Mechanical Systems and Signal Processing 81 (2016) 274–293

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

One-class classification based on the convex hull for bearing fault detection

Ming Zeng [a,b], Yu Yang [a,b,*], Songrong Luo [a,b], Junsheng Cheng [a,b]

[a] State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha 410082, PR China
[b] College of Mechanical and Vehicle Engineering, Hunan University, Changsha 410082, PR China

ARTICLE INFO

Article history:
Received 4 April 2015
Received in revised form
8 September 2015
Accepted 1 April 2016
Available online 6 April 2016

Keywords:
One-class classification
Nearest point problem
Generalized Gilbert algorithm
Reduced convex hull
Bearings
Fault detection

ABSTRACT

Originating from a nearest point problem, a novel method called one-class classification based on the convex hull (OCCCH) is proposed for one-class classification problems. The basic goal of OCCCH is to find the nearest point to the origin from the reduced convex hull of training samples. A generalized Gilbert algorithm is proposed to solve the nearest point problem. It is a geometric algorithm with high computational efficiency. OCCCH has two different forms, i.e., OCCCH-1 and OCCCH-2. The relationships among OCCCH-1, OCCCH-2 and one-class support vector machine (OCSVM) are investigated theoretically. The classification accuracy and the computational efficiency of the three methods are compared through the experiments conducted on several benchmark datasets. Experimental results show that OCCCH (including OCCCH-1 and OCCCH-2) using the generalized Gilbert algorithm performs more efficiently than OCSVM using the well-known sequential minimal optimization (SMO) algorithm; at the same time, OCCCH-2 can always obtain comparable classification accuracies to OCSVM. Finally, these methods are applied to the monitoring model constructions for bearing fault detection. Compared with OCCCH-2 and OCSVM, OCCCH-1 can significantly decrease the false alarm ratio while detecting the bearing fault successfully.

1. Introduction

The problem of classification can be defined as a problem of assigning a sample represented by a feature vector to a class (or category). Using a set of samples from the training set, a classifier learns to assign class labels to unseen samples from the test set. In traditional binary or multi-class classification problems, samples from each class are generally required for the classifier design. But in some cases, only the samples from one-class are available, and those from other classes are too difficult or too costly to acquire. This kind of classification is known as one-class classification. The available class is called the target class or the negative class, while all other possible classes are collectively called the non-target class or the positive class. By learning from a training set containing only samples from the target class, a one-class classifier tries to describe the target class and detect the samples not belonging to it. One-class classification tasks can be found in many real world applications, such as network security [1–4], medical treatment [5–8] and process monitoring [9–12]. For example, in rotating machinery systems, bearing faults cause long machine downtime, expensive maintenance cost and even human

* Corresponding author at: State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha 410082, PR China
E-mail addresses: zeng_m@126.com (M. Zeng), yangyu@hnu.edu.cn (Y. Yang), luosongrong@126.com (S. Luo), signalp@tom.com (J. Cheng).

http://dx.doi.org/10.1016/j.ymssp.2016.04.001
0888-3270/

casualties. As bearing failure patterns are unpredictable, few faulty samples exist in the history data; furthermore, these samples usually cannot characterize the failure patterns comprehensively. In this case, only the samples acquired from the normal operating condition are available to build a monitoring model. This is exactly a typical one-class classification problem. For these applications mentioned above, it is difficult or impossible to acquire a set of samples that provides a comprehensive characterization for all the possible classes. Therefore, one-class classification plays a significant role in solving such practical problems. Over recent years, many methods have been proposed to solve one-class classification problems. From a methodological viewpoint, one-class classification methods can be roughly grouped into three main categories [13,14]: density methods, reconstruction methods and boundary methods.

Density methods, as its name implies, are mostly to estimate the underlying probability density function (PDF) of the distribution of training samples and to set a threshold on this density. In the classification phase, the PDF values of new samples are compared against the threshold. Typical density methods include Gaussian mixture models (GMM) [15], K-nearest neighbors estimation (KNN) [16] and Parzen-window density estimation (PWD) [16]. A common problem of density methods is that they need a sufficiently large number of samples to make the probability density estimate statistically significant, especially when the dimension of data is high. GMM assumes that data follow multiple Gaussian distributions, therefore the applications of GMM may be restricted in situations where the assumption does not necessarily hold. Moreover, GMM is sensitive to outliers in the training set because the variance introduced by outliers can overwhelm the variance of training samples completely [13]. KNN and PWD make no assumption about the data distribution, thus being able to approximate an arbitrary distribution whose parametric form is unknown. These two methods completely depend on the training set, and as a consequence, a bad training set that does not reflect the true distribution well can ruin their performance. Another drawback of the two methods is the need to store all the training samples, which makes the estimation of probabilities slower during the test phase. In addition, KNN also suffers from outliers severely [13].

Most of reconstruction methods make assumptions about the underlying data structure, e.g., the clustering characteristics of training samples or their distribution in subspaces, then a model associated with a set of prototypes (codebooks) or subspaces is constructed to fit the data structure by minimizing a reconstruction error of training samples. The reconstruction error, reflecting the fit of the samples to the model, is evaluated. The smaller the reconstruction error is, the more likely the samples are generated by this model, namely, the more likely the samples come from the target class. Common reconstruction methods are K-means [15], self-organizing maps (SOM) [17], principal component analysis (PCA) [15], etc.. Both K-means and SOM require the number of prototypes to be provided. If the number of prototypes provided by users significantly differs from the real one, then wrong clusters may be produced by K-means. SOM and PCA impose strong assumptions about the data. For SOM, it is assumed that training samples are distributed in a low dimensional manifold (often two or three-dimensional). As for PCA, the features (variables) of training samples should be linearly related. Once training samples violate these assumptions, undesirable results would be obtained. Since outliers may bias the placement of the prototype vectors heavily, both K-means and SOM are sensitive to outliers [13]. Besides, PCA also suffers from outliers due to the variance measure involved in its algorithm, just like GMM.

Regarding boundary methods, neither the probability density nor the data structure is specifically modeled, only a closed decision boundary around training samples is optimized and is used to describe the target class. The classification is implemented through calculating the distances from new samples to the decision boundary. The well-known methods belonging to this category include support vector data description (SVDD) [18,19], one-class support vector machine (OCSVM) [20] and Mahalanobis ellipsoidal learning machine (MELM) [21]. SVDD is aimed at finding a minimum-volume hyper-sphere enclosing almost all the training sample. Instead of a hyper-sphere, a hyper-plane is used in OCSVM to separate training sample from the origin with the maximum margin. Although the goals of OCSVM and SVDD are seemingly different from each other, they are intrinsically equivalent when Gaussian kernels are used [19]. As for MELM, it is essentially a variant of SVDD, only the hyper-sphere is replaced by a hyper-ellipsoid, and correspondingly, the Mahalanobis distance is used as the measure instead of the Euclidean distance. Additionally, some researchers also explored the extension of a single OCSVM into an ensemble of OCSVMs by using clustering algorithms and fusion algorithms [22,23]. In the boundary methods, the kernel trick is commonly used, which implicitly maps samples from the input space $\mathcal{X}$ to a possibly high-dimensional feature space $\mathcal{F}$. In the feature space, a more flexible data description could be obtained, enabling boundary methods to deal with nonlinear and complex data. Boundary methods are relatively robust to outliers. By adjusting a regularization parameter (e.g., the parameter C in SVDD and the parameter ν in OCSVM) [18–20], outliers can be excluded from the resulting description. As the calculations in the test phase mainly depend on the number of support vectors (generally a small fraction of the total number of training samples), the classification of boundary methods is computationally simple and does not require significant memory (especially in contrast to KNN and PWD). Boundary methods make no prior assumption about the statistical distribution of training samples or their structure. This characteristic is greatly beneficial to the applications of boundary methods. In the real world, it might occur that comprehensive knowledge about the acquired data is completely unclear. In such case, boundary methods that directly estimate the boundary of the data might be preferable to density methods or reconstruction methods. For example, in mechanical fields understanding of machine operation principles is sometimes not straightforward. Having the routine monitoring data, we can hardly build the monitoring model based on comprehensive system physics and expert knowledge. Since only the boundary of the monitoring data is concentrated on, boundary methods are especially suitable for handling the monitoring data. Due to the above advantages, boundary methods have been becoming more and more popular. Most of boundary methods are formulated as a quadratic programming problem. Let l be the number of training samples. Since the standard algorithms for solving a quadratic

programming problem have the time complexity of order $O(l^3)$ [20], the training process of a boundary method is computationally expensive for large-scale problems. Meanwhile, a kernel matrix, with a number of elements equal to the square of the number of training samples, is used in the optimization routine of the quadratic programming problem [24], these boundary methods consequently have the space complexity of order $O(l^2)$. The computational complexity, however, may be reduced if a fast optimization algorithm, e.g., the well-known sequential minimal optimization (SMO) [20,24], is used.

From the perspective of data description, both SVDD and MELM employ a geometric model to describe the target class, i.e., one is the hyper-sphere and the other is the hyper-ellipsoid. The two geometric models motive us to explore another analogous model for possibly better data description. In this work, a novel boundary method called one-class classification based on the convex hull (OCCCH) is proposed, which describes the target class using a convex hull. OCCCH can be formulated as a nearest point problem whose goal is to find the nearest point to the origin from the convex hull. For inseparable cases, a reduced convex hull with a reduction factor is introduced instead of the convex hull. Furthermore, a generalized Gilbert algorithm is proposed to solve the nearest point problem. Compared with the existing algebraic algorithms for boundary methods, the generalized Gilbert algorithm is a geometric algorithm that has a more intuitive and explicit optimization procedure. It is computationally efficient and offers a new perspective for the solution of one-class classification problems. OCCCH inherits the common advantages of existing boundary methods, e.g., the utilization of the kernel trick, the robustness to outliers and the sparseness of the solution. By using two different biases, OCCCH has two different forms, i.e., OCCCH-1 and OCCCH-2. The relationships among OCCCH-1, OCCCH-2 and OCSVM are investigated theoretically. As we know, bearings are widely used as key components in various rotating machinery systems and usually constitute a large portion of machinery accidents. In order to prevent unexpected bearing failure, bearing faults should be detected as early as possible, especially in their early stages. As mentioned before, bearing condition monitoring is a typical one-class classification problem. In this paper, we use the proposed method to construct monitoring models for bearing fault detection.

The remaining part of the paper is organized as follows. Section 2 describes the motivation of our work. Section 3 introduces the basic theory of the proposed method. Section 4 presents our experiments and Section 5 concludes the paper.

2. Motivation

SVDD describes the target class of training samples with a hyper-sphere, while MELM uses a hyper-ellipsoid. Then, a question naturally arises that whether there exists another analogous geometric model which describes the target class of training samples appropriately. The answer is affirmative. A convex hull can be taken into consideration.

The convex hull of a sample set can be expressed as a convex combination of all the sample points where all coefficients are non-negative and sum to one. Consider a finite sample set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{l}$, the convex hull of the sample set $\boldsymbol{X}$ can be written as

$$\mathrm{CH}(\mathbf{X}) = \left\{ \sum_{i=1}^{l} \alpha_i \mathbf{x}_i \,\middle|\, \sum_{i=1}^{l} \alpha_i = 1,\ 0 \le \alpha_i \le 1 \right\} \tag{1}$$

where α_i is the combination coefficient of the ith sample point $\mathbf{x}_i$. The convex hull is the smallest convex set containing given finite samples, so it could approximate the class region tightly. A two-dimensional case of the convex hull is illustrated in Fig. 1.

Having defined the convex hull for the target class, we turn to the next question that how to generate a decision function associated with the convex hull for new samples. Firstly, let's review the decision processes of some existing boundary methods. SVDD uses the Euclidean distance as the measure to detect whether a new sample resembles the target class. If the sample lies inside the hyper-sphere, i.e., the Euclidean distance from this sample to the center of the hyper-sphere is smaller than the radius, then the new sample is considered to be from the target class. Instead of the Euclidean distance, MELM uses the Mahalanobis distance as the measure, but the decision process of MELM is similar to that of SVDD.

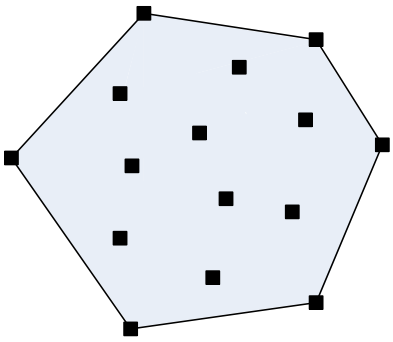

Fig. 1. A two-dimensional case of the convex hull.

