Structured One-Class Classification

Defeng Wang, *Student Member, IEEE,* Daniel S. Yeung, *Fellow, IEEE,*
and Eric C. C. Tsang, *Associate Member, IEEE*

***Abstract*—The one-class classification problem aims to distinguish a target class from outliers. The spherical one-class classifier (SOCC) solves this problem by finding a hypersphere with minimum volume that contains the target data while keeping outlier samples outside. SOCC achieves satisfactory performance only when the target samples have the same distribution tendency in all orientations. Therefore, the performance of the SOCC is limited in the way that many superfluous outliers might be mistakenly enclosed. The authors propose to exploit target data structures obtained via unsupervised methods such as agglomerative hierarchical clustering and use them in calculating a set of hyperellipsoidal separating boundaries. This method is named the structured one-class classifier (TOCC). The optimization problem in TOCC can be formulated as a series of second-order cone programming problems that can be solved with acceptable efficiency by primal-dual interior-point methods. The experimental results on artificially generated data sets and benchmark data sets demonstrate the advantages of TOCC.**

***Index Terms*—One-class classification, second-order cone programming (SOCP), structured learning, support vector machine (SVM).**

I. Introduction

IN ONE-CLASS classification [1] problems, usually, only one class of data is available, and others are too expensive to acquire or too difficult to characterize. These classification tasks can be found in many real-world scenarios like medical screening, machine faulty diagnosis, network security, etc. To solve these problems, one may find a boundary enclosing the available samples appropriately such that the chance of misclassification of unseen samples is minimized. The available class is called the target class, while all other samples not in this class are defined as outliers. In general, one cannot expect a one-class classifier to have as good performance as a two-class classifier because training samples from two classes provide more information to define the decision boundary than just sampling on one side [2]. Originating from various applications, one-class classification is also referred to as outlier detection, novelty detection, or concept learning [3]–[5].

To solve the one-class classification problem, a possible approach is to estimate the probability density function (pdf) of the data in the target class [4]. In the testing procedure, a new sample is labeled as outlier if its surrounding region has a probability density below a specified threshold. Instead of estimating the pdf, Schölkopf *et al.* proposed to separate the target samples from the origin with maximal margin using a hyperplane [6]. Implicitly mapping the training data to a high-dimensional kernel space to achieve more flexible classification results, this method shares a similar idea with large margin classifiers, e.g., support vector machine (SVM) by Vapnik [7].

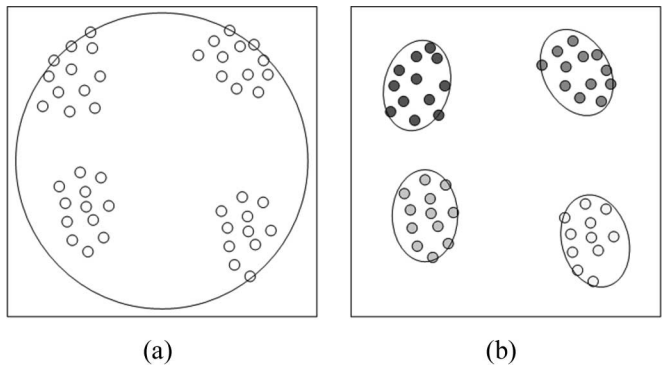


Fig. 1. Illustration of performing (a) spherical one-class classification and (b) structured one-class classification on the same 2-D data set.

Support vector data description was proposed in [8]. This method tries to describe the target data domain by finding a hypersphere that contains most of the target data. It minimizes the volume of this hypersphere (such that the chance of enclosing outlier test samples is minimized) as well as the number of target samples outside that hypersphere. In this paper, we name this approach the spherical one-class classifier (SOCC).

However, SOCC is a coarse data description and achieves good results only when the target data are roughly distributed within a single spherical region. Therefore, when using a spherical boundary to enclose the target samples, it is likely that many superfluous outliers are misclassified inside. This problem could be even more serious in the kernel space—using different kernel functions may result in sharply different class shapes in the kernel space. For example, mapping with the polynomial kernel leads to elongated classes [9], [10]. Even with the radial basis function (RBF) kernel, which is believed to map all the samples onto a hypersphere in an infinite-dimensional kernel space, it is still potentially problematic because the maps of the samples are generally not uniformly distributed on that hypersphere, but scattered with unequal densities. In an extreme case that the target samples are scattered in several small regions, using a spherical boundary to fit the data will enclose a large empty area and thus increase the chance of accepting outliers.

Manuscript received October 24, 2005; revised January 28, 2006, March 16, 2006, and March 20, 2006. This work was supported by the Hong Kong Research Grant Council under Grant B-Q519. This paper was recommended by Associate Editor Q. Zhao.

The authors are with the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: csdfwang@comp.polyu.edu.hk; csdaniel@comp.polyu.edu.hk; csetsang@comp.polyu.edu.hk).

Digital Object Identifier 10.1109/TSMCB.2006.876189

TABLE I
NOTATION AND VARIABLES

Notation	Meaning	Notation	Meaning
$\mathbf{x}_i$	the i^{th} training sample	$S_i^{(\Phi)}, M$	the i^{th} cluster, $i = 1, \cdots, M$
$\mathbf{x}, \ell$	input space, $\ell = dim(\mathbf{x})$	$\mathbf{D}_i^{\Phi}, S_i^{\Phi}$	image matrix containing sample maps in S_i^{Φ} as columns
Φ	mapping $\Phi : \Re^{\ell} \rightarrow \Re^{f}$	$\mathbf{K}_i$	kernel Gram Matrix $\mathbf{K}_i = (\mathbf{D}_i^{\Phi})^T \cdot \mathbf{D}_i^{\Phi}$
$\Phi(\mathbf{x}), f$	kernel space, $f = dim(\Phi(\mathbf{x}))$	μ_i	sample mean of cluster S_i
$\mathbf{x}_i \cdot \mathbf{x}_j$	Euclidean inner product of $\mathbf{x}_i$ and $\mathbf{x}_j$	μ_i^{Φ}	image sample mean of cluster S_i^{Φ}
a	center of a hypersphere/hyperellipsoid	$\mathbf{\Sigma}_i$	sample covariance matrix of cluster S_i
r	radius of a hypersphere/hyperellipsoid	$\mathbf{\Sigma}_i^{\Phi}$	image sample covariance matrix of cluster S_i^{Φ}
ξ_i	"slack-variable" for training sample $\mathbf{x}_i$	$\|S\|$	cardinality of set S
C	regularization parameter in SOCC/TOCC	$\|\cdot\|$	standard Euclidean norm
N	number of training samples	$(\cdot)^T$	vector/matrix transpose
$k(\cdot,\cdot)$	dot product in the kernel space		

On the other hand, data do appear in homogeneous groups in many applications, such as network intrusion detection [11], disease diagnosis [12], handwritten character recognition [13], and human face detection [14]. If the data structure information is appropriately utilized to facilitate classifier training, there would be significant improvement in the classification performance [15]–[18]. Fig. 1 illustrates the use of a single sphere and the use of multiple ellipsoids as decision boundaries.

In this paper, we propose to consider the data structures in one-class classification and use multiple hyperellipsoids to enclose the target training samples. Thus, this method is called the structured one-class classifier (TOCC). These multiple ellipsoidal boundaries generalize the single spherical boundary. In contrast to SOCC, TOCC is able to capture the target data structure and describe it in finer granularity so that the chance of outlier occurrence inside the boundary can be reduced. As TOCC considers data covariance information in minimizing the overall volume of the hyperellipsoids, it is no longer a quadratic programming problem as in SOCC. We demonstrate in this paper that this optimization problem can actually be formulated as a series of second-order cone programming (SOCP) problems, which is gaining increasing interest, and can be solved effectively by primal-dual interior-point methods. The synthetic data sets and the Matlab code to build and evaluate TOCC can be downloaded from http://www.comp.polyu.edu.hk/~csdfwang/tocc.htm. For clarity, Table I lists the notation that will be used in this paper. The bold typeface denotes vectors or matrices, and the normal typeface stands for scalars or vector components.

The rest of this paper is organized as follows. In Section II, SOCC will be briefly reviewed. Section III presents the detailed formulation of TOCC. The experiments of TOCC on artificially generated data sets and benchmark data sets are described and compared with SOCC in Section IV. Section V discusses some noteworthy aspects of TOCC and summarizes this paper.

