentiable with respect to kernel and penalty constant parameters. Recently, some evolutionary algorithms (EAs), such as genetic algorithm (GA) [11,12], particle swarm optimization algorithm (PSO) [13,14], artificial immunization algorithm (AIA) [6,15] and ant colony optimization algorithm (ACO) [16] have been adopted to optimize the SVM parameters for their better global search abilities. These EAs provide an alternative for finding global optimal solutions in non-convex, highly nonlinear, noisy, time-dependent or flat solution spaces. Consequently, they can find the optimal parameters combination for SVM with a high probability. However, these methods select the best parameter combination from the population evolved generation by generation, which requires training many SVMs. Thus, they are still time-consuming, especially when the search ranges of the parameters are large.

SVM uses kernel functions to map the input data into a high dimensional feature space. Hence, the feature space is determined by the kernel function and its parameters. When the kernel function is selected, the feature space is only determined by the kernel parameters. Selecting kernel parameters is equivalent to selecting the feature space. Obviously, we need a feature space in which the classes are more separated. The inter-cluster distances in the feature spaces (ICDF) indicate the separation of two classes in the feature space. The larger the ICDF, the easier to classify the two classes in feature space. Wu and Wang [17,18] used ICDF to choose the kernel parameters. For binary classification problem, the optimal kernel parameters are found according to the largest ICDF. Then, the selected kernel parameters with different candidate penalty parameter C are used to train SVM models. The parameters combination which results in a highest cross-validation accuracy is selected as the best one. Since calculating ICDFs does not require the information of the trained SVMs, the time needed for the training process for different kernel parameters is saved. This method can get the parameter combinations which result in SVM models perform as good as the models chosen by traditional GS method in testing accuracy, while the training time is shortened largely. However, in this method, the candidate penalty parameters and kernel parameters for calculating ICDFs are given as discretization. It needs to locate the interval of feasible solution and a suitable sampling step, which is a tricky task since a suitable sampling step varies from kernel to kernel and the grid interval may not be easy to locate without prior knowledge of the problem [19]. Furthermore, in practice, most of the classification problems are multi-class, how to extend the binary SVM with parameter optimization by ICDF to multi-class problems is a challenging task.

In this paper, a hybrid method of ICDF index and self-adaptive differential evolution (ADE) algorithm (ICDF–ADE) is proposed to optimize the parameters of SVM. The proposed method capitalizes on the strengths of both the ICDF heuristic and ADE strategy. Firstly, for each class pair of a multi-class problem, the ICDFs are calculated, and the optimal kernel parameter for the SVM of this class pair is found according to the largest ICDF. Then, a small and effective search interval of kernel parameter covering the optimal kernel parameter of each class pair is determined. At last, ADE is used to search the optimal parameter combination of SVM in the continuous intervals of kernel parameter and penalty parameter. Since the search range of kernel parameter is much smaller than that of traditional methods, the training time of the proposed method is much shortened. Moreover, differential evolution (DE) has shown performance superior to that of PSO and other EAs in the widely used benchmark problems [20] and has fewer parameters to set. Thus, it outperforms other methods in optimizing the parameters of SVM. The proposed method is tested on several real word datasets as well as fault diagnosis for rolling element bearings. The experimental results show that it has good performance in optimizing the parameters of multi-class SVM, and thus is suitable for fault diagnosis of rolling element bearings.

The remaining of this paper is organized as follows. The brief introduction of SVM is presented in Section 2. Several forms of ICDF are given in Section 3. The basic ideas of ADE are introduced in Section 4. The proposed method as well as numerical experiments is described in detail in Section 5. In Section 6, the proposed method is applied in fault diagnosis for rolling element bearings and the experimental results are compared. Finally, a general conclusion is drawn in Section 7.

2. Support vector machines

2.1. Brief introduction of support vector machines

Support vector machines (SVM) were first suggested by Vapnik [1]. The principles of SVM stem from statistical learning theory. By using the information of limited samples, SVMs search for a compromise between the model complexity and learning ability to obtain good generalization ability. In this section, some basic conceptions for SVMs are introduced. For a more detailed discussion of SVM can be found in Cristianini and Shawe-Taylor [21].

Given a dataset $(\vec{x_i}, y_i), i = 1, \ldots, l, \vec{x_i} \in R^d, y_i \in \{1, -1\}$, where R^d is the d-dimensional input space, l is the number of training samples, $\vec{x_i}$ is the ith training sample and y_i is its corresponding bipolar label. A linear decision surface can be defined by the equation $f(\vec{x}) = \langle \vec{w}, \vec{x} \rangle + b = 0$, where $\vec{w}$ is a weight vector orthogonal to the decision surface, b is an offset term, $\langle \cdot, \cdot \rangle$ indicates the inner product operation. The original formulation of SVM algorithm seeks a linear decision surface that separates

the two opposite classes with a maximal margin $1/\|\vec{w}\|$. Maximizing the margin $1/\|\vec{w}\|$ is equivalent to minimizing $\|\vec{w}\|^2$, whose solution is found after resolving the following quadratic optimization problem:

$$\begin{aligned} &\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2, \\ &\text{Subject to} \quad y_i\left(\langle \vec{w}, \vec{x_i}\rangle + b\right) \geqslant 1, \quad i = 1,\ldots,l. \end{aligned} \tag{1}$$

This optimization problem can be transformed into its corresponding dual problem:

$$W(a) = \sum_{i}^{l} a_i - \frac{1}{2}\sum_{i,j=1}^{l} a_i a_j y_i y_j \left\langle \vec{x_i}, \vec{x_j} \right\rangle, \tag{2}$$

with constraints: $a_i, a_j \geqslant 0, i = 1,\ldots,l, \sum_{i=1}^{l} y_i a_i = 0$. Where a_i are Lagrange multipliers, they can be found by optimizing Eq. (2).

Considering $\vec{w} = \sum_{i=1}^{l} a_i y_i \vec{x_i}$, the decision function can be got as follows:

$$f(\vec{x}) = \operatorname{sgn}\left(\sum_{\vec{x_i} \in SVs} a_i y_i \left\langle \vec{x_i}, \vec{x} \right\rangle + b\right), \tag{3}$$

where sgn is a signum function. SVs correspond to the set of support vectors, training examples for which the associated Lagrange multipliers are larger than zero.

In order to relax the margin constraints for the non-linearly separable data, the slack variables are introduced into the optimization problem. The following two forms of soft margin SVMs are generally discussed and applied:

$$\begin{aligned} &\min_{\xi,\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{l} \xi_i, \\ &\text{Subject to} \quad y_i\left(\left\langle \vec{w}, \vec{x_i}\right\rangle + b\right) \geqslant 1 - \xi_i, \quad i = 1,\ldots,l, \ \xi_i \geqslant 0, \end{aligned} \tag{4}$$

$$\begin{aligned} &\min_{\xi,\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2 + \frac{C}{2}\sum_{i=1}^{l} \|\xi_i\|^2, \\ &\text{Subject to} \quad y_i\left(\left\langle \vec{w}, \vec{x_i}\right\rangle + b\right) \geqslant 1 - \xi_i, \quad i = 1,\ldots,l, \ \xi_i \geqslant 0, \end{aligned} \tag{5}$$

where ξ_i, $i = 1,\ldots,l$ are slack variables and C is the penalty parameter of error. Eqs. (4) and (5) are called as L1-SVM and L2-SVM, respectively.