Once the minimum-volume hyper-sphere or hyper-ellipsoid has been obtained, the center and the radius will be naturally specified. Therefore, the decision process of SVDD or MELM is easily implemented. The convex hull, however, cannot be described by only a center and a radius.

The direct computation of a convex hull in high-dimensional spaces is computationally intensive. In general, the computational cost of a d-dimensional ($d > 3$) convex hull over l samples is $O(l^{\lfloor d/2 \rfloor})$ [25]. This cost is prohibitive in time and memory. For a one-class classification task, actually we just need to check whether a sample lies inside the convex hull or not. To avoid the direct creation of the d-dimensional convex hull, one possible way is to calculate the minimum distance from a sample to the convex hull only using the convex hull formula (1). This is a quadratic programming problem [26]. In this way, the decision process for every new sample will correspond to an optimization problem. With respect to a large number of new samples, the whole decision processes are consequently rather computationally expensive because we have to solve the same number of quadratic programming problems as the new samples. Another alternative is to approximate the original d-dimensional convex hull with a set of bi-dimensional convex hulls obtained by projecting sample points onto random planes [27,28]. On those planes, computing a bi-dimensional convex hull and checking whether a sample belongs to the convex hull are both well-known problems in computational geometry. If a projection exists where the test sample is found outside of the bi-dimensional convex hull, then the sample lies outside the original d-dimensional convex hull. Obviously, the number of projections is a critical parameter for this approximate method. However, the choice of this parameter currently lacks theoretical support.

Let's focus on another popular boundary method. OCSVM treats the origin as a representative positive sample and employs a hyper-plane to separate training samples from the origin with the maximum margin. The hyper-plane is naturally the decision function. Similarly, the origin may also be introduced to generate a decision function associated with the convex hull of training samples. Now, we have two different classes, that is, the negative class is represented by the convex hull and the positive class is represented by only the origin. A nature idea is to find a hyper-plane that separates the convex hull and the origin as well as possible, and to regard this hyper-plane as the decision function. To this end, one can find the nearest point from the convex hull to the origin, and then a separating hyper-plane could be placed between the nearest point and the origin so that the convex hull is located on one side of the hyper-plane. If a sample lies on the side where the convex hull stays, it would be considered as a target sample; otherwise, it would be viewed as an anomaly. Inspired by this idea, a new one-class classification method is proposed in this paper. As the new method employs the convex hull to describe the target class of training samples, we call it one-class classification based on the convex hull (OCCCH).

3. One-class classification based on the convex hull

3.1. Linearly separable case

The basic goal of OCCCH is to find the nearest point to the origin from the convex hull of training samples, and then the separating hyper-plane is completely determined by the nearest point. The problem of finding such nearest point can be solved by computing a nearest point problem. The separating hyper-plane will be the one that passes through the nearest point and is perpendicular to the line segment joining the nearest point and the origin. Any point that lies on the separating hyper-plane satisfies $\langle \mathbf{w}, \mathbf{x} \rangle - b = 0$, where $\boldsymbol{w}$ and b are the normal and the bias, respectively. Any point on the side of the hyper-plane where the convex hull stays satisfies $\langle \mathbf{w}, \mathbf{x} \rangle - b > 0$, and any point on the other side satisfies $\langle \mathbf{w}, \mathbf{x} \rangle - b < 0$. Once the separating hyper-plane is determined, a new sample $\boldsymbol{x}$ will be classified based on the decision function $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle - b$. If $f(\mathbf{x}) \geq 0$, then $\boldsymbol{x}$ is accepted as a sample from the target class; otherwise, it is rejected as a sample from the non-target class.

Assume $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{l}$ are training samples whose labels are all negative. According to the formula (1), any point in the convex hull can be expressed as a linear combination of all the training samples. Thus, in the input space the Euclidean distance from any point in the convex hull to the origin can be written as

$$d = \left\{ \left\| \sum_{i=1}^{l} \alpha_i \mathbf{x}_i \right\| \,\middle|\, \sum_{i=1}^{l} \alpha_i = 1,\ 0 \leq \alpha_i \leq 1 \right\} \tag{2}$$

To find the nearest point, we only need to minimize the distance d. In the linearly separable case, i.e., the origin falls outside the convex hull of training samples, finding the nearest point from the convex hull to the origin can be formulated as the following nearest point problem

$$\begin{aligned} &\min_{\boldsymbol{\alpha} \in \mathbb{R}^l} \left\| \sum_{i=1}^{l} \alpha_i \mathbf{x}_i \right\| \\ &\text{s.t.} \sum_{i=1}^{l} \alpha_i = 1,\ 0 \leq \alpha_i \leq 1,\ i = 1, 2, \cdots, l. \end{aligned} \tag{3}$$

Given the optimal solution $\boldsymbol{\alpha}^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_l^*)^{\mathrm{T}}$ and let $\mathbf{x}^*$ denote the corresponding nearest point, the normal and the

bias of the separating hyper-plane can be computed using the following equations

$$\mathbf{w}^* = \mathbf{x}^* - \mathbf{0} = \mathbf{x}^* = \sum_{i=1}^{l} \alpha_i^* \mathbf{x}_i \tag{4}$$

$$b^* = \langle \mathbf{w}^*, \mathbf{x}^* \rangle = \|\mathbf{w}^*\|^2 = \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i^* \alpha_j^* \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{5}$$

From Eq. (4) we see that the normal of the separating hyper-plane is exactly the vector from the origin to the nearest point. The basic principle of OCCCH is illustrated in Fig. 2.

Now, the core issue is how to solve the nearest point problem (3). The Gilbert algorithm is a geometric algorithm developed for finding the minimum distance from a convex set to the origin [29]. Coincidently, the convex hull is the smallest convex set containing given training samples. Therefore, it is reasonable to incorporate the Gilbert algorithm into the OCCCH optimization problem.

The Gilbert algorithm for OCCCH is given as follows:

Step 1: Set the vector $\boldsymbol{w}$ to any point $\mathbf{x} \in \mathbf{X}$, e.g., set $\mathbf{w} = \sum_{i \in \mathbf{I}} \alpha_i \mathbf{x}_i$ ($\mathbf{I} = \{1, 2, \cdots, l\}$) where $\alpha_1 = 1$ and $\alpha_i = 0$ for $i > 1$.

Step 2: Find the point $\mathbf{x}_t \in \mathbf{X}$ ($t \in \mathbf{I}$) with the minimum projection onto the direction of $\boldsymbol{w}$ as $\mathbf{x}_t = \arg\min_{\mathbf{x}_i \in \mathbf{X}} p_i$ where $p_i = \langle \mathbf{x}_i, \mathbf{w} \rangle / \|\mathbf{w}\|$ ($i \in \mathbf{I}$). The point $\mathbf{x}_t$ can be written as $\mathbf{x}_t = \sum_{i \in \mathbf{I}} \beta_i \mathbf{x}_i$, $\beta_i = \delta_{it}$ where $\delta_{it} = \begin{cases} 1, & i = t \\ 0, & i \neq t \end{cases}$. If $\|\mathbf{w}\| - p_{\min} \leq \varepsilon \|\mathbf{w}\|$ where $p_{\min} = \min_{i \in \mathbf{I}} p_i$, then $\mathbf{w}^* = \mathbf{w}$, $b^* = \|\mathbf{w}\|^2$ and stop the algorithm; otherwise, go to Step 3.

Step 3: Compute the coefficients α_i^{new} of $\mathbf{w}^{\text{new}}$, $\alpha_i^{new} = (1 - q)\alpha_i + q\beta_i$ where $q = \min(1, \langle \mathbf{w}, \mathbf{w} - \mathbf{x}_t \rangle / \|\mathbf{w} - \mathbf{x}_t\|^2)$. Set $\mathbf{w} \leftarrow \mathbf{w}^{\text{new}}$ and go to Step 2.

The elements involved in the above steps of the geometric algorithm are illustrated in Fig. 3. The idea behind the algorithm is very simple and intuitive. At each iteration step, the point $\boldsymbol{w}$, representing the nearest point (until current step), is known through the coefficients α_i ($i \in \mathbf{I}$), i.e., $\mathbf{w} = \sum_{i \in \mathbf{I}} \alpha_i \mathbf{x}_i$. This form coincides with Eq. (4).

Before updating $\boldsymbol{w}$, the minimum projection point $\mathbf{x}_t$ need to be located. The Gilbert algorithm searches $\mathbf{x}_t$ through the calculations of projections of all the sample points onto the direction of $\boldsymbol{w}$. Actually, $\mathbf{x}_t$ could always be one of the extreme points (or called vertexes) of the convex hull (see Fig. 3). Regarding a convex hull generated by a set of points, only a subset of these points constitutes the set of the extreme points, which in turn is the minimal representation of the convex hull [30]. Therefore, not every point but only the extreme points actually need to be examined in the Step 2 if we already have the exact locations of all the extreme points beforehand.

In the Step 3, $\mathbf{w}^{\text{new}}$ is actually the point with the minimum norm (nearest to the origin) on the line segment $[\mathbf{w}, \mathbf{x}_t]$. The update expression restricts $\mathbf{w}^{\text{new}}$ to be on $[\mathbf{w}, \mathbf{x}_t]$, thus $\mathbf{w}^{\text{new}}$ necessarily belongs to the convex hull. At the same time, the update can ensure $\|\mathbf{w}^{\text{new}}\| < \|\mathbf{w}\|$ (see Fig. 3), which means that the point $\boldsymbol{w}$ will be getting closer to the origin as the iteration goes on. Geometrically, the algorithm iterates until $\boldsymbol{w}$ becomes an ε-optimal nearest point. The precision parameter

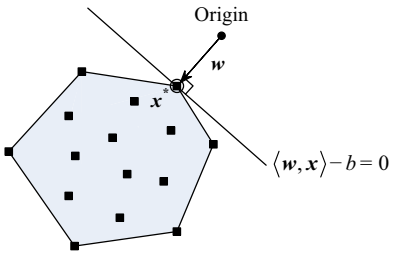

Fig. 2. Illustration of the basic principle of OCCCH.

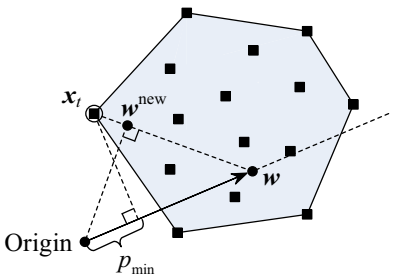

Fig. 3. Basic idea involved in the Gilbert algorithm for OCCCH.

ε could be set to a small positive value, e.g. $\varepsilon = 10^{-3}$.

For the calculations of p_i ($i \in \mathbf{I}$) and q, the quantities $\langle \mathbf{w}, \mathbf{x}_i \rangle$ ($i \in \mathbf{I}$), $\langle \mathbf{w}, \mathbf{w} \rangle$ and $\langle \mathbf{w}, \mathbf{x}_t \rangle$ need to be calculated. However, the calculations do not involve $\boldsymbol{w}$ directly but only through inner products between pairs of training samples since $\boldsymbol{w}$ is always expressed as a linear combination of training samples associated with the coefficients α_i ($i \in \mathbf{I}$). Now that the implementation of the geometric algorithm only concerns inner products, the well-known kernel trick is allowed to use in the Gilbert algorithm. Besides, a caching scheme can be adopted for $\langle \mathbf{w}, \mathbf{x}_i \rangle$ ($i \in \mathbf{I}$) and $\langle \mathbf{w}, \mathbf{w} \rangle$, with only $O(l)$ storage requirements. Common choices for the kernel function are Gaussian, polynomial and sigmoid kernels, etc. In this work, only the Gaussian kernel is considered. The Gaussian kernel is given in the following form

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right) \tag{6}$$

where σ is the Gaussian width parameter.

3.2. Linearly inseparable case

In the linearly inseparable case, i.e., the origin lies inside the convex hull of training samples, the Gilbert algorithm fails to solve the nearest point problem (3) because the minimum distance from the convex hull to the origin is always zero. Although the kernel trick might transform an inseparable problem in the input space into a separable problem in the feature space, the linear separability cannot be necessarily guaranteed in the feature space. Moreover, there may still exist outliers in the feature space. To cope with the inseparable case, one can reduce the convex hull so that the origin is located outside it, and then the nearest point problem is still solvable. Fortunately, a variant of the convex hull, the reduced convex hull, is able to accomplish this work exactly. This is illustrated in Fig. 4.