II. SOCC

To describe the domain of a given target data set (generated independent identically distributed) containing N training samples $\mathbf{x}_i \in \Re^{\ell}$, $i = 1, \ldots, N$, SOCC [8] aims at enclosing every target sample $\mathbf{x}_i$ by the smallest hypersphere, which can be described by the center $\mathbf{a}$ and the radius $r > 0$. Hence, the optimization problem in SOCC is defined as

$$\begin{aligned} &\min\ r^2 \\ &\text{s.t.}\quad \|\mathbf{x}_i - \mathbf{a}\|^2 \le r^2, \qquad i = 1, \ldots, N \\ &\quad r > 0. \end{aligned} \tag{1}$$

In practice, it is not always feasible to find a decision boundary with 100% accuracy because there could be mixtures of outliers and target samples. Thus, the distance from $\mathbf{x}_i$ to the center $\mathbf{a}$ is allowed to be larger than r but carries some penalty. Slack variables $\xi_i \ge 0$ are introduced to enable soft boundary calculation and thus the optimization problem (1) changes to

$$\begin{aligned} &\min\ r^2 + C\sum_{i=1}^{N} \xi_i \\ &\text{s.t.}\quad \|\mathbf{x}_i - \mathbf{a}\|^2 \le r^2 + \xi_i, \qquad i = 1, \ldots, N \\ &\quad r > 0 \end{aligned} \tag{2}$$

where C is a parameter that specifies the tradeoff between the sphere volume and the errors. By using Lagrange multipliers $\alpha_i \ge 0$, problem (2) can be solved by dealing with its Wolfe dual problem

$$\begin{aligned} &\max \sum_i \alpha_i (\mathbf{x}_i \cdot \mathbf{x}_i) - \sum_{i,j} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ &\text{s.t.}\quad 0 \le \alpha_i \le C \\ &\quad \sum_i \alpha_i = 1, \qquad i = 1, \ldots, N. \end{aligned} \tag{3}$$

Now (3) is a standard quadratic optimization problem and can be solved by standard quadratic optimization packages. The hypersphere center $\mathbf{a}$ can be obtained by

$$\mathbf{a} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i. \tag{4}$$

The radius r is determined by calculating the distance from the center $\mathbf{a}$ to any support vector (SV) $\mathbf{x}_i$ on the boundary. In a typical one-class classification task, only a portion of samples $\mathbf{x}_i$ with $\alpha_i \geq 0$ are needed exclusively in hypersphere calculation. These samples are called the SVs of SOCC. A test sample $\mathbf{t}$ is accepted as target if its distance to the sphere center is smaller than or equal to the radius r, i.e.,

$$\|\mathbf{t}-\mathbf{a}\|^2 = (\mathbf{t}\cdot\mathbf{t}) - 2\sum_{i=1}^{N}\alpha_i(\mathbf{t}\cdot\mathbf{x}_i) + \sum_{i,j=1}^{N}\alpha_i\alpha_j(\mathbf{x}_i\cdot\mathbf{x}_j) \leq r^2. \tag{5}$$

For some problems, an improved target data description can be achieved using the so-called kernel trick, similar to that used in the nonlinear SVM [7]. The basic idea is to map data vectors from the input space to a high-dimensional kernel space using an implicit nonlinear mapping Φ and then minimize the volume of the hypersphere containing the data maps in the kernel space. The implicit mapping Φ is substantiated by employing the kernel function $k(\mathbf{x}_i, \mathbf{x})$, which fulfills Mercer's theorem [7], [9], to compute the dot products between SVs $\Phi(\mathbf{x}_i)$ and the pattern vector $\Phi(\mathbf{x})$ in the kernel space. For simplicity, we call the SOCC in the input space "i-SOCC" and its kernelized version "k-SOCC." For k-SOCC, we just replace the inner products $\mathbf{u}\cdot\mathbf{v}$ in (3) and (5) by kernel functions $k(\mathbf{u},\mathbf{v}) = \Phi(\mathbf{u})\cdot\Phi(\mathbf{v})$, where k is a positive definite kernel, or Mercer kernel. A variety of kernel functions can be used. Typical examples include the polynomial kernel and the RBF kernel [7], [9]. The polynomial kernel induces features that include products up to d degrees, i.e.,

$$k(\mathbf{u},\mathbf{v}) = (1+\mathbf{u}\cdot\mathbf{v})^d. \tag{6}$$

The RBF kernel (or Gaussian kernel) induces an infinite-dimensional kernel space in which all image vectors have the same norm, while the kernel width parameter σ controls the scaling of the mapping, i.e.,

$$k(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{\|\mathbf{u}-\mathbf{v}\|^2}{2\sigma^2}\right). \tag{7}$$

k-SOCC produces good results only if the target data images are distributed within an ideal spherically constrained domain, and is not adaptive to more complicated data distributions generally because the potential complexity of data structures in the kernel space has not been appropriately considered. For example, the polynomial kernel usually performs poorly in k-SOCC because it maps the samples into flat elongated clusters [9], [10]. Although the RBF kernel is believed to map the samples onto a hypersphere, there are still some structures among the mapped samples [18], [19], which are generally not within a single hypersphere. Based on these observations, we propose a generalized version of SOCC, namely TOCC.

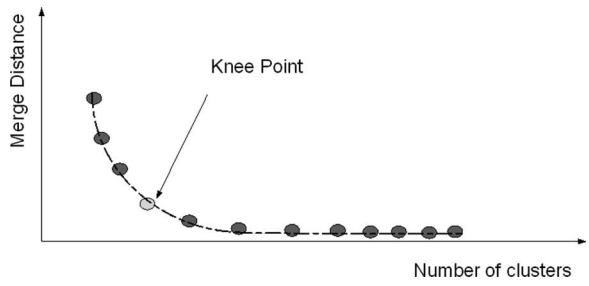


Fig. 2. Choosing knee point on the merge distance curve as the optimal number of clusters. Knee point is the point of maximum curvature.

III. TOCC

A. Detection of Data Structures in the Target Class

For the purpose of investigating the structures of the target class, we apply the agglomerative hierarchical clustering (AHC) [20] formally described as follows.

```
Initialize each point as a cluster and calculate the distance
between every two clusters
While more than one cluster remains
   Find the closest pair of clusters
   Merge the two clusters
   Update the distance between each pair of clusters
End
```

Specifically, Ward's linkage [21] is employed to find the closest pair of clusters for the reason that clusters obtained from this method are compact and spherical [22], and this provides a meaningful basis for the calculation of covariance matrix. If S and T are two clusters with means $\bar{S}$ and $\bar{T}$, respectively, Ward's linkage $W(S,T)$ between clusters S and T can be calculated as

$$W(S,T) = \frac{|S|\cdot|T|\cdot\|\bar{S}-\bar{T}\|^2}{|S|+|T|}. \tag{8}$$

Ward's linkage of two patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ is computed as $W(\mathbf{x}_i,\mathbf{x}_j) = \|\mathbf{x}_i-\mathbf{x}_j\|^2/2$. When two clusters A and B are being merged to a new cluster A', to be more computationally efficient, the Ward's linkage between A' and cluster C, i.e., $W(A',C)$, can be conveniently derived from

$$W(A',C) = \frac{(|A|+|C|)W(A,C)+(|B|+|C|)W(B,C)-|C|W(A,B)}{|A|+|B|+|C|}.$$

During hierarchical clustering, the Ward's linkage between clusters to be merged increases as the number of clusters decreases. The merge distance curve is thus drawn to represent this process. The output of AHC is a tree structure known as the dendrogram [23], which naturally determines diverse partitions when being cut at different levels. In practice, the dendrogram can be cut at the number of clusters corresponding to the knee point [24] (the point of maximum curvature) on the merge distance curve, as shown in Fig. 2.

To apply AHC to find data structures in k-TOCC, we give the following formulations to perform AHC in the kernel space.

1) The Ward's linkage between $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$, i.e., the images of patterns $\mathbf{x}_i$ and $\mathbf{x}_j$, can be calculated by [see (8)]

$$W\left(\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\right) = \frac{1}{2}\left[k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j)\right].$$

2) When two clusters A^Φ and B^Φ merge to a new cluster A'^Φ, the Ward's linkage between A'^Φ and C^Φ can be calculated as shown at the bottom of the page (see Appendix I for the derivation).