In practice, most of the problems are linearly inseparable, even though soft margin SVM is adopted. Thus, the input data is mapped into a high dimensional feature space, in which the data are sparse and possibly more separable. Suppose the mapping function is $\vec{\phi}$, then the inner product $\langle \vec{x_i}, \vec{x}\rangle$ in Eq. (3) can be replaced by $\langle \vec{\phi}(\vec{x_i}), \vec{\phi}(\vec{x})\rangle$. In SVM, $\vec{\phi}$ is not given explicitly, instead a kernel function $K(\vec{x_i}, \vec{x}) = \langle \vec{\phi}(\vec{x_i}), \vec{\phi}(\vec{x})\rangle$ is used. Consequently, the decision function Eq. (3) becomes:

$$f(\vec{x}) = \operatorname{sign}\left(\sum_{\vec{x_i} \in SVs} y_i \alpha_i K\left(\vec{x_i}, \vec{x}\right) + b\right). \tag{6}$$

The kernel function allows access to spaces of high dimensions without the need to know the mapping function $\vec{\phi}$ explicitly. The performance of SVM depends on the kernel function and its parameters largely. The generally used kernel functions are reported in Table 1. One of the most popular kernel functions is radial basis function (RBF). When use RBF kernel, the parameters (d,g) should be set properly. Generally, d is set to be 2, thus the kernel value is related to the Euclidean distance between the two samples. g is related to the kernel width, it is the key performance factor for SVMs. Too large or too small of a g value will lead to over-fitting or underfitting, respectively. This paper employs the RBF kernel. Then, only g and penalty parameter C need to be optimized for SVM.

Table 1
Generally used kernel functions.

Kernel	Formula
Linear	$K(\vec{x}, \vec{x'}) = \langle \vec{x}, \vec{x'}\rangle$
Polynomial	$K(\vec{x}, \vec{x'}) = (g\langle \vec{x}, \vec{x'}\rangle + r)^d, g > 0$
RBF	$K(\vec{x}, \vec{x'}) = \exp(-g\|\vec{x} - \vec{x'}\|^d), g > 0$
Sigmoid	$K(\vec{x}, \vec{x'}) = \tanh(g\langle \vec{x}, \vec{x'}\rangle + r)$

To show the influence of various (C,g) effect on the performance of SVM, Fig. 1 plots the fivefold average testing accuracy for two public available datasets, vowel [22] and segment [23], in a three dimensional surface, where the x-axis and the y-axis are $\log_2 C$ and $\log_2 g$, respectively. The z-axis is the fivefold average testing accuracy. Each mesh point in the (x,y)-plane stands for a parameter combination (C,g). It is easy to see that the performance of SVM not only fluctuates dramatically with the changes of (C,g), there also are many local maxima. Also these plots show that these surfaces have low-degree of regularity, which further hinders the use of traditional methods. Thus it is significant to adopt some both effective and efficient algorithm to optimize the parameters of SVM.

2.2. Performance measures for SVM parameter optimization

In the process of tuning SVM parameters, performance measures are needed to evaluate the selected parameters. The performance of SVM is mainly referred to its generalization capability, namely the capability of recognizing the new data. Obviously, the true risk of the SVM classifier is the best one. But this quantity is not accessible as the data distribution in real world is unknown. Thus, we have to select estimates or bounds for the true risk of the SVM classifier as the performance measures.

The commonest performance measures are probably the leave-one-out (LOO) procedure [9] and the k-fold cross-validation [24], both of which require the learning engine be trained multiple times to obtain a performance measure for each parameter combination. In LOO procedure one sample is left out in turn for testing, and the training and testing will be repeated l times,

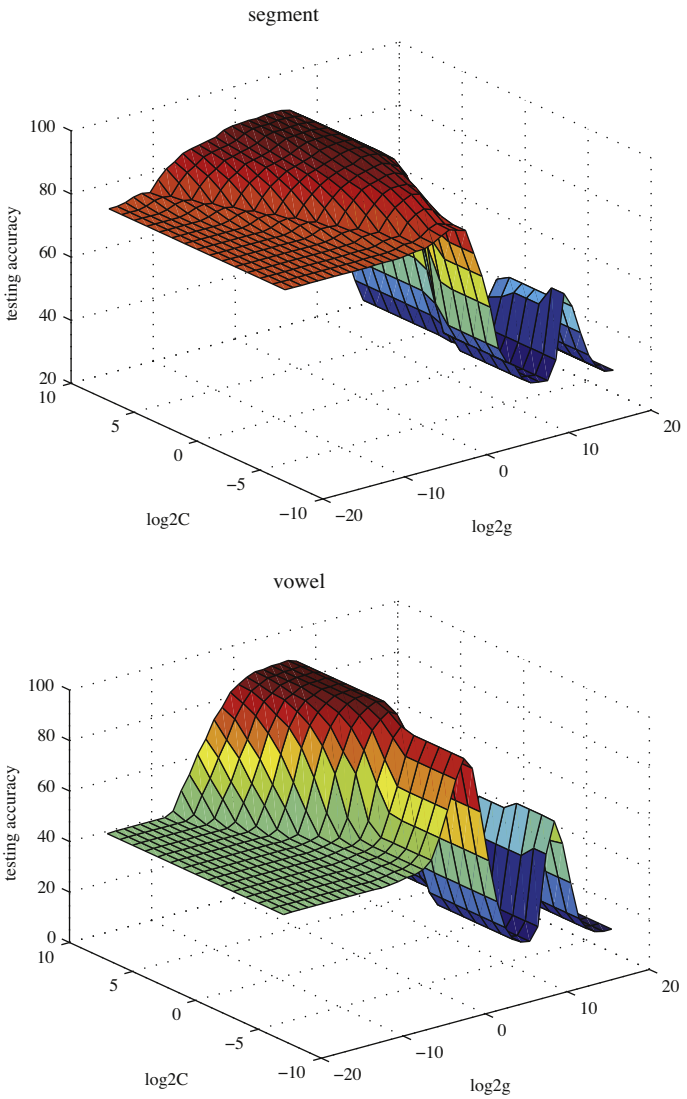

Fig. 1. The performance of RBF kernel SVM varies with parameters (C,g) in two datasets.

where l is the number of training samples. LOO procedure gives an almost unbiased estimate of the expected generalization error. However, it is very costly to actually compute. k-fold cross-validation is an alternative for LOO. In k-fold cross-validation, the training data is randomly split into k mutually exclusive subsets (the folds) of approximately equal sizes. One subset is left out in turn for testing, and the training and testing will be repeated k times. By averaging the test errors over the k trials it gives an estimate of the expected generalization error. However, it also is time-consuming as it should train k SVMs for each parameter combination. Since a non-support vector can be correctly classified by the remaining training samples when it is omitted, a coarse estimate of the LOO generalization error rate can be approximated as: nSV/l, where nSV denotes the number of support vectors, l is the number of training samples [21]. It only needs training one SVM for each parameter combination when nSV/l is selected as performance measure. This measure is both simple and effective.

3. Inter-cluster distance in the feature space

One of the important tricks of SVM is the introduction of kernels, which enables SVM to have the ability of dealing with infinite or nonlinear features in a high dimensional feature space. Training an SVM is equivalent to looking for the hyperplane that separates the data of the two classes in the feature space. When kernel function is selected, the high dimensional feature space is determined by kernel parameters. A better kernel parameter will result in a pair of more separated classes. To judge the degree of the separation of clusters in the feature space, several distance functions are introduced from [18,25,26].

Let the input vectors $\vec{x_i}, i = 1, \ldots, m$, belong to class 1 and the input vectors $\vec{y_j}, j = 1, \ldots, n$, belong to class 2, $\vec{\phi}(\cdot)$ denotes the mapping function which maps the input vectors into the high dimensional feature space, $K(\cdot,\cdot)$ denotes the kernel function.