The reduced convex hull of the sample set $\boldsymbol{X}$ can be written as

$$\mathrm{RCH}(\mathbf{X}, \mu) = \left\{ \sum_{i=1}^{l} \alpha_i \mathbf{x}_i \,\middle|\, \sum_{i=1}^{l} \alpha_i = 1,\ 0 \le \alpha_i \le \mu \right\} \tag{7}$$

where α_i ($i \in \mathbf{I}$) are the combination coefficients and $\mu \in [1/l, 1)$ is the reduction factor. The difference between the reduced convex hull and the convex hull is that the former introduces an upper bound μ on every coefficient. Clearly, the convex hull is an extreme case of the reduced convex hull in the case of $\mu = 1$. Note that the reduced convex hull itself is still a convex hull.

The effect of the reduction factor μ on the reduced convex hull is illustrated in Fig. 5. As μ decreases, the convex hull shrinks

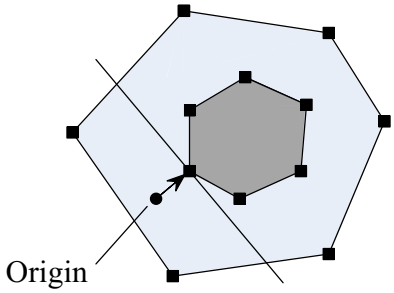

Fig. 4. A linearly inseparable case of OCCCH.

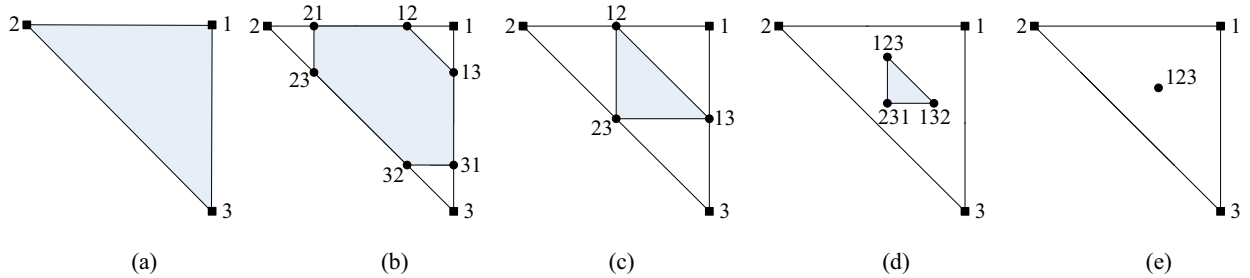

Fig. 5. Reduced convex hulls (generated by 3 points) with respect to different reduction factors. (a) $\mu = 1$, (b) $\mu = 3/4$, (c) $\mu = 1/2$, (d) $\mu = 5/12$, and (e) $\mu = 1/3$. Each extreme point (represented by a circle) in the reduced convex hulls is labeled so as to present the original extreme points (represented by squares) from which it has been constructed; the order of numbers is arranged according to the non-ascending order of the combination coefficients of the corresponding original extreme points, e.g., the extreme point (12) results from the original extreme points (1) and (2), and the combination coefficient of the point (2) is smaller than or equal to that of the point (1).

non-uniformly towards the centroid of the sample set $\boldsymbol{X}$. Any point lying in the reduced convex hull is certainly located inside the original convex hull. Particularly, $\alpha_i = 1/l$ holds for all i when $\mu = 1/l$, and then the reduced convex hull degenerates to one single point, the centroid (see Fig. 5(e)). Let's focus on the range of μ. On the one hand, the lower bound of μ cannot be less than $1/l$, otherwise the sum of the coefficients will conflict with the coefficient constraint $\sum_{i=1}^{l} \alpha_i = 1$, which means that the reduced convex hull will be empty. On the other hand, a choice of μ larger than 1 always exactly yield the convex hull, which indicates that $\mu > 1$ is always equivalent to $\mu = 1$ in terms of the shape of the resulting geometric model.

The purpose of reducing the convex hull is to make the origin fall outside the hull. As long as the reduction factor μ is appropriate, the origin linearly inseparable case will become a linearly inseparable one. In the linearly inseparable case, finding the nearest point from the reduced convex hull to the origin can be formulated as the following nearest point problem

$$\begin{aligned} &\min_{\boldsymbol{\alpha}\in\mathbb{R}^l} \left\| \sum_{i=1}^{l} \alpha_i \mathbf{x}_i \right\| \\ &\text{s.t.} \sum_{i=1}^{l} \alpha_i = 1,\ 0 \le \alpha_i \le \mu,\ i = 1, 2, \cdots, l. \end{aligned} \tag{8}$$

In this case, the normal and the bias of the separating hyper-plane can be still calculated using (Eqs. (4) and 5). When there exist outliers in the training set, the reduction factor μ can play a role in restraining the effect of outliers on the separating hyper-plane. Although the reduced convex hull is used instead of the convex hull in the linearly inseparable case, we still call the proposed method OCCCH since the reduce convex hull itself is still a convex hull.

With an appropriate reduction factor μ, the original convex hull can be reduced so that the inseparable case becomes separable. Once separable, the geometric algorithm developed for the separable case seems to be readily applied to the inseparable case. However, it turns out to be not such a straightforward task. Recall that one of the key points of the Gilbert algorithm is to find the extreme point of the convex hull with the minimum projection onto a specific direction. Therefore, it seems to be absolutely necessary to provide position information about the extreme points of the reduced convex hull if we want to achieve the direct application of the Gilbert algorithm to the inseparable case. It is known that every extreme point of the reduced convex hull certainly lies in the original convex hull. Since the set of original extreme points is the minimal representation of the original convex hull [30], every extreme point of the reduced convex hull is a linear combination of the original extreme points. However, not any combination of the original extreme points results in the extreme points of the reduced convex hull. This is illustrated in Figs. 5(b) and (c). Moreover, the corresponding combination coefficients are still unclear. These facts make the direct application of the geometric algorithm impractical.

Mavroforakis and Theodoridis [31] provided a significant theorem that the calculation of the minimum projection of the extreme points of a reduced convex hull onto a specific direction does not need the direct formation of all the possible extreme points but only the calculation of the projections of the original points, and then the minimum projection is the summation of the $\lceil 1/\mu \rceil$ ($\lceil\cdot\rceil$ denotes the ceiling function, here $\mu = 1/(\nu l)$) smaller ones among these projections, with each being multiplied by a specific coefficient belonging to the set $\{\mu, 1 - \lfloor 1/\mu \rfloor \mu\}$ ($\lfloor\cdot\rfloor$ denotes the floor function). More precisely, if $1/\mu = \lceil 1/\mu \rceil$, μ is assigned to all the coefficients of the $\lceil 1/\mu \rceil$ smaller projections; otherwise, μ is assigned to each of the former $\lceil 1/\mu \rceil - 1$ coefficients and $1 - \lfloor 1/\mu \rfloor \mu$ is assigned to the last coefficient. Also, the combination coefficients of the $\lceil 1/\mu \rceil$ smaller projections that constitute the minimum projection are completely the same as those of the $\lceil 1/\mu \rceil$ corresponding original points that constitute the minimum projection extreme point, so it is easy to derive the minimum projection extreme point of the reduced convex hull. Now that the minimum projection and corresponding extreme point, required in the Gilbert algorithm, are both received, a generalized Gilbert algorithm for OCCCH is proposed to solve the nearest point problem associated with the reduced convex hull.

The generalized Gilbert algorithm for OCCCH is given as follows:

Step 1: Ensure that $\mu \ge 1/l$. Set the vector $\boldsymbol{w}$ to the centroid of the convex hull, i.e., set $\mathbf{w} = \sum_{i\in\mathbf{I}} \alpha_i \mathbf{x}_i$ where $\alpha_i = 1/l$ for all i.

Step 2:

(1)Find the point $\mathbf{x}_t \in \mathrm{RCH}(\mathbf{X}, \mu)$ with the minimum projection onto the direction of $\boldsymbol{w}$.

(a)Compute the projections of all the points $\mathbf{x}_i \in \mathbf{X}$ ($i \in \mathbf{I}$) onto the direction of $\boldsymbol{w}$, i.e., $p_i = \langle \mathbf{x}_i, \mathbf{w} \rangle / \|\mathbf{w}\|$ ($i \in \mathbf{I}$).

(b)Sort the projections in ascending order and select the $\lceil 1/\mu \rceil$ smaller projections p_i ($i \in \tilde{\mathbf{I}}$), where $\tilde{\mathbf{I}}$ is the set of the first $\lceil 1/\mu \rceil$ indices of the original points with sorted projections.

(c)Compute the minimum projection, i.e., $p_{\min} = \mu \sum_{i\in\tilde{\mathbf{I}}(1:\mathrm{end}-1)} p_i + (1 - \lfloor 1/\mu \rfloor \mu) p_{\tilde{\mathbf{I}}(\mathrm{end})}$. The point $\mathbf{x}_t$ with the minimum projection onto the direction of $\boldsymbol{w}$ can be written as $\mathbf{x}_t = \sum_{i\in\mathbf{I}} \beta_i \mathbf{x}_i$, $\beta_i = \mu \sum_{j\in\tilde{\mathbf{I}}(1:\mathrm{end}-1)} \delta_{ij} + (1 - \lfloor 1/\mu \rfloor \mu) \sum_{j\in\tilde{\mathbf{I}}(\mathrm{end})} \delta_{ij}$ ($i \in \mathbf{I}$) where $\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \ne j \end{cases}$.

(2)If $\|\mathbf{w}\| - p_{\min} \le \varepsilon \|\mathbf{w}\|$, then $\mathbf{w}^* = \mathbf{w}$, $b^* = \|\mathbf{w}\|^2$ and stop the algorithm; otherwise, go to Step 3.

Step 3: Compute the coefficients α_i^{new} of $\mathbf{w}^{\mathrm{new}}$, $\alpha_i^{new} = (1 - q)\alpha_i + q\beta_i$ where $q = \min(1, \langle \mathbf{w}, \mathbf{w} - \mathbf{x}_t \rangle / \|\mathbf{w} - \mathbf{x}_t\|^2)$. Set $\mathbf{w} \leftarrow \mathbf{w}^{\mathrm{new}}$ and go to Step 2.

In the Step 1, the initial $\boldsymbol{w}$ is set to the centroid of the convex hull that belongs to the reduced convex hull. By this initialization, the algorithm can ensure that the point $\boldsymbol{w}$ at each iteration step is always in the reduced convex hull.

The main difference between the generalized Gilbert algorithm and the original one lies in the Step 2. The calculation of

the minimum projection here is more complicated. It needs the sort to find the $\lceil 1/\mu \rceil$ smaller projections and the weighted average calculation of these selected projections. The minimum projection point $\mathbf{x}_t$ is no longer a single original point but a linear combination of $\lceil 1/\mu \rceil$ original points. Nevertheless, the search ideas for the minimum projection point involved in the two algorithms have one thing in common, i.e., what matters is not the absolute position of each extreme point but their projections onto a specific direction.

The original geometric algorithm can be obviously regarded as a special case of the generalized one with $\mu = 1$. Therefore, the generalized Gilbert algorithm would be more widely used in practice to solve one-class classification problems. Note that all vectors involved in the calculations are not be used directly (as done in the original Gilbert algorithm) but only through inner products. As a result, the kernel trick is still applicable.

3.3. Relationship with OCSVM

OCSVM is firstly proposed by Schölkopf et al. [20] to solve one-class classification problems. The separating hyper-plane of OCSVM is parameterized by a normal $\bar{\mathbf{w}}$ and a bias ρ: $\langle \bar{\mathbf{w}}, \Phi(\mathbf{x}) \rangle - \rho = 0$. The maximum margin problem for OCSVM has the following formula [20]

$$\begin{aligned} &\min_{\bar{\mathbf{w}} \in \mathcal{F}, \rho \in \mathbb{R}, \xi \in \mathbb{R}^l} \frac{1}{2}\|\bar{\mathbf{w}}\|^2 - \rho + \frac{1}{\nu l}\sum_{i=1}^{l} \xi_i \\ &\text{s.t.} \langle \bar{\mathbf{w}}, \Phi(\mathbf{x}_i) \rangle \geq \rho - \xi_i,\ \xi_i \geq 0,\ i = 1, 2, \cdots, l. \end{aligned} \tag{9}$$

where $\nu \in (0, 1]$ is a user-specified parameter and ξ_i ($i = 1, 2, \cdots, l$) are the slack variables. It is known that ν acts as an upper bound on the fraction of rejected training samples and a lower bound on the fraction of support vectors [20].