Complexity Analysis: In the initialization step, a Ward's linkage is calculated for each pair of patterns $\mathbf{x}_i$ and $\mathbf{x}_j$ [or $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$] in the training data; thus the time complexity for this step is $O(|N|^2 \cdot \ell)$. There are $|N| - 1$ rounds of merges, and the complexity for each round of merge is $O(\delta \cdot \ell)$, where δ is the number of clusters that monotonically decreases; hence the total complexity for these steps is $O((|N| - 1) \cdot \delta \cdot \ell)$. Because δ is always smaller than $|N|$, the overall complexity is $O(|N|^2 \cdot \ell)$ for detecting data structures in TOCC via the AHC algorithm.

B. TOCC in the Input Space

For the TOCC model in the input space (i-TOCC), assume that there are M clusters detected by the AHC data structure detection algorithm in the input space, i.e., $S_1, \ldots, S_i, \ldots, S_M$. Let $\mathbf{D}_i$ denote the $\ell \times |S_i|$ data matrix containing as columns data points in the ith cluster S_i with mean $\boldsymbol{\mu}_i \in \Re^\ell$ and covariance matrix $\boldsymbol{\Sigma}_i \in \Re^{\ell \times \ell}$ (symmetric and positive semidefinite). Both $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ can be expressed in terms of training samples in the ith cluster S_i as

$$\boldsymbol{\mu}_i = E_{\mathbf{x} \in S_i}[\mathbf{x}] = \mathbf{D}_i \vec{1} \tag{9}$$

$$\begin{aligned}\boldsymbol{\Sigma}_i &= E_{\mathbf{x} \in S_i}\left[(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T\right] \\ &= \frac{1}{|S_i|}\mathbf{D}_i\mathbf{D}_i^T - \mathbf{D}_i\vec{1}\vec{1}^T\mathbf{D}_i^T\end{aligned} \tag{10}$$

where $\vec{1}$ denotes a $|S_i|$-dimensional vector with each component equal to $1/|S_i|$. In i-TOCC, we wish to determine a series of hyperellipsoids for the M clusters

$$H(\mathbf{a}_i, r_i) = \left\{\mathbf{z} | (\mathbf{z} - \mathbf{a}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{z} - \mathbf{a}_i) = r_i^2\right\}, \qquad i = 1, \ldots, M$$

in which $\mathbf{a}_i \in \Re^\ell$ is the center and $r_i \in \Re$ is the radius for the ith hyperellipsoid enclosing the ith cluster. According to the existing structures in the training data, the target region is the union of the regions inside the M hyperellipsoids. For each cluster, different from the formulation in the SOCC, the distance metric adopted here is the Mahalanobis distance instead of the Euclidean distance. The Mahalanobis distance metric takes the data distribution information in each cluster into consideration. The optimization problem of the ith cluster S_i in i-TOCC can be mathematically formulated as

$$\begin{aligned}&\min\ r_i^2 \\ &\text{s.t. } (\mathbf{x}_j - \mathbf{a}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \mathbf{a}_i) \le r_i^2,\ \mathbf{x}_j \in S_i, \\ &\qquad j = 1, \ldots, |S_i|, r_i > 0.\end{aligned} \tag{11}$$

If the covariance matrix $\boldsymbol{\Sigma}_i$ is singular, it is impossible to calculate its inverse $\boldsymbol{\Sigma}_i^{-1}$ because it does not exist. Instead, we can calculate the pseudoinverse $\boldsymbol{\Sigma}_i^+ = \mathbf{P}^T\mathbf{G}^{-1}\mathbf{P}$ to approximate $\boldsymbol{\Sigma}_i^{-1}$ if the eigenstructure of the real symmetric and positive semidefinite matrix $\boldsymbol{\Sigma}_i$ is $\mathbf{P}^T\mathbf{G}\mathbf{P}$ [25]. $\mathbf{I}' = \boldsymbol{\Sigma}_i^+\boldsymbol{\Sigma}_i = \mathbf{P}^T\mathbf{P}$ is the minimum squared error approximation of the identity and acts as the true identity for both $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\Sigma}_i^+$. The optimization problem (11) can be easily transformed to

$$\begin{aligned}&\min\ r_i \\ &\text{s.t. } \left\|\boldsymbol{\Sigma}_i^{-\frac{1}{2}}(\mathbf{x}_j - \mathbf{a}_i)\right\| \le r_i,\ \mathbf{x}_j \in S_i, \\ &\qquad j = 1, \ldots, |S_i|, r_i > 0\end{aligned} \tag{12}$$

where $\boldsymbol{\Sigma}_i^{-1/2} = \mathbf{G}^{-1/2}\mathbf{P}$.

Similar to i-SOCC, we introduce slack variables $\xi_j \ge 0$ and set the constraints such that almost all samples are within the hyperellipsoid. The minimization problem is then expressed as

$$\begin{aligned}&\min\ r_i + C\sum_{j=1}^{|S_i|}\xi_j \\ &\text{s.t. } \left\|\boldsymbol{\Sigma}_i^{-\frac{1}{2}}(\mathbf{x}_j - \mathbf{a}_i)\right\| \le r_i + \xi_j,\ \xi_j \ge 0,\ \mathbf{x}_j \in S_i, \\ &\qquad j = 1, \ldots, |S_i|, r_i > 0.\end{aligned} \tag{13}$$

The parameter C controls the tradeoff between the hyperellipsoid's volume and the errors. If there is only one cluster in the training data, i.e., $M = 1$, and the covariance matrix $\boldsymbol{\Sigma}$ of this cluster is an identity matrix, the optimization problem involved in i-TOCC becomes essentially identical to the one in i-SOCC. Therefore, i-TOCC is a generalization of i-SOCC. A test point $\mathbf{t}$ is accepted as target if it falls in any of the M hyperellipsoids, i.e.,

$$\mathbf{t} \text{ is } \begin{cases}\text{a target sample,} & \text{if } \exists i \Big(\left\|\boldsymbol{\Sigma}_i^{-\frac{1}{2}}(\mathbf{t} - \mathbf{a}_i)\right\| \le r_i \\ & \qquad\qquad \text{and } i \in \{1, \ldots, M\}\Big) \\ \text{an outlier sample,} & \text{otherwise.}\end{cases}$$

In fact, both problems in (12) and (13) are instances of SOCP. The nonlinear constraints involved in (12) and (13) are called

$$W(A'^\Phi, C^\Phi) = \frac{\left(|A^\Phi| + |C^\Phi|\right) W(A^\Phi, C^\Phi) + \left(|B^\Phi| + |C^\Phi|\right) W(B^\Phi, C^\Phi) - |C^\Phi| W(A^\Phi, B^\Phi)}{|A^\Phi| + |B^\Phi| + |C^\Phi|}$$

second-order cone (SOC) constraints. An SOC constraint on the variable $\mathbf{v} \in \Re^\ell$ is in the form of

$$\|\mathbf{C}\mathbf{v} + \mathbf{d}\| \leq \mathbf{e}^T\mathbf{v} + b \tag{14}$$

where $\mathbf{e} \in \Re^\ell$, $\mathbf{d} \in \Re^m$, and $\mathbf{C} \in \Re^{m\times\ell}$ are given. Minimizing a linear objective over SOC and linear constraints is known as an SOCP problem. Recent advances in interior-point methods for convex nonlinear optimization [26] have made such problems feasible. As a special case of convex nonlinear optimization, SOCP has gained much attention recently and can be handled efficiently by existing software such as SeDuMi [27] and Mosek [28]. For a discussion of efficient algorithms and applications of SOCP, see Lobo *et al.* [29].