The Euclidian distance between the closet points of two classes in the feature space is calculated by

$$\begin{aligned}\delta_1 &= \min_{ij}\left\|\vec{\phi}\left(\vec{x_i}\right)-\vec{\phi}\left(\vec{y_j}\right)\right\| = \min_{ij}\sqrt{\left\|\vec{\phi}\left(\vec{x_i}\right)-\vec{\phi}\left(\vec{y_j}\right)\right\|^2} = \min_{ij}\sqrt{\left\|\vec{\phi}\left(\vec{x_i}\right)\right\|^2-2\left(\vec{\phi}\left(\vec{x_i}\right)\cdot\vec{\phi}\left(\vec{y_j}\right)\right)+\left\|\vec{\phi}\left(\vec{y_j}\right)\right\|^2}\\ &= \min_{ij}\sqrt{K\left(\vec{x_i},\vec{x_i}\right)-2K\left(\vec{x_i},\vec{y_j}\right)+K\left(\vec{y_j},\vec{y_j}\right)}\end{aligned} \tag{7}$$

The longest and the average Euclidian distance between two points of two classes in the feature space are calculated by

$$\delta_2 = \max_{ij}\left\|\vec{\phi}\left(\vec{x_i}\right)-\vec{\phi}\left(\vec{y_j}\right)\right\| = \max_{ij}\sqrt{K\left(\vec{x_i},\vec{x_i}\right)-2K\left(\vec{x_i},\vec{y_j}\right)+K\left(\vec{y_j},\vec{y_j}\right)}, \tag{8}$$

$$\delta_3 = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left\|\vec{\phi}\left(\vec{x_i}\right)-\vec{\phi}\left(\vec{y_j}\right)\right\| = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\sqrt{K\left(\vec{x_i},\vec{x_i}\right)-2K\left(\vec{x_i},\vec{y_j}\right)+K\left(\vec{y_j},\vec{y_j}\right)}. \tag{9}$$

The Euclidian distance between the means of two classes in the feature space is calculated by

$$\begin{aligned}\delta_4 &= \left\|\frac{1}{m}\sum_{i=1}^{m}\vec{\phi}\left(\vec{x_i}\right)-\frac{1}{n}\sum_{j=1}^{n}\vec{\phi}\left(\vec{y_j}\right)\right\| = \sqrt{\left\|\frac{1}{m}\sum_{i=1}^{m}\vec{\phi}\left(\vec{x_i}\right)-\frac{1}{n}\sum_{j=1}^{n}\vec{\phi}\left(\vec{y_j}\right)\right\|^2}\\ &= \sqrt{\frac{1}{m^2}\left\|\sum_{i=1}^{m}\vec{\phi}\left(\vec{x_i}\right)\right\|^2-\frac{2}{mn}\left(\sum_{i=1}^{m}\vec{\phi}\left(\vec{x_i}\right)\cdot\sum_{j=1}^{n}\vec{\phi}\left(\vec{y_j}\right)\right)+\frac{1}{n^2}\left\|\sum_{i=1}^{n}\vec{\phi}\left(\vec{y_j}\right)\right\|^2}\\ &= \sqrt{\frac{1}{m^2}\left(\sum_{i=1}^{m}\vec{\phi}\left(\vec{x_i}\right)\cdot\sum_{j=1}^{m}\vec{\phi}\left(\vec{x_j}\right)\right)-\frac{2}{mn}\left(\sum_{i=1}^{m}\vec{\phi}\left(\vec{x_i}\right)\cdot\sum_{j=1}^{n}\vec{\phi}\left(\vec{y_j}\right)\right)+\frac{1}{n^2}\left(\sum_{i=1}^{n}\vec{\phi}\left(\vec{y_i}\right)\cdot\sum_{j=1}^{n}\vec{\phi}\left(\vec{y_j}\right)\right)}\\ &= \sqrt{\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left(\vec{\phi}\left(\vec{x_i}\right)\cdot\vec{\phi}\left(\vec{x_j}\right)\right)-\frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\vec{\phi}\left(\vec{x_i}\right)\cdot\vec{\phi}\left(\vec{y_j}\right)\right)+\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\vec{\phi}\left(\vec{y_i}\right)\cdot\vec{\phi}\left(\vec{y_j}\right)\right)}\\ &= \sqrt{\frac{1}{m^2}\sum_{ij=1}^{m}K\left(\vec{x_i},\vec{x_j}\right)-\frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}K\left(\vec{x_i},\vec{y_j}\right)+\frac{1}{n^2}\sum_{ij=1}^{n}K\left(\vec{y_i},\vec{y_j}\right)}.\end{aligned} \tag{10}$$

In the above four forms of distances, δ_4 is less affected by noises than $\delta_1-\delta_3$. It can indicate the class separation robustly, and perform well in most cases. Here, Fisher's iris data [22] is taken as an example to describe δ_4. Iris dataset includes three different classes of flowers: setosa, virginica and versicolor. Each class consists of 50 samples, and each sample has four attributes. These attributes are sepal length, sepal width, petal length and petal width. Fig. 2 plots the two-dimensional graphic of Fisher's iris data and the distances between different class means, where the x-axis and the y-axis are petal length and petal width, respectively, and d_{svir} denotes the distance between means of setosa and virginic, d_{sver} denotes the distance between means of setosa and versicolor, d_{veir} denotes the distance between means of versicolor and virginic. From Fig. 2 we can see

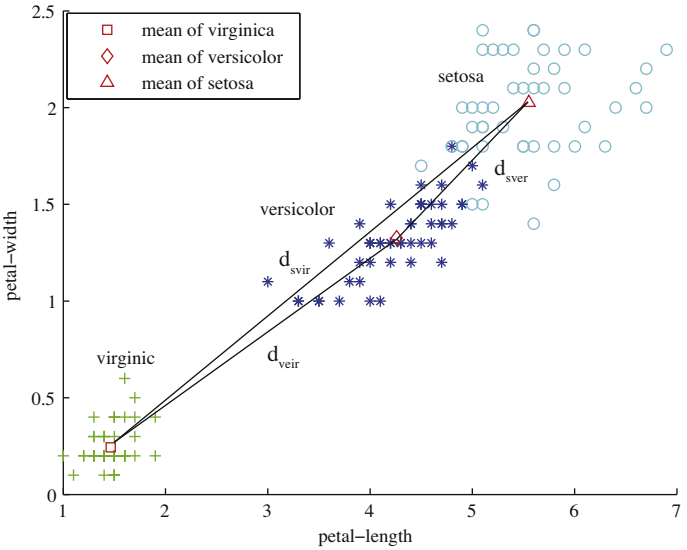

Fig. 2. The distances between classes' means of Fisher's iris data.

that the location of class mean can indicate the class location, as for classes more separate, the distance between class means increases. In this paper δ_4 is selected as the index to choose the optimal kernel parameters for SVMs.

4. Self-adaptive differential evolution

DE, an improved version of GA, is an exceptionally simple global optimization algorithm which is significantly faster and robust at numerical optimization and is more likely to find a function's true global optimum [27,28]. The optimization process of DE is carried out with three basic evolution operators: mutation, crossover, and selection, which are described in detail below.

Without loss of generality, global optimization problems can be transformed into solving the following minimization problem:

$$\begin{aligned} &\min \quad f(x_1, x_2, \ldots, x_D), \\ &\text{Subject to} \quad x_j^L \leqslant x_j \leqslant x_j^U, \quad j = 1, 2, \ldots, D, \end{aligned} \tag{11}$$

where D is the dimension of solution space, x_j^L and x_j^U are the lower and upper bound of x_j.

The initial population of NP vectors is randomly selected based on uniform probability distribution.

$$x_{j,i}(0) = x_{j,i}^L + rand(0, 1) \cdot \left(x_{j,i}^U - x_{j,i}^L\right), \tag{12}$$

where $i = 1,2,\ldots,NP$, $j = 1,2,\ldots,D$, NP denotes population size, $x_{j,i}(0)$ is the jth dimension of $x_i(0)$, rand $(0,1)$ denotes the random number uniformly distribute in $(0,1)$.

Mutation: The mutation of DE is realized by adding the weighted difference vector between two population members to a third member:

$$\overrightarrow{v_i}(g+1) = \overrightarrow{x_{r1}}(g) + F \cdot \left(\overrightarrow{x_{r2}}(g) - \overrightarrow{x_{r3}}(g)\right), \tag{13}$$

where $\overrightarrow{v_i}(g+1)$ is a noisy vector after mutation, $\overrightarrow{x_{r1}}(g)$, $\overrightarrow{x_{r2}}(g)$ and $\overrightarrow{x_{r3}}(g)$ are selected randomly from N_p vectors at the gth generation and $i \neq r_1 \neq r_2 \neq r_3$, the last term represents the mutation step size, F is a scale factor used to control the amplification of the differential variation.