The dual problem of the maximum margin problem (9) is formulated as

$$\begin{aligned} &\max_{\bar{\boldsymbol{\alpha}} \in \mathbb{R}^l} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \bar{\alpha}_i \bar{\alpha}_j k(\mathbf{x}_i, \mathbf{x}_j) \\ &\text{s.t.} \sum_{i=1}^{l} \bar{\alpha}_i = 1,\ 0 \leq \bar{\alpha}_i \leq \frac{1}{\nu l},\ i = 1, 2, \cdots, l. \end{aligned} \tag{10}$$

where $\bar{\alpha}_i$ ($i = 1, 2, \cdots, l$) are the Lagrange multipliers. This is a quadratic programming problem that can be solved by using standard algebraic optimization algorithms. However, these algebraic algorithms usually have high computational complexity. Schölkopf et al. [20] employed a sequential minimal optimization (SMO) algorithm to solve the optimization problem described above. The SMO algorithm breaks this problem into a series of smallest possible sub-problems. Each QP sub-problem can be solved analytically in an efficient way since only two variables are involved. In this way, the SMO algorithm is able to reduce the computational complexity. Given the optimal solution $\bar{\boldsymbol{\alpha}}^* = (\bar{\alpha}_1^*, \bar{\alpha}_2^*, \cdots, \bar{\alpha}_l^*)^{\mathrm{T}}$, the normal and the bias of the separating hyper-plane can be computed using the following equations

$$\bar{\mathbf{w}}^* = \sum_{i=1}^{l} \bar{\alpha}_i^* \Phi(\mathbf{x}_i) \tag{11}$$

$$\rho^* = \sum_{j=1}^{l} \bar{\alpha}_j^* k(\mathbf{x}_{\mathrm{nbsv}}, \mathbf{x}_j) \tag{12}$$

where $\mathbf{x}_{\mathrm{nbsv}}$ is any non-bound support vector, i.e., a sample whose Lagrange multiplier α_i^* satisfies $0 < \alpha_i^* < 1/(\nu l)$.

Squaring the objective function of the nearest point problem (8) and replacing the inner product $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with the kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$, we can obtain an equivalent formula of the kernelized version of the optimization problem (8), i.e.,

$$\begin{aligned} &\min_{\boldsymbol{\alpha} \in \mathbb{R}^l} \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ &\text{s.t.} \sum_{i=1}^{l} \alpha_i = 1,\ 0 \leq \alpha_i \leq \mu,\ i = 1, 2, \cdots, l. \end{aligned} \tag{13}$$

Correspondingly, (Eqs. (4) and 5) for computing the normal and the bias can be rewritten as

$$\mathbf{w}^* = \sum_{i=1}^{l} \alpha_i^* \Phi(\mathbf{x}_i) \tag{14}$$

$$b^* = \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i^* \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \tag{15}$$

From the comparison of (Eqs. (10)–12) and (Eqs. (13)–15), it is clearly seen that the nearest point problem (13) with the parameter setting $\mu = 1/(\nu l)$ is mathematically equivalent to the dual problem (10). In the case of $\mu = 1/(\nu l)$, OCCCH and OCSVM will generate separating hyper-planes with the same normal, but the positions (perpendicular distances from the origin) of these hyper-planes may be different. If the separating hyper-plane $\left\langle \bar{\mathbf{w}}^*, \Phi(\mathbf{x}_i) \right\rangle - \rho^* = 0$ of OCSVM has been provided, the bias of the separating hyper-planes of OCCCH can be given as follows

$$b^* = \rho^* - \mu \sum_{i=1}^{l} \xi_i^* \tag{16}$$

where ξ_i^* $(i = 1, 2, \cdots, l)$ are the slack variables associated with the separating hyper-plane $\left\langle \bar{\mathbf{w}}^*, \Phi(\mathbf{x}_i) \right\rangle - \rho^* = 0$, i.e., $\xi_i^* = -(\left\langle \bar{\mathbf{w}}^*, \Phi(\mathbf{x}_i) \right\rangle - \rho^*)$. The proof is given in Appendix A. As all ξ_i^* satisfy the constraints $\xi_i^* \geq 0$ $(i = 1, 2, \cdots, l)$, $b^* \leq \rho^*$ always holds. If there exist no rejected training samples, i.e., $\xi_i^* = 0$ hold for all i, then $b^* = \rho^*$. In this case, OCCCH with $\mu = 1/(\nu l)$ theoretically leads to the completely same separating hyper-plane as OCSVM. If there exists at least one rejected training sample, i.e., $\xi_i^* > 0$ holds for at least one i, then $b^* < \rho^*$. This implies that the separating hyper-plane resulting from OCCCH is more closer to the origin. Correspondingly, in the input space the decision boundary generated by OCCCH is looser than the one generated by OCSVM. If the hyper-plane generated by OCSVM is shifted towards the origin by a perpendicular distance $(\mu \sum_{i=1}^{l} \xi_i^*)/\|\bar{\mathbf{w}}^*\|$, it will coincide with the hyper-plane generated by OCCCH. Conversely, if the separating hyper-plane $\left\langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \right\rangle - b^* = 0$ of OCCCH has been provided, it can be shifted to the position of the hyper-plane resulting from OCSVM by using the bias given as follows

$$b_2^* = b^* + \frac{\mu}{1 - l_2\mu} \sum_{i \in \mathbf{I}_2} \eta_i^* \left(1 - l_2\mu \neq 0\right) \tag{17}$$

where $\mathbf{I}_2$ is the set of the indices of sample points whose combination coefficients are equal to μ and l_2 is the size of the set $\mathbf{I}_2$. η_i^* $(i \in \mathbf{I}_2)$ are calculated as follows

$$\eta_i^* = -\left(\left\langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \right\rangle - b^*\right) \tag{18}$$

The proof is given in Appendix B. When $1 - l_2\mu = 0$, all the coefficients α_i^* $(i = 1, 2, \cdots, l)$ take the bound value 0 or μ. Nevertheless, this case seldom occurs.

As a result of the above analysis, we specially offer two different biases for OCCCH in this paper, i.e., the original bias b^* and the new bias b_2^*. OCCCH using the bias b^* is given the name OCCCH-1 and the other OCCCH-2. Once the nearest point problem is solved, we compute the biases b^* and b_2^* using Eqs. (15) and (17) separately, and then obtain OCCCH-1 and OCCCH-2. Theoretically, OCCCH-2 and OCSVM yield the same separating hyper-planes as well as the corresponding decision boundaries. However, OCCCH-2 constructs the hyper-plane through the geometric algorithm (i.e., the generalized Gilbert algorithm), while OCSVM usually through the algebraic algorithm (e.g., the SMO algorithm). Consequently, the two hyper-planes may show a subtle difference. As for OCCCH-1 and OCCCH-2, it should be emphasized that there is no a priori evidence that which one is better. The only definite thing we know is that, compared with OCCCH-2, OCCCH-1 is able to generate a looser decision boundary. As it will soon be seen in our experiments, the looser decision boundary could contribute to reducing the false positive rate in some cases.

4. Experiments

OCCCH-1 and OCCCH-2 were firstly tested through the experiments conducted on benchmark datasets. Also, we compared the performance of OCCCH-1, OCCCH-2 and OCSVM quantitatively with respect to the classification accuracy and the

Table 1
Details of benchmark datasets.

Dataset	Dimension	#Negative	#Positive
Biomed	5	127	67
Heart	13	150	120
Ionosphere	34	225	126
Breast	9	458	241
Diabetes	8	500	268
Cardiotocography	21	1655	471
Spambase	57	2788	1813
Svmguide1	4	2000 (2000)	1089 (2000)
Shuttle	9	34108 (11478)	9392 (3022)
Cod_rna	8	39690 (181078/ 104942)	19845 (90539/ 52471)

Notes: For Svmguide1, Shuttle or Cod_rna, there is more than one number in the column "#Negative" or "#Positive". The first number represents the number of training samples, and the remaining number(s) in the parentheses represents the number of test (validation) samples.

computational efficiency. Next, to verify the practicability, OCCCH was used to construct monitoring models for bearing fault detection. Meanwhile, the performance comparison was conducted between OCCCH-1 and OCSVM. All the experiments were performed on a machine with 2.8 GHz Intel Quad-Core CPU and 4 GB RAM. Training samples in each training set were normalized to zero mean and unit variance before they were used for training. In the test stage, test samples were also normalized using the mean and the variance of the corresponding training set. Since the nearest point problem (13) in OCCCH and the dual problem (10) in OCSVM are mathematically equivalent in the case of $\mu = 1/(\nu l)$, it is therefore reasonable to set the same or equivalent parameter values for OCCCH and OCSVM when comparing their performance with respect to the same dataset. On the one hand, the kernel parameters σ of OCCCH and OCSVM should take the same value; on the other hand, the reduction factor μ of OCCCH and the regularization parameter ν of OCSVM should satisfy the case $\mu = 1/(\nu l)$. No matter in the experiments conducted on the benchmark datasets or in the experiments conducted on the bearing dataset, the parameter values of OCCCH and OCSVM were always determined based on these rules. Under such parameter settings, we believe the comparisons are fair and meaningful. OCCCH constructed classification models by using the generalized Gilbert algorithm, while OCSVM by using the well-known SMO algorithm.

4.1. Benchmark datasets

The benchmark datasets include 9 datasets from UCI machine learning repository [32] and Svmguide1 from LIBSVM data collection [33]. Details of these benchmark datasets are listed in Table 1. These datasets vary greatly in the number of samples as well as the dimensions. Cardiotocography is originally used for three-class classification (3 fetal states: normal, suspect and pathologic), so we just regarded the normal state as the negative class and combined other two states into the positive class. Regarding Shuttle, we also only chose the class label "1" as the negative class and other labels as the positive class. For each dataset, the negative class contains the largest number of samples. Among these benchmark datasets, only Svmguide1, Shuttle and Cod_rna have designated a training set and a test set for users. Particularly, Cod_rna have also designated an extra validation set. Here, this validation set was incorporated into the test set. Regarding these three datasets, only the negative samples in the designated training set were used to construct a new training set. All the positive samples in the designated training set and all the samples in the designated test set were combined into a new test set. As for other datasets, 2/3 of the negative samples were used to form a training set, and the remaining 1/3 negative and all the positive samples composed a test set. Therefore, training sets contain only negative samples, while test sets contain both positive and negative samples.

To quantitatively evaluate the performance of different one-class classification methods, we used five evaluation indicators: the g-mean, the true positive rate (TPR) and the true negative rate (TNR) for the classification accuracy evaluation; the training time and the number of kernel evaluations for the computational efficiency evaluation. Among these indicators, the g-mean metric is defined as the geometric mean of TPR and TNR [34], i.e., $g\text{–mean} = \sqrt{\text{TPR} \times \text{TNR}}$. This metric takes into account the classification results on both the positive and the negative classes, thus giving an overall classification accuracy. Sometimes, we are highly interested in the effective detection ability for one-class (the positive or negative class). For such problems, TPR and TNR are often adopted. On the other hand, we also concern the computational efficiency of different methods. For this purpose, the training time is the most commonly used measurement. Additionally, both OCCCH and OCSVM involve a large number of kernel evaluations (between two training samples) performed during training. Therefore, the number of kernel evaluations can be taken as another indicator for the computational efficiency evaluation.

Regarding each dataset, both the precision parameter of the generalized Gilbert algorithm and the Karush-Kuhn-Tucker (KKT) tolerance parameter of the SMO algorithm were set to 10^{-3}. The parameter values of ν (for OCSVM) and σ for the benchmark datasets are given in Table 2. At the same time, the reduction factor μ of OCCCH can be obtained according to the conversion formula $\mu = 1/(\nu l)$. Our parameter values were chosen to be the ones that made these methods get as high accuracies as possible.

The experimental results are reported in Table 3. Note that the training time and the number of kernel evaluations of

Table 2
Parameter values of ν and σ for the benchmark datasets.

Dataset	ν	σ
Biomed	0.1	7.949
Heart	0.25	4.661
Ionosphere	0.1	2.690
Breast	0.07	20.155
Diabetes	0.3	4.312
Cardiotocography	0.3	25.441
Spambase	0.3	36.557
Svmguide1	0.1	8.097
Shuttle	0.1	0.497
Cod_rna	0.2	0.666

Notes: The regularization parameter ν converts to the reduction factor μ by using $\mu = 1/(\nu l)$, where l is the number of training samples.