C. TOCC in the Kernel Space

To improve the data description ability, in this section, we describe the "kernelization" of TOCC (namely, k-TOCC). Each sample $\mathbf{x}_i$ is mapped into an implicit high-dimensional kernel space $\Re^f$ via an implicit mapping $\Phi : \Re^\ell \to \Re^f$. This nonlinear transformation can be achieved by using the Mercer kernel [7]. We first use the kernelized AHC to find the data structures in the kernel space and assume there exist M clusters: $S_1^\Phi, \ldots, S_i^\Phi, \ldots, S_M^\Phi$. Let D_i^Φ be the image sample matrix containing as columns the mapped data points $\Phi(\mathbf{x}_i)$ in the ith cluster S_i^Φ, and $\boldsymbol{\mu}_i^\Phi$ and $\boldsymbol{\Sigma}_i^\Phi$ are, respectively, the mean vector and covariance matrix of the corresponding cluster S_i^Φ. In k-TOCC, we seek to find a series of compact hyperellipsoids to enclose the M clusters, respectively, i.e.,

$$H\left(\mathbf{a}_i^\Phi, r_i\right) = \Big\{\Phi(\mathbf{z}) \in \Re^f \,|\, \left(\Phi(\mathbf{z}) - \mathbf{a}_i^\Phi\right)^T \times \boldsymbol{\Sigma}_i^{\Phi^{-1}} \left(\Phi(\mathbf{z}) - \mathbf{a}_i^\Phi\right) = r_i^2\Big\}, \qquad i = 1, \ldots, M.$$

Taking the cluster S_i^Φ for example, the hyperellipsoid can be obtained by solving the optimization problem

$$\begin{aligned} &\min r_i + C\sum_{j=1}^{|S_i^\Phi|} \xi_j \\ &\text{s.t. } \sqrt{\left(\Phi(\mathbf{x}_j) - \mathbf{a}_i^\Phi\right)^T \boldsymbol{\Sigma}_i^{\Phi^{-1}} \left(\Phi(\mathbf{x}_j) - \mathbf{a}_i^\Phi\right)} \leq r_i + \xi_j, \\ &\quad \xi_j \geq 0,\ \Phi(\mathbf{x}_j) \in S_i^\Phi, \qquad j = 1, \ldots, |S_i^\Phi|, r_i > 0. \end{aligned} \tag{15}$$

We need to reformulate the optimization problem in terms of a given kernel function $k(\mathbf{u}, \mathbf{v}) = \Phi(\mathbf{u})^T\Phi(\mathbf{v})$. The kernel trick will work only if problem (15) can be entirely expressed in terms of dot products of the mapped data point $\Phi(\mathbf{x})$. Because the kernel space decided by the kernel function is implicit, we are not able to directly manipulate center $\mathbf{a}$, image sample covariance matrix $\boldsymbol{\Sigma}_i^\Phi$, or its inverse $\boldsymbol{\Sigma}_i^{\Phi^{-1}}$. It is necessary to reformulate them in the form of dot products. Fortunately, this is indeed the case for center $\mathbf{a}$ provided we use appropriate estimates for the mean vector and covariance matrix of the cluster S_i^Φ, as defined in Theorem 1 (see Appendix II for proof).

Theorem 1: Let $\{\Phi(\mathbf{x}_j)\}_{j=1}^{|S_i^\Phi|}$ be the training data images in the kernel space via nonlinear mapping Φ. If $\boldsymbol{\mu}_i^\Phi$ and $\boldsymbol{\Sigma}_i^\Phi$ can be written as

$$\boldsymbol{\mu}_i^\Phi = \frac{1}{|S_i^\Phi|} \sum_{\Phi(\mathbf{x}_j)\in S_i^\Phi} \Phi(\mathbf{x}_j) \tag{16}$$

$$\boldsymbol{\Sigma}_i{}^\Phi = \frac{1}{|S_i^\Phi|} \sum_{\Phi(\mathbf{x}_j)\in S_i^\Phi} \left(\Phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\Phi\right)\left(\Phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\Phi\right)^T \tag{17}$$

then the optimal $\mathbf{a}_i^\Phi$ in problem (15) will lie in the span of the data images $\{\Phi(\mathbf{x}_j)\}_{j=1}^{|S_i^\Phi|}$.

Consequently, the center $\mathbf{a}_i^\Phi$ of the hyperellipsoid enclosing almost all the $|S_i^\Phi|$ samples lies in the space spanned by the data images in the cluster S_i^Φ. Thus, we can write the center $\mathbf{a}_i^\Phi$ as a linear combination of data images

$$\mathbf{a}_i^\Phi = \sum_{j=1}^{|S_i^\Phi|} w_i^j \Phi(\mathbf{x}_j) = \mathbf{D}_i^\Phi \mathbf{w}_i \tag{18}$$

and solve for the coefficients $\mathbf{w}_i$.

For the convenience of deriving the expression of $\boldsymbol{\Sigma}_i^{\Phi^{-1}}$ in terms of dot products of data images, we first define the centered kernel matrix $\mathbf{K}_i^C$ as

$$\mathbf{K}_i^C = \mathbf{K}_i - \mathbf{E}\mathbf{K}_i - \mathbf{K}_i\mathbf{E} + \mathbf{E}\mathbf{K}_i\mathbf{E} \tag{19}$$

where $\mathbf{E}$ is a $|S_i^\Phi| \times |S_i^\Phi|$ matrix with all entries equal to $1/|S_i^\Phi|$ [30]. The $|S_i^\Phi| \times |S_i^\Phi|$ symmetric kernel Gram matrix $\mathbf{K}_i = \{\Phi(\mathbf{x}_u) \cdot \Phi(\mathbf{x}_v)\}_{u,v} = \{k(\mathbf{x}_u, \mathbf{x}_v)\}_{u,v} = \mathbf{D}_I^{\Phi^T}\mathbf{D}_i^\Phi$ is a pairwise combination of dot products of the mapped samples in the cluster S_i^Φ. Then the following theorem (see Appendix III for the proof) holds.

Theorem 2: Let the eigenstructures of the centered kernel matrix $\mathbf{K}_i^C$ for the cluster S_i^Φ be denoted by $\mathbf{K}_i^C = \mathbf{A}_i^T\boldsymbol{\Omega}_i\mathbf{A}_i$. Then the covariance matrix $\boldsymbol{\Sigma}_i^\Phi$ can be diagonalized as

$$\boldsymbol{\Sigma}_i^\Phi = \left(\boldsymbol{\Omega}_i^{-\frac{1}{2}}\mathbf{A}_i\mathbf{D}_i^{\Phi^T}\right)^T \left(\frac{1}{|S_i^\Phi|}\boldsymbol{\Omega}_i\right)\left(\boldsymbol{\Omega}_i^{-\frac{1}{2}}\mathbf{A}_i\mathbf{D}_i^{\Phi^T}\right). \tag{20}$$

According to Theorem 2, we can approximate $\boldsymbol{\Sigma}_i^{\Phi^{-1}}$ by calculating the pseudoinverse $\boldsymbol{\Sigma}_i^{\Phi^+}$ as

$$\boldsymbol{\Sigma}_i^{\Phi^+} = \left|S_i^\Phi\right| \mathbf{D}_i^\Phi \mathbf{A}_i^T \boldsymbol{\Omega}_i^{-2} \mathbf{A}_i \mathbf{D}_i^{\Phi^T} \tag{21}$$

where exponent -2 denotes the squared inverse.

By simply substituting (18) and (21) into the optimization problem (15), we obtain the kernel form of (15) as (see Appendix IV for the derivation)

$$\begin{aligned} &\min\ r_i + C\sum_{j=1}^{|S_i^\Phi|} \xi_j \\ &\text{s.t. } \left\|\sqrt{\left|S_i^\Phi\right|}\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i(\mathbf{K}_i^j - \mathbf{K}_i\mathbf{w}_i)\right\| \leq r_i + \xi_j, \\ &\quad \xi_j \geq 0,\ \Phi(\mathbf{x}_j) \in S_i^\Phi, \qquad j = 1, \ldots, \left|S_i^\Phi\right|, r_i > 0 \end{aligned} \tag{22}$$

where $\mathbf{K}_i^j$ represents the jth column in the kernel Gram matrix $\mathbf{K}_i$. In (22), one can easily identify that the optimization problem is entirely expressed in terms of inner products between data images only, which makes k-TOCC solvable. The nonlinear constraints involved in (22) are exactly in the form of SOC defined in (14). The optimal ellipsoidal boundary $H(\mathbf{a}_i^{\Phi}, r_i)$ can be determined by solving the SOCP problem (22). Normally, only a proportion of data points with coefficients w_i^j are not zero, which are called the SVs of the k-TOCC. A test point $\mathbf{t}$ is discriminated as target if it falls in any of the M hyperellipsoids in the kernel space, as that

$$\mathbf{t} \text{ is } \begin{cases} \text{a target sample,} & \text{if } \exists i \Big(\Big\| \sqrt{|S_i^{\Phi}|}\, \boldsymbol{\Omega}_i^{-1} \mathbf{A}_i (\mathbf{K}_i^t - \mathbf{K}_i \mathbf{w}_i) \Big\| \le r_i, \\ & \qquad \text{and } i \in \{1, \ldots, M\} \Big) \\ \text{an outlier sample,} & \text{otherwise} \end{cases}$$

where $\mathbf{K}_i^t$ is a kernel column vector with the jth entry being $k(\mathbf{t}, \mathbf{x}_j)$, and $\Phi(\mathbf{x}_j) \in S_i^{\Phi}$.