Crossover: DE uses the binomial crossover to create a trial vector.

$$\overrightarrow{u_i}(g+1) = \left(u_{1,i}(g+1), u_{2,i}(g+1), \ldots, u_{j,i}(g+1), \ldots, u_{j,D}(g+1)\right),$$

where

$$u_{j,i}(g+1) = \begin{cases} v_{j,i}(g+1), & \text{if} \quad rand(0, 1) \leqslant CR \quad \text{or} \quad j = j_{rand}, \\ x_{j,i}(g), & \text{otherwise}, \end{cases} \tag{14}$$

while CR is the probability of reproduction, which controls the diversity of the population and aids the algorithm to escape from local optima, j_{rand} is a random number in $[1,2,\ldots,D]$.

Selection: DE implements a very simple selection procedure. The generated offspring (trial vector), $\overrightarrow{u_i}(g+1)$, replaces the parent, $\overrightarrow{x_i}(g)$, only if the fitness of the offspring is better than that of the parent.

$$\overrightarrow{x_i}(g+1) = \begin{cases} \overrightarrow{u_i}(g+1), & \text{if} \quad f\left(\overrightarrow{u_i}(g+1)\right) \leqslant f\left(\overrightarrow{x_i}(g)\right) \\ \overrightarrow{x_i}(g), & \text{otherwise} \end{cases} \tag{15}$$

The key control parameters in DE are population size *NP*, scaling factor *F*, and crossover rate *CR*. In practice, *F* is not easy to set, as it controls the amplification of the differential variation. A too big *F* will result in random search, and a too small *F* will cause a low differential variation, which can make the algorithm premature. Hence, it is important to select a proper *F* for DE.

In this paper, we use a self-adaptive scaling factor operator [29], in which the scaling factor is determined according to the evolved generations. The adaptive scaling factor operator is designed as follow:

$$\lambda = \exp\left(1 - \frac{Gm}{G_m + 1 - G}\right), \tag{16}$$

$$F = F_0 * 2^{\lambda}, \tag{17}$$

where F_0 is mutation parameter, Gm is the maximum evolution generation, and G is the current evolution generation. At the initial generations, $F = 2F_0$, it has a bigger scaling factor to preserve the diversity of individuals. With the running of the algorithm, the scaling factor reduces gradually. In the later stage, F is close to F_0, which can enhance the probability of obtaining the global optimum.

5. Multi-class SVM optimized by ICDF and self-adaptive DE

5.1. Procedures of the proposed method

In real world, most of pattern recognition problems are multi-class. However, the SVMs were originally conceived for the solution of binary classification problems. Extending binary SVM to multi-class problems is still an ongoing research issue. Several methods have been proposed for solving multi-class problems applying the SVMs, such as one-against-one [30], one-against-rest [31] and direct acyclic graph (DAG) SVM [32] strategies. From the research of Hsu and Lin [8], the one-against-one method outperforms other methods for extending binary SVM to multi-class.

When one-against-one strategy is used to extend the binary SVM to n-class, it will produce $n(n-1)/2$ SVM classifiers. There are two methods to set the parameters for these $n(n-1)/2$ SVM classifiers. The first one is searching for a set of parameter values which will be common to all classifiers. The second approach, a set of values is selected for each binary classifier. Intuitively, the second one is ideal, but it is complex, especially for the problem with very many classes. Relate work [11,33] have shown that the use of the same parameter values in all binary SVMs is sufficient to obtain good results in most of the problems. In this paper, we use the same parameter values in all binary SVMs. The optimal parameter combination is searched in the intervals of kernel parameter and penalty parameter. Obviously, the wider or the finer the search intervals are, the more possibility it finds the best parameter combination. However, the wider the search intervals are, the more time it consumes. Consequently, it is significant to determine the both small and effective search intervals of kernel parameter and penalty parameter.

The optimal kernel parameters for each class pair can be found by calculating the ICDFs. The optimal kernel parameter for the multi-class problem most likely appears in the continuous search interval covering the optimal kernel parameter of each class pair. Searching the optimal parameter combination of (C,g) essentially is an optimization problem. Thus, ADE is used to search the best (C,g) in the continuous intervals of kernel parameter and penalty parameter. The overall process of the proposed method is illustrated in Fig. 3. It mainly contains the following steps.

Step 1: For each class pair i of $n(n-1)/2$ pairwise subsets of a n-class problem, the ICDFs for all the candidate kernel parameters are calculated and the largest one with its corresponding kernel parameter value g_i is found.
Step 2: Sort these g_i, where $i = 1,2,\ldots,n(n-1)/2$, to find the minimum $g_{\min}$ and the maximum $g_{\max}$.
Step 3: Determine the new search interval of kernel parameter g as $(g_{\min}, g_{\max})$.
Step 4: The population of (C,g) is randomly initialized in $(C_{\min}, C_{\max})$ and $(g_{\min}, g_{\max})$.
Step 5: SVM is trained with each parameter combination (C,g), and its corresponding fitness value for the SVM is evaluated. If the fitness value satisfies the termination criterion, then the parameter combination (C,g) is selected as the optimal one; If the fitness value does not satisfy the termination criterion, then the algorithm enters the next generation to evolve.
Step 6: According to Eqs. (12)–(17), perform mutation, crossover and selection for the current population to generate a new population of (C,g) in $(C_{\min}, C_{\max})$ and $(g_{\min}, g_{\max})$. Go to step 5.
Step (5)–(6) are repeated until the termination condition is satisfied.
Step 7: Train one-against-one multi-class SVM with the obtained optimal parameters and get the trained SVM model.
Step 8: Use the trained SVM model to predict the test data.