Table 3
Results of different methods on the benchmark datasets.

Dataset	Method	G-mean (%)	TPR (%)	TNR (%)	Training time (s)	Kernel Eval.
Biomed	OCCCH-1	85.14	76.12	95.24	0.106	58311
	OCCCH-2	83.12	80.60	85.71	0.106	58311
	OCSVM	83.12	80.60	85.71	0.907	86173
Heart	OCCCH-1	65.52	46.67	92.00	0.148	95000
	OCCCH-2	70.68	67.50	74.00	0.148	95000
	OCSVM	69.58	69.17	70.00	6.170	1046475
Ionosphere	OCCCH-1	81.81	98.41	68.00	19.473	17952525
	OCCCH-2	81.81	98.41	68.00	19.473	17952525
	OCSVM	81.81	98.41	68.00	149.472	24093073
Breast	OCCCH-1	91.58	86.72	96.71	0.214	375933
	OCCCH-2	94.63	95.85	93.42	0.214	375933
	OCSVM	95.29	95.85	94.74	30643.239	17403406603
Diabetes	OCCCH-1	51.52	30.60	86.75	0.251	476527
	OCCCH-2	65.52	63.06	68.07	0.251	476527
	OCSVM	65.02	61.57	68.67	7.891	4989719
Cardiotocography	OCCCH-1	70.23	54.56	90.38	0.646	2547525
	OCCCH-2	76.30	90.87	64.07	0.646	2547525
	OCSVM	76.54	84.50	69.33	98.788	276311460
Spambase	OCCCH-1	64.13	44.73	91.93	3.441	7222569
	OCCCH-2	72.36	77.33	67.71	3.441	7222569
	OCSVM	72.31	75.90	68.89	457.587	797607266
Svmguide1	OCCCH-1	68.85	48.59	97.55	3.473	28240000
	OCCCH-2	86.14	82.16	90.30	3.473	28240000
	OCSVM	86.88	84.91	88.90	117.129	639492107
Shuttle	OCCCH-1	94.26	100	88.85	3027.410	10238061928
	OCCCH-2	94.35	100	89.02	3027.410	10238061928
	OCSVM	94.63	100	89.55	24826.144	152205405435
Cod_rna	OCCCH-1	71.26	57.93	87.67	2802.378	12602368800
	OCCCH-2	76.40	76.11	76.69	2802.378	12602368800
	OCSVM	76.34	80.86	72.06	86757.215	559266295227

OCCCH-1 and OCCCH-2 are always the same, due to the same geometric algorithm that they employed. From the results shown in Table 3, it is apparent that OCCCH-2 always gives comparable g-means to OCSVM, and the g-means of the two methods are both higher than that of OCCCH-1 with respect to most of the datasets. To statistically compare the three methods in terms of the classification accuracy, we perform the Friedman test with the Nemenyi post-hoc test [35]. Concerning the g-means, the average ranks of OCCCH-1, OCCCH-2 and OCSVM over the ten datasets are $R_1 = 2.70$, $R_2 = 1.65$ and $R_3 = 1.65$, respectively. The Friedman test checks whether the measured average ranks are significantly different from the mean rank $\bar{R} = 2$ expected under the null-hypothesis which states that all the methods perform equivalently. With three methods and ten datasets, the Friedman statistics are: $\chi_F^2 = 12 \times 10/(3 \times 4) \times (2.70^2 + 1.65^2 + 1.65^2 - 3 \times 4^2/4) = 7.35$, $F_F = 9 \times \chi_F^2/(10 \times 2 - \chi_F^2) = 5.23$. F_F is distributed according to the F distribution with $3 - 1 = 2$ and $(3 - 1) \times (10 - 1) = 18$ degrees of freedom. For a significance level of $\alpha = 0.05$, the critical value of $F(2, 18)$ is 3.55, so we reject the null-hypothesis. Next, we proceed with the Nemenyi post-hoc test for pairwise comparisons. The performance of two methods is significantly different if the corresponding average ranks differ by at least the critical difference: $CD = 2.343 \times \sqrt{3 \times 4/(6 \times 10)} = 1.048$. The differences of the average ranks are: $R_1 - R_2 = 1.05 > CD$, $R_1 - R_3 = 1.05 > CD$, $R_2 - R_3 = 0 < CD$. In terms of the classification accuracy, we can therefore conclude that OCCCH-2 and OCSVM perform equally well and both of them perform significantly better than OCCCH-1. Regarding OCCCH-2 and OCSVM, the comparable g-means are actually to be expected since the two methods are theoretically equivalent under our parameter settings. Except for Biomed, Ionosphere and Shuttle, OCCCH-1 receives high TNRs as well as low TPRs with respect to the rest datasets, indicating that the resulting decision boundaries of OCCCH-1 tend to be loose. Compared with OCCCH-2 and OCSVM, as mentioned previously, OCCCH-1 is able to generate looser decision boundaries in the input space. For most of the benchmark datasets here, the looser decision boundaries do not improve the both test accuracies on the positive samples and the negative samples, but decrease the g-means.

Concerning the computational efficiency, the proposed generalized Gilbert algorithm brings very encouraging results. As shown in Table 3, OCCCH using the proposed geometric algorithm achieves less training time and fewer number of kernel evaluations overall the datasets, performing significantly better than OCSVM using the SMO algorithm. So, there is no need to provide the statistical test for the comparison on the computational efficiency. Note that OCSVM consumes too much training time particularly with respect to Breast. In the experiment with SMO applied to this dataset, we found it to reach the maximum number of iterations (i.e., 10000) before the KKT conditions were fulfilled within a tolerance, thus it seems not to converge. This may be due to the natural drawback of the SMO algorithm that it cannot ensure convergence in some unusual situations [20]. Unfortunately, Breast seems to be such a case. The enhanced computational efficiency of the new geometric algorithm against its most popular algebraic competitor can be explained by the fact that, although the SMO

algorithm makes a clever utilization of the caching scheme, it cannot avoid the repetitive searches for the best pair of Lagrange multipliers to be optimized in the next iteration step through ambiguous and sometimes inefficient heuristics. On the contrary, the generalized Gilbert algorithm is a straightforward optimization algorithm, with an intuitive and explicit optimization target at each iteration step.

In summary, it can be concluded that OCCCH is computationally more efficient than OCSVM, and moreover, OCCCH-2 can receive comparable accuracies to OCSVM.

4.2. Bearing fault detection

In order to verify the practicability of the proposed method, we used OCCCH to construct monitoring models for bearing fault detection. Also, OCSVM was implemented for comparison. A bearing run-to-failure test under a constant load condition was performed on a specially designed test rig as shown in Fig. 6 [36]. The bearing test rig hosts four Rexnord ZA-2115 double row bearings on one shaft. The shaft is driven by an AC motor and coupled by rub belts. A radial load of 6000 lbs is added to the shaft and the bearings by a spring mechanism. The test bearings have 16 rollers in each row, a pitch diameter of 2.815 in., a roller diameter of 0.331 in. and a tapered contact angle of 15.17°. All the bearings are force lubricated. An oil circulation system regulates the flow and the temperature of the lubricant. A magnetic plug installed in the oil feedback pipe collects debris from the oil as evidence of the bearing degradation. The test will stop when the accumulated debris adhered to the magnetic plug exceeds a certain level and then causes an electrical switch to close. A PCB 353B33 High Sensitivity Quartz ICPs Accelerometer is installed on each bearing housing. Four thermocouples are attached to the outer race of each bearing to record bearing temperature for monitoring the lubrication purposes. The rotation speed was kept constant at 2000 rpm. Vibration signals were collected every 10 min at a sampling rate of 20 kHz by a National Instruments DAQ Card-6062E data acquisition card. Each signal segment has a length of 20480 data points. This run-to-failure test was carried out for about 164 h (a total of 982 samples) until a significant amount of metal debris was found on the magnetic plug of the test bearing. At the end of the test, a defect was found to occur on the outer ring of bearing #1.

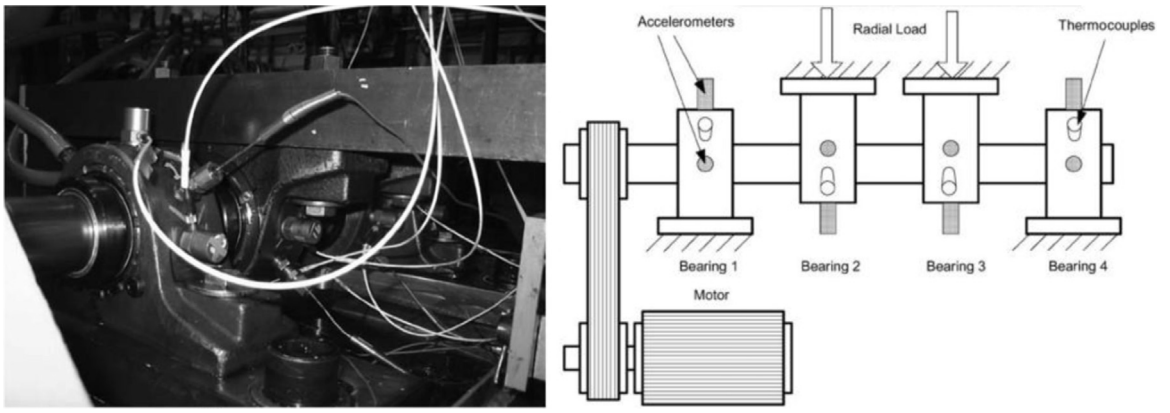

Fig. 6. Bearing test rig for the run-to-failure test.

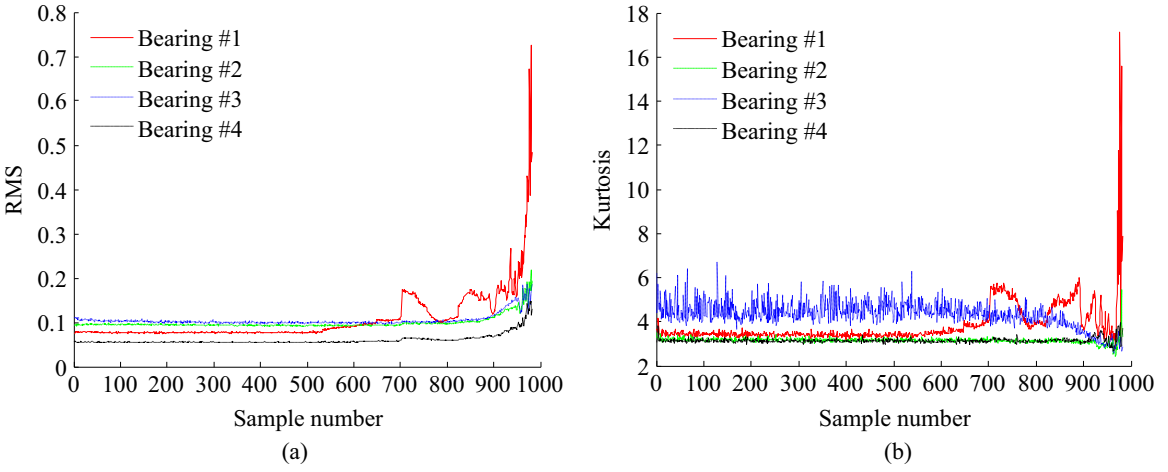

Fig. 7. Original feature plots of all the test bearings. (a) RMS (b) kurtosis.

4.2.1. Feature extraction

Vibration analysis is one of the most popular techniques in the fields of machine health monitoring. When bearing faults occur, the amplitudes and the distributions of acquired vibration signals in the time-domain are usually different from those under the normal state. Therefore, time-domain statistical parameters (e.g., root mean square, skewness, kurtosis, crest factor, impulse factor) are used widely as features for bearing fault detection [37–39]. In our work, only root mean square (RMS) and kurtosis were selected as the features.

RMS reflects the average vibration amplitude and hence is a good indicator of gradually developed faults. However, it can hardly detect incipient faults. On the contrary, kurtosis is sensitive to incipient faults and transient impulses. Therefore, the two complementary features can be employed jointly to detect both gradual deteriorations and abrupt faults. Another reason for choosing only two features is that a good visual presentation can be shown for two-dimensional data.