To derive the theoretical generalization error bound of k-TOCC, we first recall the leave-one-out theorem from statistical learning theory [7], as stated in Lemma 1.

Lemma 1 (Leave-One-Out Theorem): The leave-one-out procedure gives an almost unbiased estimate of the expected generalization error.

Theorem 3 (Generalization Error Bound of k-TOCC): Given a training set with N samples, k-TOCC detects the structures of the training data (supposed to be M clusters), and then finds M hyperellipsoids containing almost all the samples with m_i SVs for cluster S_i^{Φ}. Thus, the leave-one-out error estimate on the target set holds, i.e.,

$$\widetilde{P}_{\text{error}} \le \frac{\sum_{i=1}^{M} m_i}{N}. \tag{23}$$

Proof: According to Lemma 1, we only need to show that the number of errors on the target set by the leave-one-out method does not exceed the number of SVs. Actually, if we leave a non-SV out and then perform training on the remaining target data, the decision hyperellipsoids will not change because they are only determined by the SVs. Therefore, non-SVs will be classified correctly, i.e., the number of leave-one-out errors does not exceed the number of SVs. ■

We now analyze the time complexity of TOCC. Suppose M clusters are detected in the training data via the AHC algorithm, then there are M independent SOCPs to be solved. As indicated in [29], if each SOCP is solved by the interior-point method, it has the worst-case complexity of $O(\ell^3)$ (ℓ represents the dimension of the data point). Adding the cost of forming the constraint matrix for cluster $S_i^{(\Phi)}$, which is $O(|S_i^{(\Phi)}| \cdot \ell^3)$, the complexity would be $O(|S_i^{(\Phi)}| \cdot \ell^3)$ for the SOCP in calculating the hyperellipsoid boundary of the cluster $S_i^{(\Phi)}$. Thus, the total time complexity for either i-TOCC or k-TOCC is $O(\sum_{i=1}^{M}(|S_i^{(\Phi)}| \cdot \ell^3)) = O(N \cdot \ell^3)$, which can be solved within polynomial time.

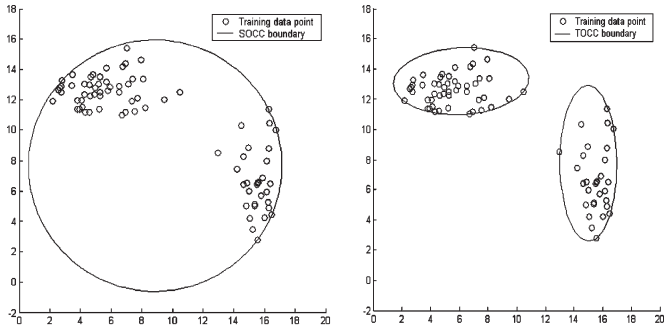


Fig. 3. Decision boundaries learned from (a) i-SOCC and (b) i-TOCC on an artificially generated two-cluster data set.

IV. Experiments

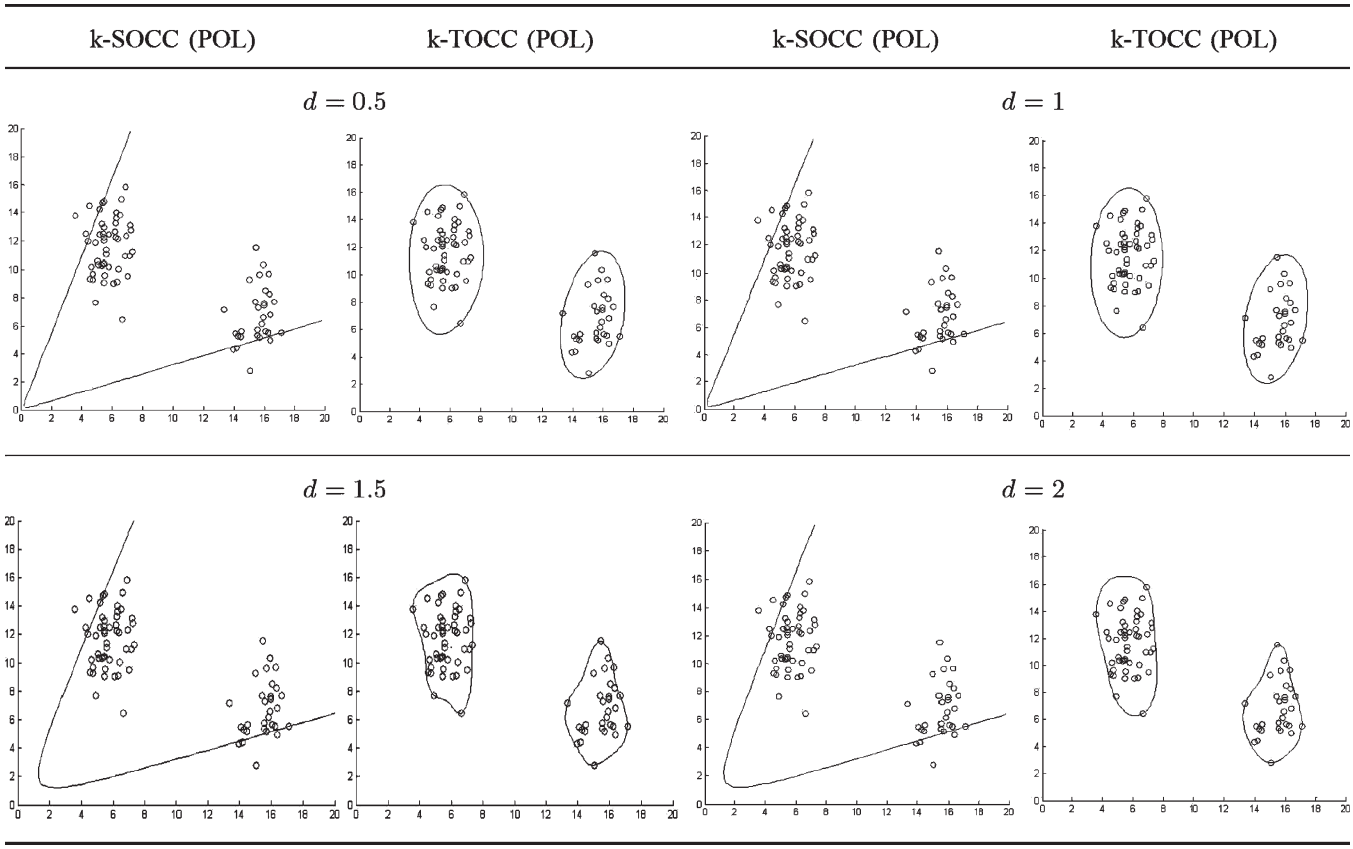
Our method described in Section III is implemented and tested on artificially generated data sets as well as several real-world benchmark data sets. To analyze the general trends of different classifiers' behaviors, we test our method on three synthetic data sets, where one is used for testing the classifiers in the input space, and the other two are generated to evaluate their kernelized versions with polynomial and RBF kernels, respectively. Experiments on real-world data sets give numerical results for assessment and comparison among different classifiers. All the programs were written in MATLAB 6.5 and executed on a personal computer with a single 1.2-GHz Pentium IV CPU and 256-MB memory. The SOCP problem was solved using the software SeDuMi [27]. For the implementation of k-SOCC, we used the SVM-KM toolbox (http://asi.insa-rouen.fr/~arakotom/toolbox/index.html).

A. Synthetic Data Sets

1) i-TOCC: We generate two clusters of two-dimensional (2-D) Gaussian-distributed training samples, each with different covariances along two features (see Fig. 3). With the regularization parameter C set to 0.4, we get the decision boundaries learned by i-SOCC and i-TOCC, respectively, in Fig. 3(a) and (b). Actually, although the boundary tightness can be controlled by the value of C (a larger C results in looser boundaries as less outliers are kept in the outside of the hypersphere and a smaller C corresponds to tighter boundaries), i-SOCC encloses all the target samples with a sphere that still occupies a much larger area than the overall area of the two ellipsoids calculated by i-TOCC. Therefore, i-TOCC is able to characterize the target class distribution much finer than i-SOCC so that the probability of false positive (FP) error is reduced by a significant degree.