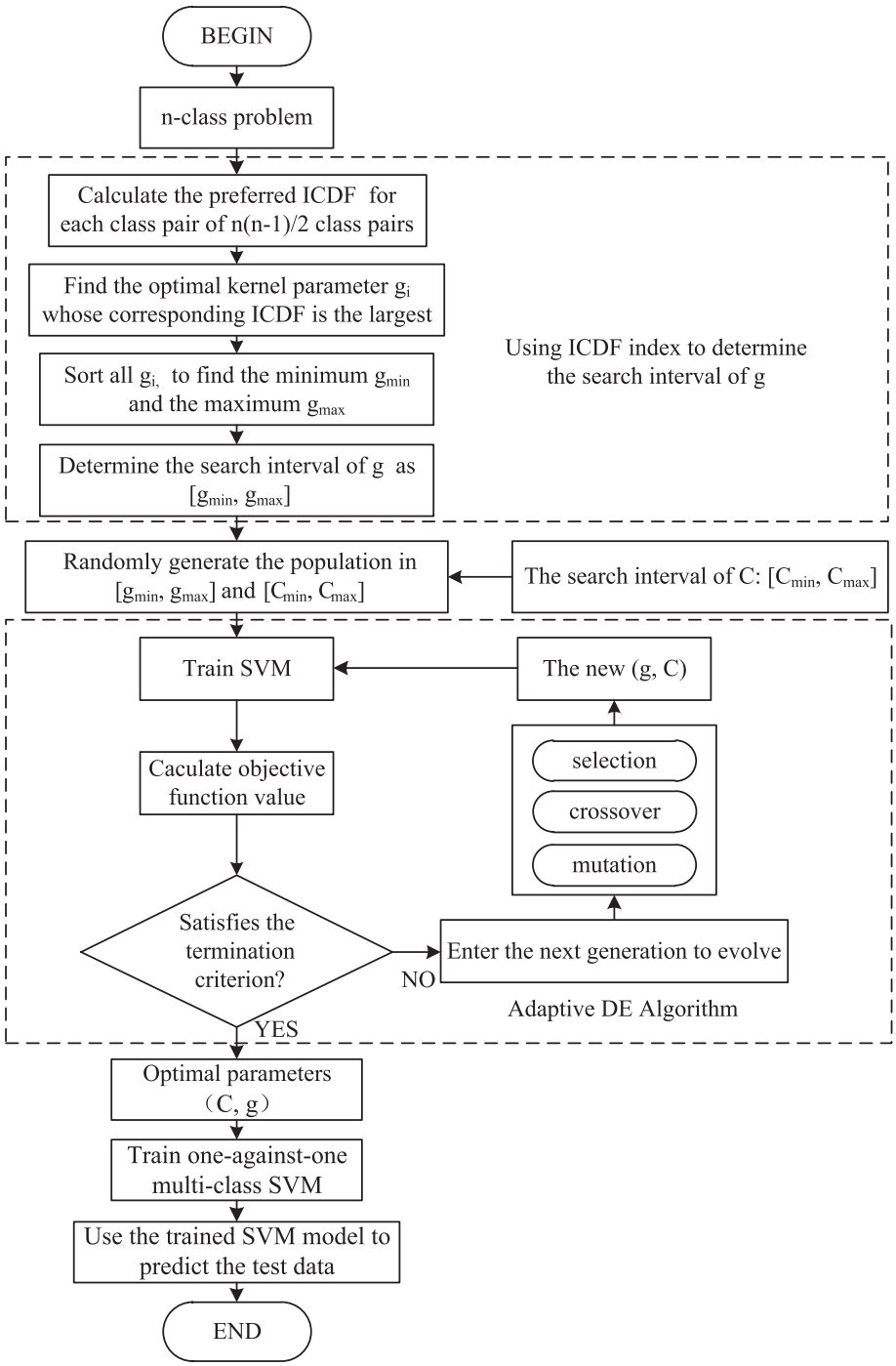

Fig. 3. The overall process of the proposed method.

5.2. Numerical experiments

In this section, experimental results on several problems from the UCI Repository of machine learning datasets [22] and Statlog collection [23] are presented to evaluate the proposed method. From UCI Repository the following datasets: iris, wine, glass, and vowel are selected. The datasets: vehicle, segment, dna, satimage are from Statlog collection. Table 2 shows the basic information of these datasets. Note that except problem dna whose attributes are binary value, all training data are scaled to be in $[-1,1]$. Then test data are adjusted to $[-1,1]$ accordingly, and training sets of dna and satimage are further separated to training and validation sets.

Table 2
The basic information of datasets.

Dataset	Number of training data	Number of validation data	Number of testing data	Number of class	Number of attributes
Iris	150	0	0	3	4
Wine	178	0	0	3	13
Glass	214	0	0	6	13
Vowel	528	0	0	11	10
Vehicle	846	0	0	4	18
Segment	2310	0	0	7	19
Dna	1400	600	1186	3	180
Satimage	3104	1331	2000	6	36

The computing platform is a PC with the following features: Intel Pentium IV 3.0 GHz CPU, 1 GB RAM, a Windows XP operating system and the Visual Studio 2008 and MATLAB R2011a development environment. All the experiments are done with the help of LIBSVM [24]. We use the RBF kernel with $d = 2$, $C \in [2^{-7}, 2^{7}]$ and the candidate g are $[2^{-20}, 2^{-19}, \ldots, 2^{20}]$. The parameters employed for the ADE are set as follow: Population size $NP = 40$. Probability of reproduction $CR = 0.9$, maximum evolution generation $Gm = 200$, the mutation parameter of adaptive scaling factor $F_0 = 0.5$. When the number of iterations exceeds $Gm = 200$, or when the fitness value does not change for 20 times iterations the algorithm is terminated.

For datasets dna and satimage where training, validation and testing sets are available, the validation rate is selected as fitness function. For each pair of (C,g), the validation performance is measured by training the training set and testing the validation set. Then, the pair of (C,g) that achieves the best validation rate is used to train all the training and validation set. Finally, the trained SVM model is used to predict the testing set and the testing rate is reported. For the other six datasets where validation set and testing set may not be available, for each pair of (C,g), we conduct a tenfold cross-validation on the

Table 3
A comparison between the proposed method and the traditional grid search (better rate bold-faced).

Datasets	The proposed method			Grid search		
	Search interval of g	Chosen (C,g)	Rate (%)	Search interval of g	Chosen (C,g)	Rate (%)
Iris	$[2^{-3},2^{-1}]$	(0.66,0.22)	97.78	$[2^{-20},2^{20}]$	$(2^{-2},2^{-1})$	97.78
Wine	$[2^{-2},2^{-1}]$	(1.15,0.25)	**99.44**	$[2^{-20},2^{20}]$	$(2^{-1},2^{-3})$	99.20
Glass	$[2^{-1},2^{5}]$	(1.15,8)	**71.50**	$[2^{-20},2^{20}]$	$(2^{1},2^{2})$	69.16
Vowel	$[2^{-1},2^{3}]$	(6.06,1.52)	**99.05**	$[2^{-20},2^{20}]$	$(2^{2}, 2^{1})$	98.73
Vehicle	$[2^{-2},2^{0}]$	(55.72,0.25)	85.14	$[2^{-20},2^{20}]$	$(2^{6},2^{-2})$	85.14
Segment	$[2^{-3},2^{0}]$	(111.43,0.38)	**97.32**	$[2^{-20},2^{20}]$	$(2^{7},2^{-6})$	97.14
Dna	$[2^{-6},2^{-1}]$	(12.96,0.15)	**95.87**	$[2^{-20},2^{20}]$	$(2^{3},2^{-5})$	95.45
Satimage	$[2^{0},2^{4}]$	(3.78,4.05)	**92.35**	$[2^{-20},2^{20}]$	$(2^{3},2^{1})$	91.30

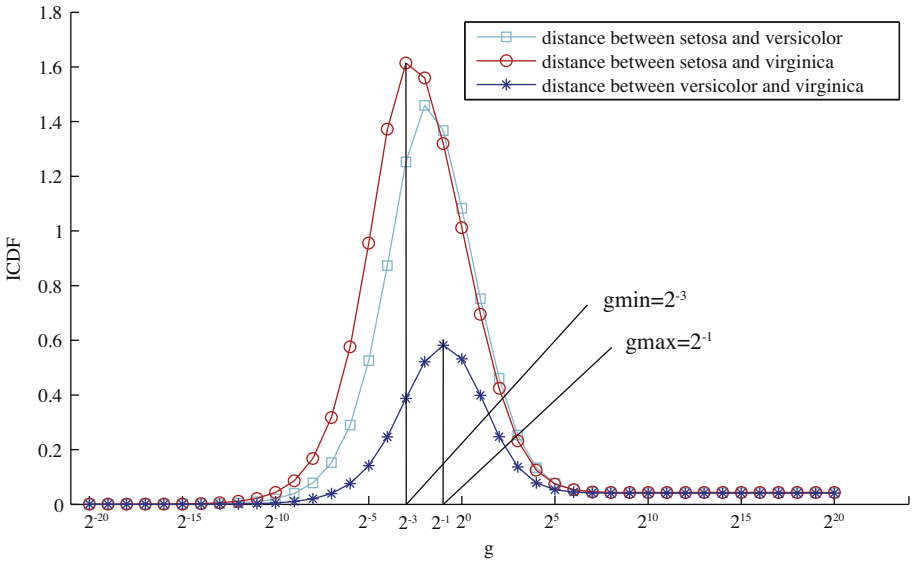

Fig. 4. The ICDFs for Iris data.

whole training data. In tenfold cross-validation, the training data is randomly split into ten mutually exclusive subsets (the folds) of approximately equal sizes. One subset is left out in turn for testing while the other nine subsets are in turn for training and the training and testing are repeated 10 times. The tenfold cross-validation rate is selected as fitness value for ADE. At last, we report the optimal (C, g) and its corresponding best cross-validation rate.

The results are compared to those of the traditional GS method [8]. For GS the search step is set as 1 and the search range of C and g are $[2^{-7},2^{7}]$, $[2^{-20},2^{20}]$, respectively.