In terms of the whole lifetime of a bearing, the bearing always experiences a degradation process from the normal state to the abnormal state, and the time period for the normal state is generally much longer than that for the abnormal state. In this sense, a proper amount of data from the initial phase of the bearing lifetime can be considered to be normal. Here, the first 100 samples collected from each bearing were viewed as healthy data, then 400 feature vectors were extracted from these samples. To obtain a comprehensive description of the normal region of the bearings, we used all the feature vectors together to construct the monitoring model.

Due to individual differences in manufacturing, assembly, etc., the same type of feature extracted from different bearings under the normal state might exhibit significant difference. Fig. 7 shows the original RMS and kurtosis plots of all the test bearings. Although all the bearings are the same type and are tested under the same rotation speed, it can be observed that RMS or kurtosis values of different bearings have significantly different ranges. To obtain the feature values that perform consistent assessment to the health degradations of different bearings, we need to normalize the original RMS and kurtosis before utilizing them to construct monitoring models. The normalization strategy used here was realized as follows: firstly, the mean and the standard deviation of the 100 RMS values (kurtosis values) from each bearing were calculated separately; then, the RMS value (or kurtosis value) extracted from each sample, including each new sample to be monitored, was normalized by using the mean and the standard deviation of the bearing from which this sample was collected. The normalized RMS and kurtosis of all the test bearings are shown in Fig. 8. Clearly, the normalized RMS values or kurtosis values of different bearings have similar ranges and comparable fluctuation margins.

4.2.2. Monitoring index

In the field of anomaly detection, a sample is identified as an anomaly when its corresponding monitoring index value exceeds a control threshold. For OCCCH or OCSVM, the negative decision function value $-f(\mathbf{x})$ can be directly chosen as the monitoring index and the control threshold takes zero naturally. Hence, a sample with the monitoring index value larger than zero will be recognized as an anomaly.

4.2.3. 2.3 Results and discussion

Here, both the precision parameter of the generalized Gilbert algorithm and the KKT tolerance parameter of the SMO algorithm took the value 10^{-5}. The regularization parameter ν was set to 0.03 and the Gaussian kernel parameter σ was set to 2.976. The reduction factor μ was given by the conversion formula $\mu = 1/(\nu l)$. Fig. 9 shows the decision boundaries generated by OCCCH-1, OCCCH-2 and OCSVM, respectively. The relationships among the three methods are clearly illustrated again through the good visual presentation. That is, the decision boundaries of OCCCH-2 and OCSVM almost coincide, and the decision boundary of OCCCH-1 is looser than those of other two methods. Due to the above relationships,

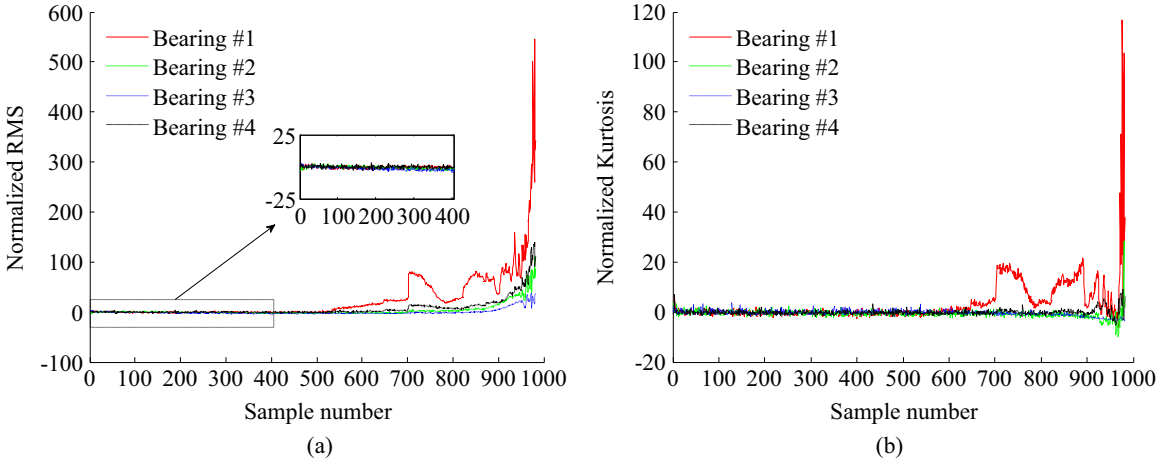

Fig. 8. Normalized feature plots of all the test bearings. (a) normalized RMS, and (b) normalized kurtosis.

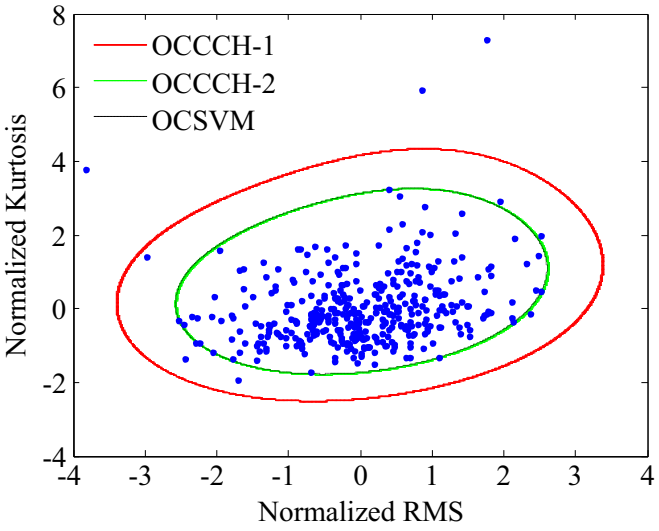

Fig. 9. Decision boundaries of different monitoring models on the bearing dataset.

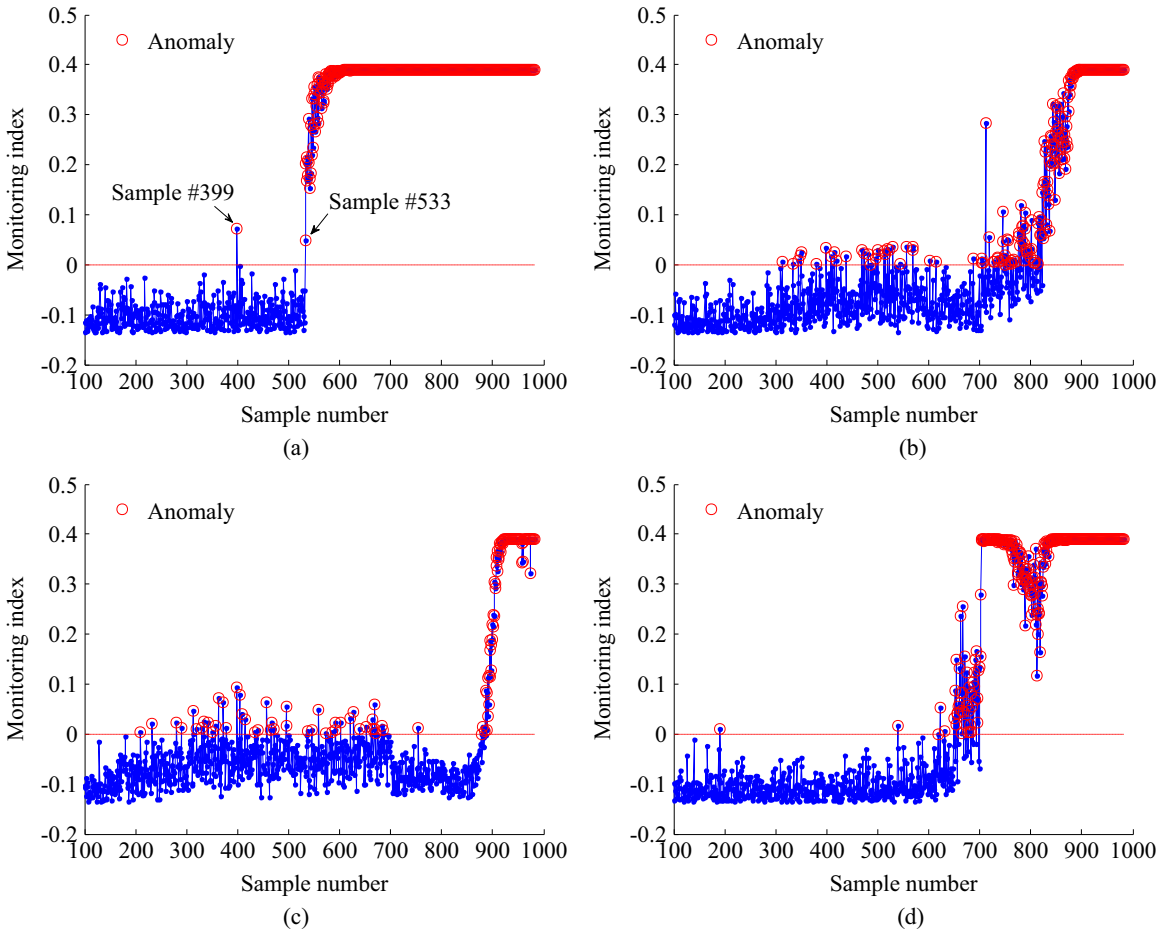

Fig. 10. Monitoring results of the OCCCH-1 model. (a) bearing #1, (b) bearing #2, (c) bearing #3, and (d) bearing #4.

only OCCCH-1 and OCSVM were employed to monitor the health conditions of all the test bearings from the sample #101 to the last sample (sample #982), respectively.

Fig. 10 shows the monitoring results of the OCCCH-1 model. For bearing #1, as shown in Fig. 10(a), a fault alarm is triggered at the sample #533, and then the fault alarm always exists until the end of the run-to-failure test. In order to verify

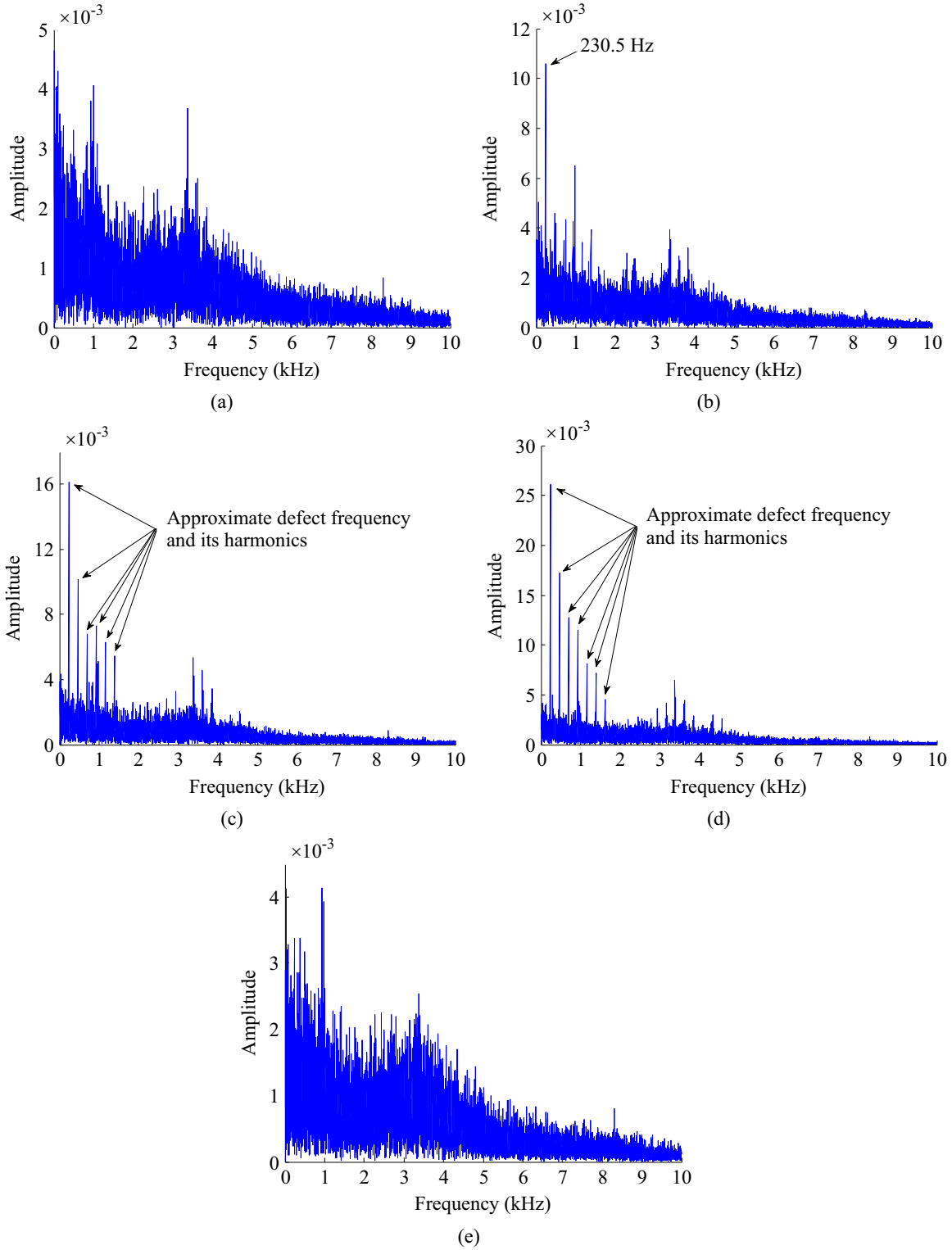

Fig. 11. Envelope spectrums of several samples from bearing #1. (a) sample #532, (b) sample #533, (c) sample #545, (d) sample #600, and (e) sample #399.