2) k-TOCC: Kernelized classifiers usually have various behaviors given different kernel functions and different parameter settings. In the previous discussions, we see the merits of classifiers that are able to capture and utilize data structures. On the other hand, we expect the classifier to have relatively stronger stability, i.e., lower sensitivity to different kernels and parameter settings. Experiments in this subsection show

TABLE II
DECISION BOUNDARIES CALCULATED BY k-SOCC AND k-TOCC WITH POLYNOMIAL KERNELS GIVEN DIFFERENT VALUES FOR THE DEGREE PARAMETER d

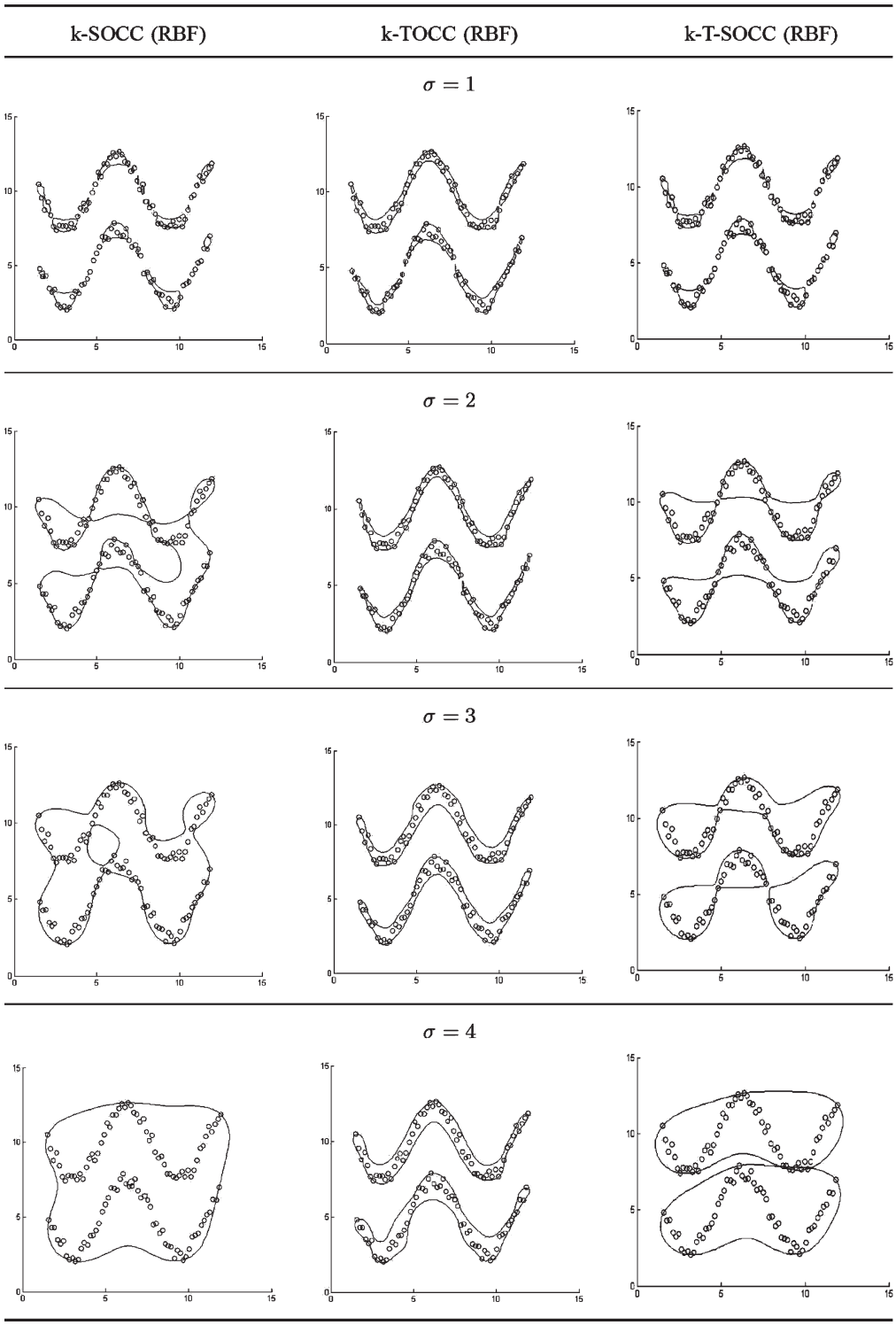


the evaluation and comparison of k-SOCC and k-TOCC using polynomial and RBF kernels with different parameter values.

With the polynomial kernel: In most cases, the performance of a classifier using the polynomial kernel is inferior to that using the RBF kernel. In this experiment, however, the decision boundary learned by k-TOCC with the polynomial kernel is still tight and discriminative, while the performance of k-SOCC with the same kernel is apparently inferior. We use a 2-D Gaussian-distributed training set with two clusters, as shown in Table II, and test the two classifiers by trying different values in $\{0.5, 1, 1.5, 2\}$ for the degree parameter d. The resulting boundaries generated by k-TOCC do not vary much, which shows the stability of k-TOCC in this case (see Table II). In comparison, the nonclosed boundaries calculated by k-SOCC (POL) enclose a very large (actually infinite) superfluous space as the target region, which is obviously inferior to that resulting from k-TOCC (POL). That is because the polynomial kernel maps the data clusters to elongated ones in the kernel space, and in this case, using a hypersphere to enclose these data images, just like SOCC, could include a very large superfluous space, which corresponds to the infinite areas in the input space. However, TOCC detects data clusters in the kernel space and uses hyperellipsoids to surround each of them separately so that the resulting decision boundaries in the input space enclose the samples tightly.

With the RBF kernel: In this experiment, a data set with samples in two band-shaped clusters distributed with wavy trends is generated to train the classifiers k-SOCC and k-TOCC with the RBF kernel when the kernel width parameter σ is set to $\{1, 2, 3, 4\}$. One may be interested in whether using a hypersphere in kernel space to enclose each cluster will also achieve satisfactory results. We name this approach the kernelized structured SOCC, i.e., k-T-SOCC. In fact, although the data clusters generated by Ward's linkage AHC tend to be spherical, this "spherical" is just a very general and rough description, not exactly the perfect sphere. However, in the SOCC algorithm, a perfect sphere has to be calculated precisely. Therefore, simply applying the SOCC algorithm to calculate a perfect sphere for each not really spherical cluster generated by AHC will still enclose superfluous areas and cannot yield desirable results. One may find the performances of k-SOCC (RBF), k-TOCC (RBF), and k-T-SOCC (RBF) in Table III. It turns out that given different σ values, the decision boundaries learned by k-TOCC are consistently tighter and more discriminative when compared with the boundaries learned by k-SOCC. For example, the resultant boundaries calculated by k-SOCC with σ equal to 1 leave superfluous empty regions in the inner part of the turning points while seemingly overfitting in other parts. As σ increases, k-SOCC tends to just rely on the outermost samples to generate a near-globular boundary and ignore the particulars of data structures so that the two previously separated boundaries gradually merge to one. In contrast, with slightly different degrees of tightness, the boundaries calculated by k-TOCC are always capable of capturing the data structures

TABLE III
RESULTING BOUNDARIES CALCULATED BY k-SOCC, k-TOCC, AND k-T-SOCC WITH RBF KERNELS GIVEN DIFFERENT VALUES FOR THE WIDTH PARAMETER σ



when σ is set to any value from 1 to 4. Whereas one can find that k-T-SOCC is able to keep the two significant bands well separated but still cannot generate tight boundaries as σ increases. That is because only cluster partition information is utilized while the distribution tendency of each cluster is ignored.

B. Real-World Data Sets

As there are few benchmark data sets for one-class classification tasks, we use data sets containing two or multiple classes, and each time we take one of them as the target class and all others as outliers. We use six benchmark data sets in the UCI machine learning repository [31] to test the performance of

TABLE IV
AUC RESULTS ACHIEVED ON UCI BENCHMARK DATA SETS WITH SOCC AND TOCC

Dataset	i-SOCC	i-TOCC	k-SOCC(POL)	k-TOCC(POL)	k-SOCC(RBF)	k-TOCC(RBF)
Balance						
1 (288,337)	80.44	85.61	93.06	95.40	96.83	97.71
2 (49,576)	51.50	58.43	66.88	68.96	75.47	80.03
3 (288,337)	83.46	87.72	90.49	95.92	97.01	97.85
Ionosphere						
1 (225,126)	83.88	87.62	92.35	95.81	95.79	96.25
2 (126,225)	41.46	56.07	60.23	63.59	69.85	76.44
Iris						
1 (50,100)	96.85	97.91	98.79	99.39	100.00	100.00
2 (50,100)	90.16	93.78	95.46	96.27	98.79	99.28
3 (50,100)	82.67	86.50	90.46	93.29	96.92	97.94
Liver-disorders						
1 (145,200)	50.10	52.41	56.35	56.91	58.90	61.33
2 (200,145)	40.31	45.56	48.27	48.94	50.65	56.73
Pima						
1 (500,268)	48.80	50.55	60.39	67.88	75.05	79.53
2 (268,500)	43.48	48.41	55.05	57.71	61.50	66.42
Wine						
1 (59,119)	59.13	63.84	68.37	77.92	75.32	89.85
2 (71,107)	56.78	58.66	65.88	68.92	69.77	78.37
3 (48,130)	55.19	60.15	59.91	65.75	63.63	71.82
Average	64.28	68.88	73.46	76.84	79.03	83.30

SOCC and TOCC. In Table IV, the class ID of each data set is followed by the number of target patterns and the number of outlier patterns.