The comparison using the proposed method and GS is presented in Table 3 where the search interval of g, the chosen optimal (C,g), the best validation rate or testing rate are compared. From Table 3 we can see that the search ranges of g in all datasets are largely shortened by ICDF heuristic. Fig. 4 shows the ICDFs of iris data problem. In Fig. 4, $g_{\min} = 2^{-3} = 0.125$, $g_{\max} = 2^{-1} = 0.5$, so the new search interval of g is determined as $[0.125, 0.5]$, which covers the maximum ICDF value of every class pair. It is much smaller than that of traditional GS. Note that in GS the values of the chosen optimal C and g are the power of 2, as in GS method both C and g are sampled as the integer power of 2. Obviously, it may omit the optimal (C,g) by using the traditional GS method as the interval between two adjacent grid points is too large. For example the interval value between 2^2 and 2^3 is $2^3-2^2 = 4$. From Table 3 we also note that the proposed method yields better validation or testing rate in datasets: wine, glass, vowel, segment, dna and satimage while in other datasets the two methods produce the same results. These results prove that the proposed method outperforms the traditional GS in searching the optimal parameters for multi-class SVM, as in the proposed method the ICDF can determine an effective and small range of g, and the ADE searches the optimal (C,g) in the continuous intervals of C and g, which can avoid omitting the optimal parameters.

6. Applications in fault diagnosis for rolling element bearings

Rolling element bearings constitute a major part of almost every rotating machine, as for it is the interface between the stationary and the rotating part of the machine. Faults occurring in the bearings may lead to fatal breakdowns of machines and can drive to unacceptably long time maintenance stops, which will result in large personal casualties and economical loss. Therefore, it is significant to detect fast, accurately and easily the existence and severity of the faults in the bearings. Visual inspection of frequency–domain features of the vibration signals may be sufficient to identify the faults, but often this kind of diagnostic requires a certain level of expertise to be carried out. Reliable, fast and automated diagnostic techniques allowing relatively unskilled operators to make important decisions without a specialist to examine the data are therefore valuable [34]. Much research has been devoted to this subject [34–36]. Vibration-based monitoring is the most widely applied technique. It is possible to obtain vital diagnosis information from the vibration signals by using some signal processing techniques. Then, the feature extraction and selection are undertaken. At last, the pattern recognition methods are used to diagnose the faults. In this section the proposed ICDF–ADE method is applied to fault diagnosis for rolling element bearings.

6.1. Data collection and feature extraction

Data collection: The data used in this paper comes from the dataset of the rolling element bearings [37], by installing the deep groove ball bearings manufactured by SKF in a motor-driven mechanical system. Three faults, outer race fault, inner race fault and ball fault, are introduced into the drive-end bearing of the motor. The fault seeded at the outer race was placed at a position equivalent to 12:00 o'clock time configuration. The defect sizes (diameter, depth) of these three faults are the same: 0.007 or 0.021 in. Each bearing was tested under the four different loads, 0, 1, 2 and 3 hp. Data was collected at 12000 samples/s and 24000 samples/s. In this paper, we use the data that collected at 24000 samples/s under 2 hp load, and the defect sizes are 0.007 or 0.021 in. Each original signal was divided into 100 signals, and the length of the signal after divided is 4096 data points. Consequently, 600 samples covering 6 types' faults were produced, as list in Table 4. The time responses of these three types' faults are shown in Fig. 5.

Feature extraction: According to literature [36], one time-domain feature (F1), two frequency-spectrum features (F2, F3) and one demodulation-spectrum feature (F4) were selected for fault diagnosis of the rolling element bearings. These four features were selected from 43 features covering time-domain statistical characteristics, frequency-domain statistical characteristics and empirical mode decomposition (EMD) energy entropies. They provide dominant fault-related information of vibration signals. Consequently, they are superior to other features in fault diagnosis for rolling element bearings.

Table 4
Description of bearing fault dataset.

The number of fault samples	Defect size (in)	Position of fault	Label of class
100	0.007	Outer race	1
100	0.007	Inner race	2
100	0.007	Ball	3
100	0.021	Outer race	4
100	0.021	Inner race	5
100	0.021	Ball	6

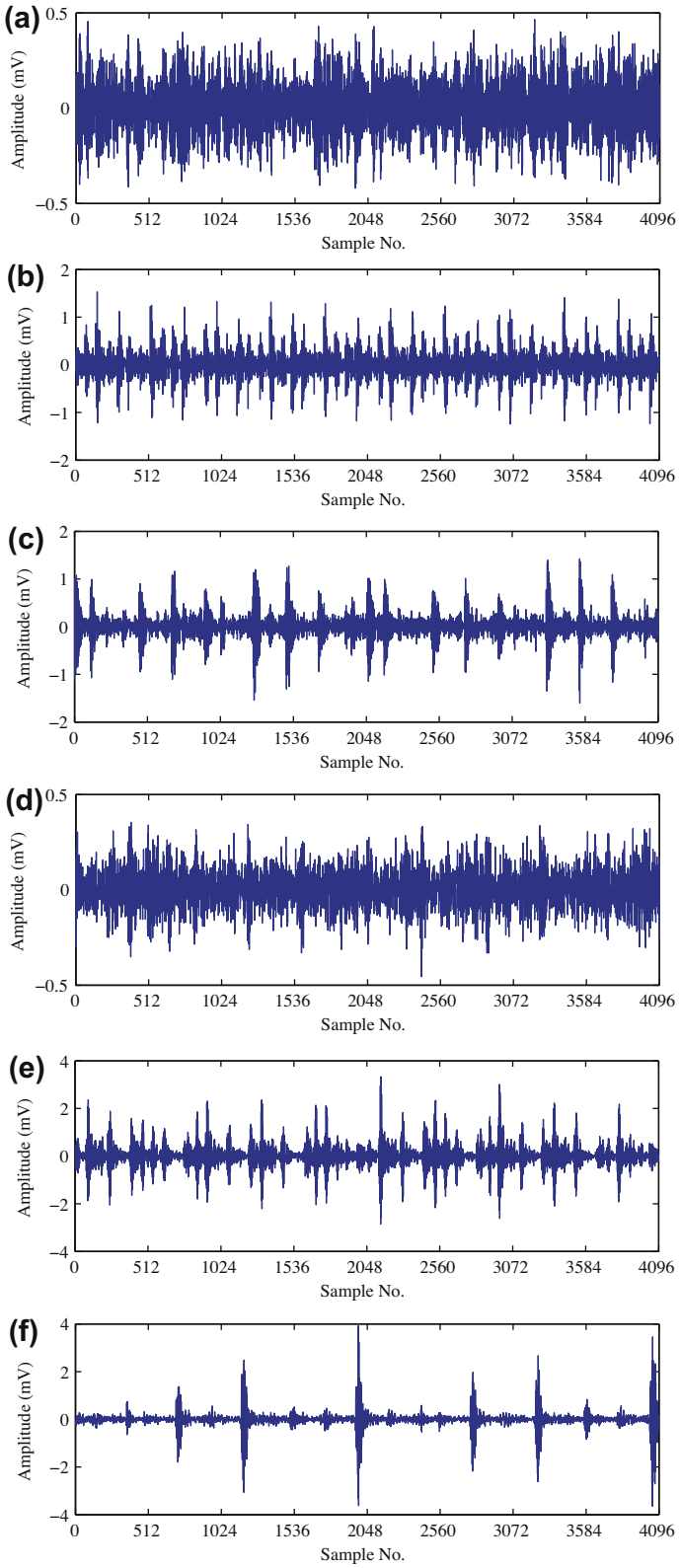

Fig. 5. Time series of fault signals. (a) Ball fault with 0.007 in. (b) Inner race fault with 0.007 in. (c) Outer race fault with 0.007 in. (d) Ball fault with 0.021 in. (e) Inner race fault with 0.021 in. (f) Outer race fault with 0.021 in.

6.2. Fault diagnosis by proposed method

After data collection and feature extraction, 600 samples with four dimensions were obtained. All of the data were scaled to be in [0,1]. By using the first 3 attributes, F1, F2, F3, the three-dimensional data distribution of the fault samples is plotted in Fig. 6. Where b007, b021, ir007, ir021, or007, or021 denote ball fault with 0.007 in, ball fault with 0.021in, inner race fault with 0.007 in, inner race fault with 0.021 in, outer race fault with 0.007 in and outer race fault with 0.021 in, respectively.