whether a bearing defect really occurred at the sample #533, we analyzed several signal samples using Hilbert transform based envelope spectrum analysis. If an obvious defect frequency or its harmonics exist in the envelope spectrum, the corresponding defect location could be usually recognized. Different defect locations (generally inner ring, outer ring, ball

and retainer) correspond to different defect frequencies (BPFI=296.9 Hz, BPFO=236.4 Hz, BSF=139.9 Hz, FTF=14.8 Hz). The envelope spectrums are shown in Fig. 11. At the sample #532, no obvious defect frequency can be found in its envelope spectrum. By contrast, we can observe a clear component frequency of 230.5 Hz in the envelope spectrum of the sample #533. This frequency is very close to the defect frequency of outer ring: BPFO=236.4 Hz. The existence of the approximate BPFO accords with the experimental observation on the run-to-failure test that a defect occurred on the outer ring of bearing #1. With the further performance deterioration of bearing #1, harmonics of the approximate BPFO begin to appear and corresponding spectrum amplitudes gradually increase (see Figs. 11(c) and (d)). From the above analysis, it is evident that an outer ring defect indeed occurred at the sample #533. At the same time, we also notice that only one fault alarm is triggered at the sample #399 before the sample #533. Fig. 11(e) shows the envelope spectrum of the sample #399, from which we cannot find the defect frequency. This means that the alarm at the sample #399 is a false alarm. Actually, if a monitoring index value exceeds the control threshold, there are two possible outcomes: (1) the bearing fault has occurred; (2) the bearing is still under the normal state and the alarm is more likely to be caused by random variations. Of course, it is unreasonable to conclude that an abnormal condition has occurred by sampling only one fault alarm. Generally, faults are likely to last for a time period, thus it is necessary to check several consecutive samples when the monitoring model triggers an alarm. If their monitoring index values all the control threshold, a bearing fault has occurred; otherwise, the bearing is considered to be still working under the normal state. In our experiments, five consecutive samples are checked when the monitoring model triggers an alarm. In summary, it can be concluded that bearing #1 was working continuously under the normal state until an outer ring defect occurred at the sample #533.

Regarding the monitoring results of other three bearings, we first concern the samples before the sample #533, where a defect occurred on the bearing #1. As shown in Figs. 10(b)–(d), no five consecutive anomalies are detected before the sample #533. Therefore, there is no enough evidence that any defect occurred on these bearings during this time period. Although a few samples are identified as anomalies before the sample #533, as mentioned before, these isolated anomalies are not caused by bearing defects but more likely to be caused by random variations. Compared with bearings #1 and #4, bearings #2 and #3 suffer more false alarms. This is because a radial load is imposed on each of the two bearings. Now, let's concern the samples collected from bearings #2, #3 and #4 after the sample #533. In the last phase of this monitoring period, the

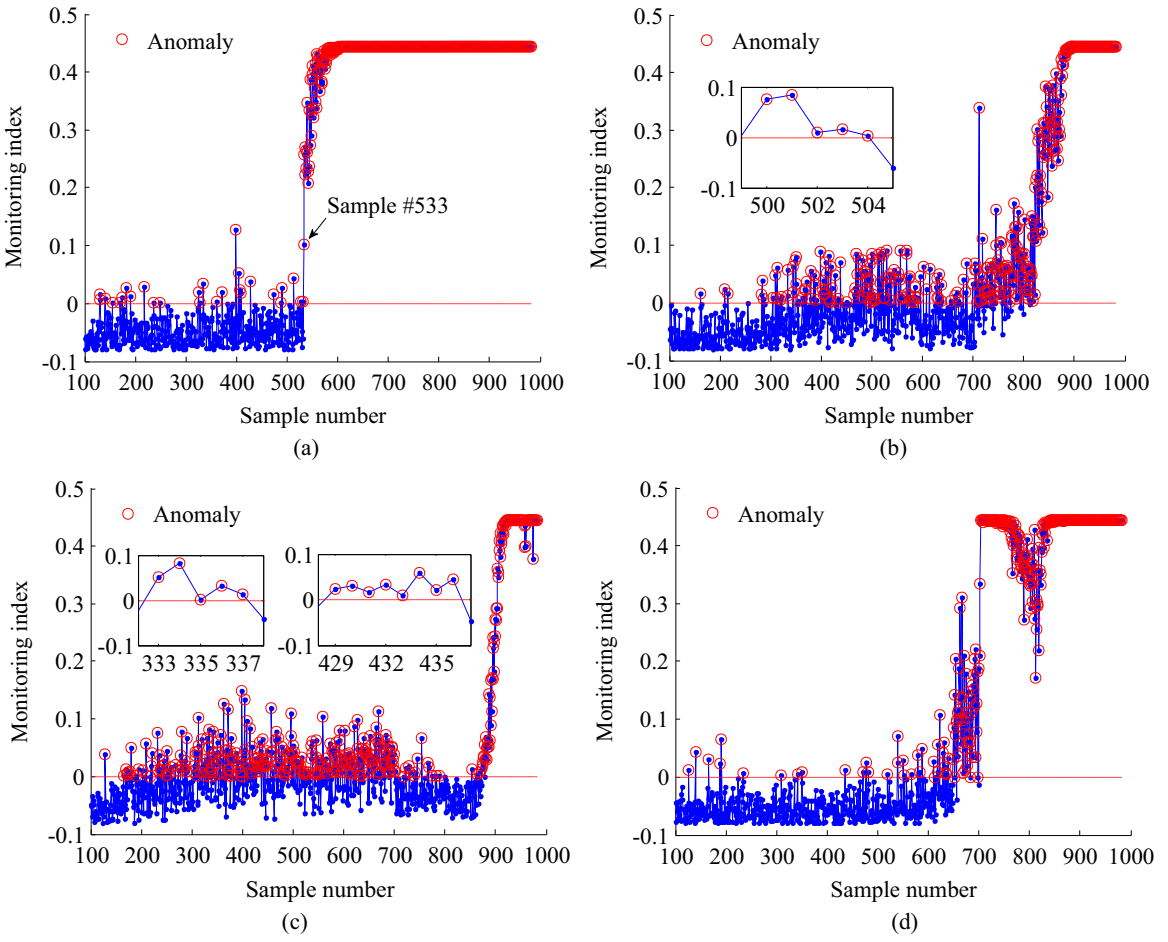

Fig. 12. Monitoring results of the OCSVM model. (a) bearing #1, (b) bearing #2, (c) bearing #3, and (d) bearing #4.

Table 4
False positive rates (%) of OCCCH-1 and OCSVM on the bearing dataset.

Method	Bearing #1	Bearing #2	Bearing #3	Bearing #4	Average FPR
OCCCH-1	0.23	5.32	6.02	0.23	2.95
OCSVM	6.25	19.44	38.43	3.24	16.84

anomalies seem to develop into faulty samples because there exist five consecutive monitoring index values all exceeding the control threshold. Yet the experimental observation on the run-to-failure test tells us bearings #2, #3, and #4 actually do not suffer any defect. As the four bearings are installed on the same shaft, the abnormal vibrations of bearing #1 resulting from the gradual performance deterioration transfer energy from bearing #1 to other three bearings, which consequently arouses their abnormal vibrations in the last phase of the run-to-failure test. In practice, once a bearing defect has been detected by the monitoring model, users will have to consider replacing the faulty bearing before its final failure, and then the monitoring index values of other normal coaxial bearings are supposed to drop below the control threshold.

For comparison, the monitoring results of the OCSVM model are shown in Fig. 12. For bearing #1, the OCSVM model also manages to detect the bearing defect at the sample #533. Moreover, there exists no five consecutive anomalies before the sample #533 during the monitoring processes for bearings #1 and #4. But for bearings #2 and #3, we observe five or even more than five consecutive anomalies appear (see the partial enlarged views in Figs. 12(b) and (c)). Consequently, the OCSVM model will consider that defects have occurred on the bearings #2 and #3. However, both bearing #2 and bearing #3 actually suffer no any defect. This means that the OCSVM model gives wrong monitoring conclusions. At the same time, for each of the four bearings, we observe that the OCSVM model reports more anomalies than the OCCCH-1 model before the sample #533. Table 4 gives the false positive rates of the OCCCH-1 and the OCSVM models for the four bearings. The false positive rate (FPR) is also known as the false alarm ratio. We only concern the monitored samples from #101 to # 532 because we have known that no bearing defect occurred before the sample #533. Any anomaly before the sample #533 is undoubtedly a false alarm. For each of the bearings, as listed in Table 4, the FPR resulting from the OCCCH-1 model is significantly lower than that of the OCSVM model. In terms of this bearing dataset, the natural health region of these bearings extends beyond the decision boundary of the OCSVM model. In other words, this decision boundary is a substantial under-approximation to the health region. As a result, some normal new samples that lie outside the decision boundary of the OCSVM mode are identified as anomalies incorrectly. As opposed to the OCSVM model, the OCCCH-1 model generates a looser decision boundary (see Fig. 9) that seems to describe the health region of the bearings more favorably. In this case, the looser decision boundary makes a contribution to reducing the FPR.

From the comparison between the monitoring results of the OCCCH-1 and the OCSVM models, it can be concluded that the OCCCH-1 model not only detects the bearing fault accurately but also receives lower FPR than the OCSVM model for this bearing dataset. As the decision boundary of the OCCCH-2 model nearly coincides with that of the OCSVM model (see Fig. 9), we can reasonably infer that the OCCCH-1 model also performs better than the OCCCH-2 model for this bearing dataset.

5. Conclusions

OCCCH is aimed at finding the nearest point from the reduced convex hull of training samples to the origin. A generalized Gilbert algorithm is proposed to solve this nearest point problem. No matter for the separable or the inseparable case, this geometric algorithm is applicable. Two different biases are provided for OCCCH, resulting in two different forms, i.e., OCCCH-1 and OCCCH-2. The relationships among OCCCH-1, OCCCH-2 and OCSVM were investigated theoretically. The normals of separating hyper-planes resulting from the three methods are completely the same. For appropriate choices of the parameters, OCCCH-2 is mathematically equivalent to OCSVM, which indicates that the two methods theoretically generate the same decision boundaries. In addition, OCCCH-1 is able to result in a looser decision boundary than both OCCCH-2 and OCSVM. We implemented OCCCH-1, OCCCH-2 and OCSVM on the benchmark datasets to investigate their respective performance. OCCCH using the generalized Gilbert algorithm is computationally more efficient than OCSVM using the well-known SMO algorithm. The superiority of OCCCH in the computational efficiency implies its potential applications in large-scale problems. In terms of the classification accuracy, OCCCH-2 always achieves comparable accuracies to OCSVM. However, OCCCH-1 performs worse than OCCCH-2 and OCSVM, due to its looser decision boundary. To verify the practicability of the proposed method, OCCCH was finally applied to the monitoring model constructions for bearing fault detection. The experimental results show that OCCCH-1 manages to detect the bearing fault accurately; what's more, it is superior to OCCCH-2 or OCSVM in terms of the FPR. For this application, the looser decision boundary of OCCCH-1 in turn contributes to the performance improvement in the classification accuracy. Note that the only difference between OCCCH-1 and OCCCH-2 lies in the utilization of the different biases, but there is no a priori evidence that which one is better. The choice between the two different forms depends on the real world applications.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant nos. 51175158 and 51375152) and the Hunan Provincial Innovation Foundation for Postgraduate (Grant no. CX2014B146).