Each data set is arbitrarily partitioned to a training set and a testing set with the same size. Tenfold cross validation is used to adjust the parameters C, d, and σ. Afterward, a receiver operating characteristic (ROC) curve [32] is obtained for each target class by plotting the true positive (TP) rate on the y axis and the FP rate on the x axis. The TP and FP rates are calculated as

$$\text{TP} = \frac{\text{target samples correctly classified}}{\text{total target samples}}$$

$$\text{FP} = \frac{\text{outliers incorrectly classified}}{\text{total outliers}}.$$

We use the area under the ROC curve (AUC), a commonly used indicator to evaluate one-class classifiers, as the performance measurement. AUC is calculated by integrating the TP rate when varying the FP rate. The AUC value is always between 0 and 1, and the larger AUC indicates the better performance [32]. AUC values in Table IV reflect the performances of i-SOCC, i-TOCC, and their kernelized version with the polynomial and RBF kernels. From these results, we conclude that for every data set, TOCC outperforms SOCC and kernelized methods achieve better results compared with their nonkernelized counterparts. It is therefore validated that the structure-based method is a generalized and improved strategy of the sphere-based ones.

TOCC outperforms SOCC mainly because of its proper consideration of the data structure information. To demonstrate the existence of data structures in kernel space, which is impractical to be directly displayed because of the infinite dimensionality, we choose to plot data images in kernel space by kernel principal component analysis (KPCA) [33], i.e., projecting them onto the three most principal kernel components in the kernel space. The structure of an arbitrarily selected data set in the RBF kernel space is illustrated in Fig. 4. Note that the value

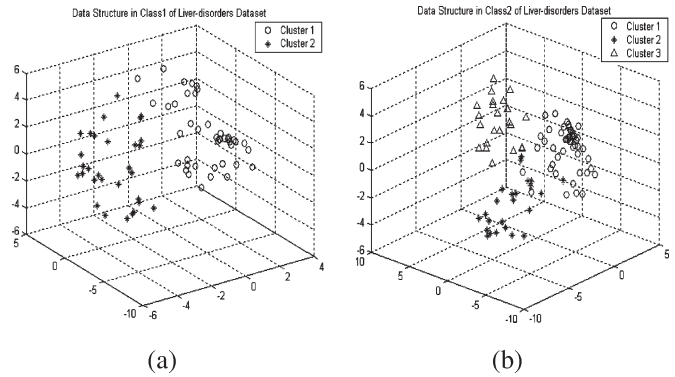


Fig. 4. Visualization of data structures in (a) class 1 and (b) class 2 of liver disorders data set by projecting the data images in the RBF kernel space onto the three most principal kernel components.

for the kernel width σ is just the same as that determined in k-TOCC training.

V. DISCUSSION AND CONCLUSION

SOCC achieves satisfactory results only when data distribute roughly in a globular shape, but generally data do not have such an ideal structure. Properly considering data distribution will refine the learning process greatly, which inspired the design of the TOCC. Although the training set size may affect the performance of TOCC, what matters most is not the training set size but its clarity in data structure. Theoretically, the chance of TOCC outperforming SOCC increases as the training set becomes larger, because the data distribution tendency is more pronounced when the data set increases in size, and thus the data structure detected by AHC is getting increasingly reliable. Based on the reliable structure to calculate the decision boundary, TOCC has a better data description ability than SOCC. On the contrary, as the training set shrinks, the data distribution tendency becomes obscure. In the case that there are too few samples to support the derivation of reliable clustering results, the performance of TOCC will become poor, and one can skip the clustering procedure and just calculate the covariance matrix of the whole training set. In practice, as the target samples are not difficult to acquire, normally there are large enough training sets for the clustering to be meaningful.

Another possible way, as presented in [10], [34], and [35], to incorporate data structures is by first mapping the samples onto the principal components of the covariance matrix and then rescaling the data in kernel space with the eigenvalues and eigenvectors of the centered kernel Gram matrix. Instead of detecting structures in the data, methods in this direction essentially preprocess the data, which is not guaranteed to generate data distributed compactly inside one hypersphere. Moreover, some information contained in the data are likely to be lost during the projection, and there is little theoretical basis to determine the number of principal components. In addition, these methods are of high testing complexity—to project each test sample onto a principal direction, the whole training set has to be taken into account.

For learning models with similar principles, such as mean square error minimizing approaches [multilayer perceptron (MLP) and RBF network (RBFN)], the "structured" one

(RBFN) generally outperforms the "unstructured" one (MLP). Large-margin classifiers have been widely applied, and the work in [18] has shown the merits of their "structured" counterparts. TOCC can be considered a substantiation of the structured learning concept to solve one-class classification problems.

TOCC can be further extended to have stronger scalability. After data structure information is extracted, all samples (except those near to the decision boundary) can be simply ignored as they are no longer helpful in decision hyperellipsoid determination [17]. Specifically, the Mahalanobis distances between each point and the centroid of the cluster it belongs to will be calculated, and based on the MD ranking, one can remove a proportion of data points that are close to the centroid. In TOCC, each data point corresponds to an SOC constraint in the optimization problem. Therefore, when the number of data points is reduced, the overall complexity of TOCC is reduced accordingly.

This paper presents a new one-class classifier, i.e., the TOCC, with merits of the "structured" learning. As it considers data distribution information, TOCC achieves better classification performance and generalizes the SOCC. Formulating the optimization problem as a series of SOCP problems, the TOCC method provides a direct way to calculate the multiple ellipsoidal boundaries that enclose the target data. The experimental results demonstrate the high utility of TOCC and also prove its stability. Motivated by the underlying idea of TOCC, more structured classifiers could be constructed by taking the data structure information into account.

Appendix I
Derivation of the Ward's Linkage Updating Formula in the Kernel Space

Suppose A^Φ, B^Φ, and C^Φ are clusters in the kernel space. A^Φ and B^Φ are combined together to form a larger cluster A'^Φ, i.e., $A'^\Phi = A^\Phi \cup B^\Phi$. We derive the formula for the new Ward's linkage between cluster A'^Φ and C^Φ.

According to Ward's linkage definition, we have

$$W\left(A'^\Phi, C^\Phi\right) = \frac{\left|A'^\Phi\right| \cdot |C^\Phi|}{\left|A'^\Phi\right| + |C^\Phi|} \left\| m_{A'^\Phi} - m_{C^\Phi} \right\|^2 \tag{24}$$

where m_{S^Φ} represents the mean of cluster S^Φ in the kernel space. Replacing $m_{A'^\Phi}$ with $(|A^\Phi| m_{A^\Phi} + |B^\Phi| m_{B^\Phi})/(|A^\Phi| + |B^\Phi|)$ in [24], we have

$$W\left(A'^\Phi, C^\Phi\right) = \frac{\left(|A^\Phi| + |B^\Phi|\right) \cdot |C^\Phi|}{|A^\Phi| + |B^\Phi| + |C^\Phi|} \times \left\| \frac{|A^\Phi| m_{A^\Phi} + |B^\Phi| m_{B^\Phi}}{|A^\Phi| + |B^\Phi|} - m_{C^\Phi} \right\|^2. \tag{25}$$