The experimental setup and the parameter setting for ADE are the same as that of Section 5.2. The experiments were repeated 30 times. In each time experiment, 30 samples for each class were randomly selected as training data and the rest 70 samples were selected as testing data. The results obtained by the proposed method were compared with those of traditional GS and ADE without ICDF. For the method ADE without ICDF, the ADE algorithm is directly used to search the optimal parameters (C,g) in $[2^{-7},2^{7}]$ and $[2^{-10},2^{10}]$. The parameters for ADE without ICDF are the same as that of ICDF–ADE. Fitness function has a decisive impact on the diagnosis results. Two types of fitness, fivefold cross-validation rate and nSV/l, which have been described in Section 2.2, were selected for each method respectively and the results were compared.

Table 5 shows the comparison of diagnostic results. The values of chosen optimal (C,g) are the optimal parameters in one time experiment. In the "mean (%) and variance of testing rate" column the average testing rate and its standard deviation value are reported. In the last column, we report the average testing time and its standard deviation value. For the proposed method, the training time is the sum of the time for calculating ICDF values for all kernel parameters and the time for training SVMs.

Fig. 7 shows testing accuracy for different methods with two types of fitness function. It is clear from Table 5 and Fig. 7 that the proposed method ICDF–ADE yields the best testing rate in these three methods. We also note that there are no significant differences of testing accuracy between these two types of fitness for the same method. These results suggest that the ICDF can determine an effective and small range of g. ADE searches the optimal (C,g) in the continuous intervals of C and g, which can avoid omitting the optimal parameters. And selecting nSV/l as fitness function can achieve the same performance as fivefold cross-validation. The training time for different methods with two types of fitness function is presented in Fig. 8. From Table 5 and Fig. 8 we can see that for each method fivefold cross-validation consumes much more time than nSV/l, as it needs training five SVM models for each parameter combination by using fivefold cross-validation, while it only needs training one SVM model for each parameter combination by using nSV/l. From Fig. 8, it also can be seen that ADE is the

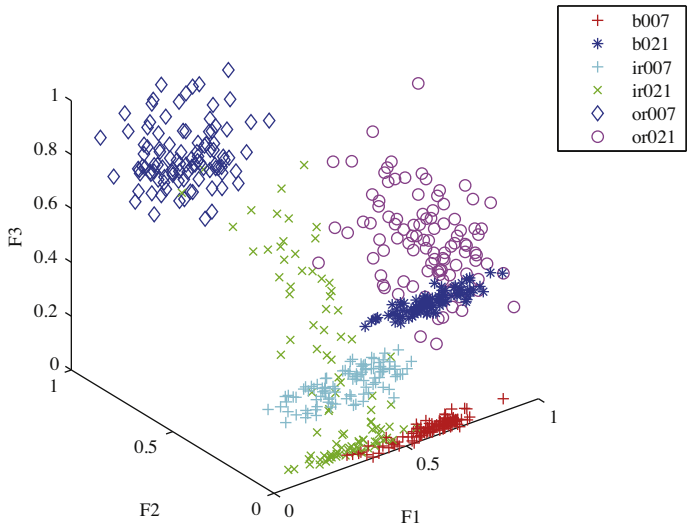

Fig. 6. The data distribution of 6 classes' faults of rolling element bearings.

Table 5
Comparison of diagnostic results.

Fitness	Algorithms	Chosen optimal (C,g)	Mean (%) and variance of testing rate	Mean(s) and variance of training time
Fivefold cross-validation rate	GS	(128,4)	94.98, 1.36	35.40, 0.070
	ADE	(74.44,37.48)	95.33, 2.15	205.21, 21.261
	ICDF–ADE	(126.0,6.16)	96.23, 2.48	48.11, 13.459
nSV/l	GS	(128,2)	94.48, 2.58	11.01, 0.008
	ADE	(21.31,75.14)	95.43, 1.53	54.15, 3.469
	ICDF–ADE	(75.59,2.59)	96.83, 2.55	13.93, 1.255

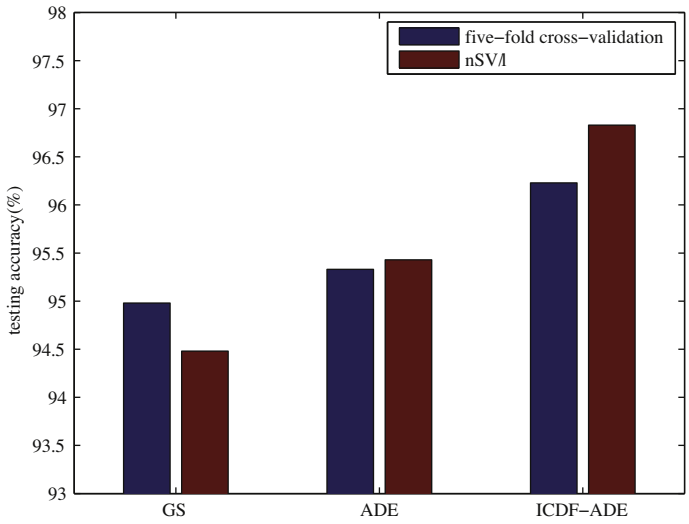

Fig. 7. Bar plot of average testing accuracy for different methods with two types of fitness function in Table 5.

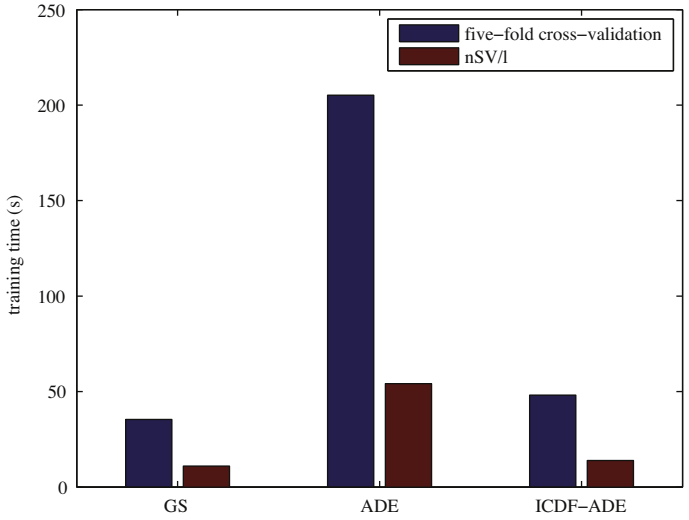

Fig. 8. Bar plot of average training time for different methods with two types of fitness function in Table 5.

most time-consuming method in these three methods. In ADE method, the search intervals of (C,g) are $[2^{-7},2^{7}]$ and $[2^{-10},2^{10}]$ while in ICDF–ADE the search interval of g is largely shortened by ICDF. Since calculating ICDF is a simple non-iterated process and does not require the information of the trained classifiers, the training time of ICDF–ADE is much less than that of ADE. Note that the training time of GS is slightly less than that of ICDF–ADE as ADE selects the optimal parameter combination from the population evolved generation by generation, it may require training more SVMs than GS even the search interval of g is shortened. For GS it needs to train the same number of SVMs for each pair of (C,g), hence the standard deviation values of both training time and testing accuracy for GS are lower than those of the other two methods which can be seen from Table 5. However, ICDF–ADE also yields satisfactory results of standard deviation values. The combined results suggest that the proposed method, ICDF–ADE with nSV/l as fitness function, can stably find the optimal parameters for multi-class SVM with high testing accuracy and satisfactory training time. Thus, it is feasible for fault diagnosis of rolling element bearings.