Appendix A

When the case $\mu = 1/(\nu l)$ holds, we have $\alpha_i^* = \bar{\alpha}_i^*$ $(i = 1, 2, \cdots, l)$. Then

$$\begin{aligned} b^* &= \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i^* \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{i=1}^{l}\sum_{j=1}^{l} \bar{\alpha}_i^* \bar{\alpha}_j^* k(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{i=1}^{l} \bar{\alpha}_i^* \left(\sum_{j=1}^{l} \bar{\alpha}_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right) \\ &= \sum_{i=1}^{l} \bar{\alpha}_i^* \left(\sum_{j=1}^{l} \bar{\alpha}_j^* \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \right) \\ &= \sum_{i=1}^{l} \bar{\alpha}_i^* \left\langle \sum_{j=1}^{l} \bar{\alpha}_j^* \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_i) \right\rangle \end{aligned} \tag{A.1}$$

Submit Eq. (11) to Eq. (A.1), then generate

$$\begin{aligned} b^* &= \sum_{i=1}^{l} \bar{\alpha}_i^* \langle \bar{\mathbf{w}}^*, \Phi(\mathbf{x}_i) \rangle \\ &= \rho^* \sum_{i=1}^{l} \bar{\alpha}_i^* + \sum_{i=1}^{l} \bar{\alpha}_i^* \left(\langle \bar{\mathbf{w}}^*, \Phi(\mathbf{x}_i) \rangle - \rho^* \right) \end{aligned} \tag{A.2}$$

Note that $\sum_{i=1}^{l} \bar{\alpha}_i^* = 1$ and let $\xi_i^* = -(\langle \bar{\mathbf{w}}^*, \Phi(\mathbf{x}_i) \rangle - \rho^*)$ $(i = 1, 2, \cdots, l)$, we have

$$b^* = \rho^* - \sum_{i=1}^{l} \bar{\alpha}_i^* \xi_i^* \tag{A.3}$$

According to the KKT conditions, it is known that if $\bar{\alpha}_i^* = 1/(\nu l) = \mu$, then $\xi_i^* > 0$; otherwise, $\xi_i^* = 0$. Therefore, Eq. (A.3) can be further written as

$$b^* = \rho^* - \mu \sum_{i=1}^{l} \xi_i^* \tag{A.4}$$

The proof is complete.

Appendix B

When the case $\mu = 1/(\nu l)$ holds, we have $\alpha_i^* = \bar{\alpha}_i^*$ $(i = 1, 2, \cdots, l)$. Then

$$\begin{aligned} b^* &= \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i^* \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \\ &= \sum_{i \in \mathbf{I}_1} \alpha_i^* \left(\sum_{j=1}^{l} \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right) + \sum_{i \in \mathbf{I}_2} \alpha_i^* \left(\sum_{j=1}^{l} \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right) \\ &= \sum_{i \in \mathbf{I}_1} \alpha_i^* \left(\sum_{j=1}^{l} \bar{\alpha}_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right) + \mu \sum_{i \in \mathbf{I}_2} \left(\sum_{j=1}^{l} \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right) \end{aligned} \tag{B.1}$$

where $\mathbf{I}_1$ is the set of the indices of sample points whose combination coefficients satisfy $0 < \alpha_i^* < \mu$ and $\mathbf{I}_2$ is the set of the indices of sample points whose combination coefficients satisfy $\alpha_i^* = \mu$.

Submit Eq. (12) to Eq. (B.1)

$$
\begin{aligned}
b^* &= \rho^* \sum_{i\in \mathbf{I}_1} \alpha_i^* + \mu \sum_{i\in \mathbf{I}_2} \left(\sum_{j=1}^{l} \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right) \\
&= \rho^* \sum_{i\in \mathbf{I}_1} \alpha_i^* + \mu \sum_{i\in \mathbf{I}_2} \left(\sum_{j=1}^{l} \alpha_j^* \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \right) \\
&= \rho^* \sum_{i\in \mathbf{I}_1} \alpha_i^* + \mu \sum_{i\in \mathbf{I}_2} \left\langle \sum_{j=1}^{l} \alpha_j^* \Phi(\mathbf{x}_j), \Phi(\mathbf{x}_i) \right\rangle
\end{aligned} \tag{B.2}
$$

Submit Eq. (14) to Eq. (B.2)

$$
\begin{aligned}
b^* &= \rho^* \sum_{i\in \mathbf{I}_1} \alpha_i^* + \mu \sum_{i\in \mathbf{I}_2} \langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \rangle \\
&= \rho^* \sum_{i\in \mathbf{I}_1} \alpha_i^* + \mu \sum_{i\in \mathbf{I}_2} \left(\langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \rangle - b^* \right) + \mu \sum_{i\in \mathbf{I}_2} b^*
\end{aligned} \tag{B.3}
$$

Note that $\sum_{i=1}^{l} \alpha_i^* = \sum_{i\in \mathbf{I}_1} \alpha_i^* + l_2\mu = 1$ where l_2 is the size of the set $\mathbf{I}_2$, and let $\eta_i^* = -(\langle \mathbf{w}^*, \Phi(\mathbf{x}_i) \rangle - b^*)$ $(i = 1, 2, \cdots, l)$, we have

$$
b^* = \rho^*(1 - l_2\mu) - \mu \sum_{i\in \mathbf{I}_2} \eta_i^* + l_2\mu b^* \tag{B.4}
$$

Move ρ^* to the left of the equation, then generate

$$
\rho^* = b^* + \frac{\mu}{1 - l_2\mu} \sum_{i\in \mathbf{I}_2} \eta_i^* \tag{B.5}
$$

if $1 - l_2\mu \neq 0$. Let $b_2^* = \rho^*$, then the proof is complete.

References

[1] J. Luo, Z. Pan, G. Hu, A network security audit system based on support vector data description algorithm, CAAI Trans. Intell. Syst. 2 (4) (2007) 69–73.
[2] C. Wang, H. Yu, H. Wang, K. Liu, SOM-based anomaly intrusion detection system, in: T. Kuo, E. Sha, M. Guo, L.T. Yang, Z. Shao (Eds.), Proceedings of the 2007 IFIP International Conference on Embedded and Ubiquitous Computing, Springer, Berlin Heidelberg, Berlin, 2007, pp. 356–366.
[3] M. Hejazi, Y.P. Singh, One-class support vector machines approach to anomaly detection, Appl. Artif. Intell. 27 (5) (2013) 351–366.
[4] X. Di, H. Yang, H. Qi, Low-rate application-layer ddos attacks detection by principal component analysis (PCA) through user browsing behavior, Appl. Mech. Mater. 397 (2013) 1945–1948.
[5] G.S. Uttreshwar, A.A. Ghatol, Hepatitis B diagnosis using logical inference and self-organizing map, J. Comput. Sci. 4 (12) (2008) 1042–1050.
[6] G. Cohen, H. Sax, A. Geissbuhler, Novelty detection using one-class Parzen density estimator. An application to surveillance of nosocomial infections, Stud. Health Technol. Inform. 136 (2008) 21–26.
[7] J.M. Górriz, F. Segovia, J. Ramírez, A. Lassl, D. Salas-Gonzalez, GMM based SPECT image classification for the diagnosis of Alzheimer's disease, Appl. Soft Comput. 11 (2) (2011) 2313–2325.
[8] I. Saini, D. Singh, A. Khosla, Detection of QRS-complex using K-nearest neighbour algorithm, Int. J. Med. Eng. Inform. 5 (1) (2013) 81–101.
[9] Y. Pan, J. Chen, L. Guo, Robust bearing performance degradation assessment method based on improved wavelet packet – support vector data description, Mech. Syst. Signal Process. 23 (3) (2009) 669–681.
[10] S. Mahadevan, S.L. Shah, Fault detection and diagnosis in process data using one-class support vector machines, J. Process Control 19 (10) (2009) 1627–1639.
[11] R. Grbić, D. Slišković, P. Kadlec, Adaptive soft sensor for online prediction and process monitoring based on a mixture of Gaussian process models, Comput. Chem. Eng. 58 (2013) 84–97.
[12] Q. Jiang, X. Yan, Probabilistic weighted NPE-SVDD for chemical process monitoring, Control Eng. Pract. 28 (2014) 74–89.
[13] D.M.J. Tax, One-Class Classification, Ph.D. thesis, Delft University of Technology, Delft, Netherlands, 2001.
[14] O. Mazhelis, One-class classifiers: a review and analysis of suitability in the context of mobile-masquerader detection, South Afr. Comput. J. 36 (2006) 29–48.
[15] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.
[16] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification 2nd ed., John Wily & Sons, Inc., New York, 2000.
[17] T. Kohonen, Self-Organizing Maps, Springer-Verlag, Heidelberg, Germany, 1995.
[18] D.M.J. Tax, R.P.W. Duin, Support vector domain description, Pattern Recognit. Lett. 20 (11) (1999) 1191–1199.
[19] D.M.J. Tax, R.P.W. Duin, Support vector data description, Mach. Learn. 54 (1) (2004) 45–66.
[20] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.
[21] X.K. Wei, G.B. Huang, Y.H. Li, Mahalanobis ellipsoidal learning machine for one class classification, in: Proceedings of the 2007 IEEE International Conference on Machine Learning and Cybernetics, IEEE, Piscataway (NJ), 2007, pp. 3528–3533.
[22] B. Cyganek, One-class support vector ensembles for image segmentation and classification, J. Math. Imaging Vis. 42 (2–3) (2012) 103–117.
[23] B. Krawczyk, M. Wozniak, B. Cyganek, Clustering-based ensembles for one-class classification, Inf. Sci. 264 (2014) 182–195.
[24] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: B. Schölkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods–Support Vector Learning, MIT Press, Cambridge, 1999, pp. 185–208.
[25] B. Chazelle, An optimal convex hull algorithm in any fixed dimension, Discrete Comput. Geom. 10 (1) (1993) 377–409.
[26] X. Zhou, Y. Shi, Nearest neighbor convex hull classification method for face recognition, in: G. Allen, J. Nabrzyski, E. Seidel, G.D. Albada, J. Dongarra, P. M.A. Slootn (Eds.), Proceedings of the 9th International Conference on Computational Science-ICCS 2009, Springer Berlin Heidelberg, Berlin, 2009, pp. 570–577.
[27] P. Casale, O. Pujol, P. Radeva, Approximate convex hulls family for one-class classification, in: C. Sansone, J. Kittler, F. Roli (Eds.), Proceedings of the 10th

International Workshop on Multiple Classifier Systems, Springer Berlin Heidelberg, Berlin, 2011, pp. 106-115.
[28] P. Casale, O. Pujol, P. Radeva, Approximate polytope ensemble for one-class classification, Pattern Recognit. 47 (2) (2014) 854–864.
[29] E.G. Gilbert, An iterative procedure for computing the minimum of a quadratic form on a convex set, SIAM J. Control 4 (1) (1966) 61–80.
[30] J.B. Hiriart-Urruty, C. Lemaréchal, Convex Analysis and Minimization Algorithms I, Springer-Verlag, New York, 1991.
[31] M.E. Mavroforakis, S. Theodoridis, A geometric approach to support vector machine (SVM) classification, IEEE Trans. Neural Netw. 17 (3) (2006) 671–682.
[32] M. Lichman, UCI Machine Learning Repository ⟨http://archive.ics.uci.edu/ml⟩, University of California, Irvine, School of Information and Computer Science, 2013.
[33] C. Hsu, C. Chang, C. Lin, A practical Guide to Support Vector Classification ⟨http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf⟩.
[34] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection. In: D. Fisher(Ed.), Proceedings of the 14th International Conference on Machine Learning, Morgan Kaufmann, San Mateo (CA), 1997, pp. 179–186.
[35] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[36] H. Qiu, J. Lee, J. Lin, G. Yu, Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics, J. Sound Vib. 289 (4) (2006) 1066–1090.
[37] Y. Lei, Z. He, Y. Zi, Q. Hu, Fault diagnosis of rotating machinery based on multiple ANFIS combination with GAs, Mech. Syst. Signal Process. 21 (5) (2007) 2280–2294.
[38] Z. Xu, J. Xuan, T. Shi, B. Wu, Y. Hu, Application of a modified fuzzy ARTMAP with feature-weight learning for the fault diagnosis of bearing, Expert Syst. Appl. 36 (6) (2009) 9961–9968.
[39] B. Li, P. Liu, R. Hu, S. Mi, J. Fu, Fuzzy lattice classifier and its application to bearing fault diagnosis, Appl. Soft Comput. 12 (6) (2012) 1708–1719.