7. Conclusion

In this paper, the inter-cluster distances in the feature spaces and self-adaptive differential evolution algorithm were used to optimize the parameters of multi-class SVM. For each class pair of a multi-class problem, the inter-cluster distances were

calculated, and the optimal kernel parameter for the SVM of this class pair was found according to the largest inter-cluster distance. Then, a small and effective search interval of kernel parameter covering the optimal kernel parameter of each class pair was determined. Self-adaptive differential evolution algorithm was used to search the optimal parameter combination in the search intervals of kernel parameter and penalty parameter. In the proposed method, the search interval of kernel parameter was much smaller than that of traditional methods, and Self-adaptive differential evolution algorithm could search the optimal (C,g) in the continuous intervals of C and g, which can avoid omitting the optimal parameters. Hence, the proposed method can yield high testing rate while the training time is much shortened. Some standard datasets were used to evaluate the proposed method. At last the proposed method was used to diagnose the faults of rolling element bearings. The experimental results show that the proposed method is both effective and computationally efficient for parameters optimization of multi-class SVM.

Acknowledgements

We would like to thank Kenneth A. Loparo, Ph.D., Case Western Reserve University, for his experimental data provided. In addition, the work described in this paper is supported by the National Natural Science Foundation of China under Grant No. 51079057, the National Scientific and Technological Support Projects of China under Grant No. 2008BAB29B05 and National Natural Science Foundation of China under Grant No. 51109088.

References

[1] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, New York, 1999.
[2] K. Polat, S. Gunes, Detection of ECG Arrhythmia using a differential expert system approach based on principal component analysis and least square support vector machine, Appl. Math. Comput. 186 (2007) 898–906.
[3] K. Polat, S. Gunes, A novel approach to estimation of *E. coli* promoter gene sequences: combining feature selection and least square support vector machine (FS_LSSVM), Appl. Math. Comput. 190 (2007) 1574–1582.
[4] O. Amayri, N. Bouguila, A study of spam filtering using support vector machines, Artif. Intell. Rev. 34 (2010) 73–108.
[5] T.S. Guzella, W.M. Caminhas, A review of machine learning approaches to spam filtering, Expert Syst. Appl. 36 (2009) 10206–10222.
[6] S.F. Yuan, F.L. Chu, Fault diagnosis based on support vector machines with parameter optimisation by artificial immunisation algorithm, Mech. Syst. Signal. Process. 21 (2007) 1318–1330.
[7] Q. Wu, Z.H. Ni, Car assembly line fault diagnosis based on triangular fuzzy Gaussian support vector classifier machine and modified genetic algorithm, Expert Syst. Appl. 38 (2011) 4734–4740.
[8] C.W. Hsu, C.J. Lin, A comparison of methods for multiclass support vector machines, IEEE Trans. Neural Networ. 13 (2002) 415–425.
[9] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Mach. Learn. 46 (2002) 131–159.
[10] S. Keerthi, V. Sindhwani, O. Chapelle, An efficient method for gradient-based adaptation of hyperparameters in SVM models, Adv. Neur. In. 19 (2007) 673–680.
[11] A.C. Lorena, A.C.P.L.F. de Carvalho, Evolutionary tuning of SVM parameter values in multiclass problems, Neurocomputing 71 (2008) 3326–3334.
[12] L.M. Saini, S.K. Aggarwal, A. Kumar, Parameter optimisation using genetic algorithm for support vector machine-based price-forecasting model in National Electricity Market, IET Gener. Transm. Distrib. 4 (2010) 36–49.
[13] S.T. Li, M.K. Tan, Tuning SVM parameters by using a hybrid CLPSO–BFGS algorithm, Neurocomputing 73 (2010) 2089–2096.
[14] S.W. Lin, K.C. Ying, S.C. Chen, Z.J. Lee, Particle swarm optimization for parameter determination and feature selection of support vector machines, Expert Syst. Appl. 35 (2008) 1817–1824.
[15] I. Aydin, M. Karakose, E. Akin, A multi-objective artificial immune algorithm for parameter optimization in support vector machine, Appl. Soft Comput. 11 (2011) 120–129.
[16] X.L. Zhang, X.F. Chen, Z.J. He, An ACO-based algorithm for parameter optimization of support vector machines, Expert Syst. Appl. 37 (2010) 6618–6628.
[17] K.-P. Wu, S.-D. Wang, Choosing the kernel parameters of support vector machines according to the inter-cluster distance, in: International Joint Conference on Neural Networks 2006, IJCNN'06, July 16, 2006 – July 21, 2006, Institute of Electrical and Electronics Engineers Inc, Vancouver, BC, Canada, 2006, pp. 1205–1211.
[18] K.P. Wu, S.D. Wang, Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space, Pattern Recogn. 42 (2009) 710–717.
[19] N.E. Ayat, M. Cheriet, C.Y. Suen, Automatic model selection for the optimization of SVM kernels, Pattern Recogn. 38 (2005) 1733–1745.
[20] J. Vesterstrom, R. Thomsen, A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, in: Proceedings of the 2004 Congress on Evolutionary Computation, 19–23 June, 2004, IEEE, Piscataway, NJ, USA, 2004, pp. 1980–1987.
[21] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, 2000.
[22] C.L. Blake, C.J. Merz, UCI Repository of Machine Learning Databases, Department of Information and Computer Science, University of California, Irvine, 1998. <http://www.ics.uci.edu/~mlearn/MI.Repository.html> (last accessed 8.28.11).
[23] D. Michie, D.J. Spiegelhalter, C.C. Taylor, Machine Learning, Neural and Statistical Classification. <ftp.ncc.up.pt/pub/statlog/>, 1994 (last accessed 8.28.11.
[24] C.-W. Hsu, C.-C. Chang, C.-J. Lin, A Practical Guide to Support Vector Classification. <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>, 2009 (last accessed 8.28.11).
[25] J.C. Bezdek, N.R. Pal, Some new indexes of cluster validity, IEEE Trans. Syst. Man Cybernet. B 28 (1998) 301–315.
[26] R. Debnath, N. Takahide, H. Takahashi, A decision based one-against-one method for multi-class support vector machine, Pattern Anal. Appl. 7 (2004) 164–175.
[27] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359.
[28] K. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer-Verlag, New York, 2005.
[29] X.-F. Yan, J. Yu, F. Qian, J.-W. Ding, Kinetic parameter estimation of oxidation in supercritical water based on modified differential evolution, J. East China Univ. Sci. Technol. (Natural Science Edition) 32 (2006). pp. 94–97, 124, in Chinese.
[30] U.H.-G. Kreßel, Pairwise classification and support vector machines, in: Advances in Kernel Methods, MIT Press, 1999, pp. 255–268.
[31] L. Bottou, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, L.D. Jackel, Y. LeCun, U.A. Muller, E. Sackinger, P. Simard, V. Vapnik, Comparison of classifier methods: a case study in handwritten digit recognition, in: Proceedings of 12th International Conference on Pattern Recognition, 9–13 October, 1994, IEEE Computer Society Press, Los Alamitos, CA, USA, 1994, pp. 77–82.

[32] J.C. Platt, N. Cristianini, J. Shawe-Taylor, Large margin DAGs for multiclass classification, Adv. Neur. In. 12 (2000) 547–553.
[33] M. Liepert, Topological fields chuncking for German with SVMs: optimizing SVM-parameters with GAs, in: Proceedings of the International Conference on Recent Advances in Natural Language Processing, RANLP, 2003, pp. 32–45.
[34] A. Rojas, A.K. Nandi, Practical scheme for fast detection and classification of rolling-element bearing faults using support vector machines, Mech. Syst. Signal. Process. 20 (2006) 1523–1536.
[35] I. Trendafilova, An automated procedure for detection and identification of ball bearing damage using multivariate statistics and pattern recognition, Mech. Syst. Signal. Process. 24 (2010) 1858–1869.
[36] Y.G. Lei, Z.J. He, Y.Y. Zi, A new approach to intelligent fault diagnosis of rotating machinery, Expert Syst. Appl. 35 (2008) 1593–1600.
[37] K.A. Loparo, Bearings Vibration Dataset, Case Western Reserve University, 2011. <http://www.eecs.case.edu/laboratory/bearing/download.htm> (last accessed 08.28.11).