In fact

$$\begin{aligned} &\left\| |A^\Phi|(m_{A^\Phi} - m_{C^\Phi}) + |B^\Phi|(m_{B^\Phi} - m_{C^\Phi}) \right\|^2 \\ &= |A^\Phi|^2 \left\| m_{A^\Phi} - m_{C^\Phi} \right\|^2 + |B^\Phi|^2 \left\| m_{B^\Phi} - m_{C^\Phi} \right\|^2 \\ &\quad + |A^\Phi||B^\Phi| \left(\left\| m_{A^\Phi} - m_{C^\Phi} \right\|^2 + \left\| m_{B^\Phi} - m_{C^\Phi} \right\|^2 \right. \\ &\qquad \left. - \left\| m_{A^\Phi} - m_{B^\Phi} \right\|^2 \right) \\ &= |A^\Phi|^2 \frac{|A^\Phi| + |C^\Phi|}{|A^\Phi||C^\Phi|} W(A^\Phi, C^\Phi) \\ &\quad + |B^\Phi|^2 \frac{|B^\Phi| + |C^\Phi|}{|B^\Phi||C^\Phi|} W(B^\Phi, C^\Phi) + |A^\Phi||B^\Phi| \\ &\quad \times \left(\frac{|A^\Phi| + |C^\Phi|}{|A^\Phi||C^\Phi|} W(A^\Phi, C^\Phi) + \frac{|B^\Phi| + |C^\Phi|}{|B^\Phi||C^\Phi|} W(B^\Phi, C^\Phi) \right. \\ &\qquad \left. - \frac{|A^\Phi| + |B^\Phi|}{|A^\Phi||B^\Phi|} W(A^\Phi, B^\Phi) \right) \end{aligned} \tag{26}$$

where $W(A^\Phi, C^\Phi) = (|A^\Phi| \cdot |C^\Phi|)/(|A^\Phi| + |C^\Phi|) \| m_{A^\Phi} - m_{C^\Phi} \|^2$, $W(B^\Phi, C^\Phi) = (|B^\Phi| \cdot |C^\Phi|)/(|B^\Phi| + |C^\Phi|) \| m_{B^\Phi} - m_{C^\Phi} \|^2$, and $W(A^\Phi, B^\Phi) = (|A^\Phi| \cdot |B^\Phi|)/(|A^\Phi| + |B^\Phi|) \| m_{A^\Phi} - m_{B^\Phi} \|^2$.

We can substitute (26) into (25) and then obtain the expression shown at the bottom of the page.

Appendix II
Proof of Theorem 1

Proof: Assume that $\mathbf{a}_i^\Phi = \mathbf{a}_p + \mathbf{a}_q$, where $\mathbf{a}_p$ is the projection of $\mathbf{a}_i^\Phi$ in the vector space spanned by all the kernel maps in the ith cluster S_i^Φ, and $\mathbf{a}_q$ is the perpendicular component to this vector space. Then the optimization problem in k-TOCC model is

$$\begin{aligned} &\min\ r_i + C \sum_{j=1}^{|S_i^\Phi|} \xi_j \\ &\text{s.t.} \sqrt{\left(\Phi(\mathbf{x}_j) - \mathbf{a}_p - \mathbf{a}_q\right)^T \mathbf{\Sigma}_i^{\Phi-1} \left(\Phi(\mathbf{x}_j) - \mathbf{a}_p - \mathbf{a}_q\right)} \\ &\qquad \le r_i + \xi_j, \quad \xi_j \ge 0, \Phi(\mathbf{x}_j) \in S_i^\Phi, r_i > 0. \end{aligned} \tag{27}$$

According to this assumption, $\mathbf{a}_q$ is orthogonal to $\Phi(\mathbf{x}_j)$, and we have

$$\mathbf{a}_q^T \Phi(\mathbf{x}_j) = 0. \tag{28}$$

Supported by Theorem 2, $\mathbf{\Sigma}_i^{\Phi^{-1}}$ can be represented in terms of $\mathbf{D}_i^\Phi$ as

$$\mathbf{\Sigma}_i^{\Phi-1} = \left|S_i^\Phi\right| \mathbf{D}_i^\Phi \mathbf{A}_i^T \mathbf{\Omega}_i^{-2} \mathbf{A}_i \mathbf{D}_i^{\Phi^T}. \tag{29}$$

$$W\left(A'^\Phi, C^\Phi\right) = \frac{\left(|A^\Phi| + |C^\Phi|\right) W(A^\Phi, C^\Phi) + \left(|B^\Phi| + |C^\Phi|\right) W(B^\Phi, C^\Phi) - |C^\Phi| W(A^\Phi, B^\Phi)}{|A^\Phi| + |B^\Phi| + |C^\Phi|}$$

Using (28) and (29), we can easily obtain

$$\mathbf{a}_q^T \boldsymbol{\Sigma}_i^{\Phi-1} = \vec{0}. \tag{30}$$

By substituting (30) into the optimization problem (27), we have

$$\begin{aligned} &\min\ r_i + C\sum_{j=1}^{|S_i^\Phi|}\xi_j \\ &\text{s.t.}\quad \sqrt{\left(\Phi(\mathbf{x}_j)-\mathbf{a}_p\right)^T \boldsymbol{\Sigma}_i^{\Phi-1}\left(\Phi(\mathbf{x}_j)-\mathbf{a}_p\right)} \\ &\qquad \le r_i+\xi_j, \qquad \xi_j \ge 0,\ \Phi(\mathbf{x}_j)\in S_i^\Phi, r_i>0. \end{aligned} \tag{31}$$

Inspecting the difference between the optimization problems (27) and (31), one can find that $\mathbf{a}_p + \mathbf{a}_q = \mathbf{a}_p$, i.e., $\mathbf{a}_q = 0$. This means that the optimal $\mathbf{a}_i^\Phi$ follows in the vector space spanned by all the training data images in cluster S_i^Φ. ■

APPENDIX III
PROOF OF THEOREM 2

Proof: We first recall the expression of the covariance matrix $\boldsymbol{\Sigma}_i^\Phi$ in kernel space

$$\begin{aligned} \boldsymbol{\Sigma}_i^\Phi &= \frac{1}{|S_i^\Phi|}\sum_{\Phi(\mathbf{x}_j)\in S_i^\Phi}\left(\Phi(\mathbf{x}_j)-\boldsymbol{\mu}_i^\Phi\right)\left(\Phi(\mathbf{x}_j)-\boldsymbol{\mu}_i^\Phi\right)^T \\ &= \frac{1}{|S_i^\Phi|}\mathbf{D}_i^\Phi \mathbf{D}_i^{\Phi T} - \mathbf{D}_i^{\Phi T}\vec{1}\vec{1}^T\mathbf{D}_i^\Phi \end{aligned} \tag{32}$$

where $\vec{1}$ denotes a $|S_i^\Phi|$-dimensional vector with all the components equal to $1/|S_i^\Phi|$.

The spectral decomposition of the covariance matrix $\boldsymbol{\Sigma}_i^\Phi$ could be assumed as in the form

$$\boldsymbol{\Sigma}_i^\Phi = \mathbf{B}^T\mathbf{P}\mathbf{B}. \tag{33}$$

By combining (32) and (33), the equation holds for

$$\frac{1}{|S_i^\Phi|}\mathbf{D}_i^\Phi \mathbf{D}_i^{\Phi T} - \mathbf{D}_i^{\Phi T}\vec{1}\vec{1}^T\mathbf{D}_i^\Phi = \mathbf{B}^T\mathbf{P}\mathbf{B}. \tag{34}$$

Considering the fact that eigenvectors lie in the span of the centered data, $\mathbf{B}$ can then be written as the linear combination

$$\mathbf{B} = \eta\left(\mathbf{D}_i^{\Phi T} - \mathbf{E}\mathbf{D}_i^{\Phi T}\right) \tag{35}$$

where $\mathbf{E}$ is a $|S_i^\Phi|\times|S_i^\Phi|$ matrix with all entries equal to $1/|S_i^\Phi|$. Multiplying (34) by $\mathbf{D}_i^{\Phi T} - \mathbf{E}\mathbf{D}_i^{\Phi T}$ from the left side, and by $\mathbf{B}^T$ from the right side, respectively, at the same time, we have

$$\frac{1}{|S_i^\Phi|}\left(\mathbf{K}_i^C\right)^2\eta^T = \mathbf{K}_i^C\eta^T\mathbf{P} \tag{36}$$

where $\mathbf{K}_i^C = \mathbf{K}_i - \mathbf{E}\mathbf{K}_i - \mathbf{K}_i\mathbf{E} + \mathbf{E}\mathbf{K}_i\mathbf{E}$ is the centered kernel matrix. Multiplying (36) by the pseudoinverse $\mathbf{K}_i^{C+}$ from the left side, we have

$$\frac{1}{|S_i^\Phi|}\left(\mathbf{K}_i^C\right)\eta^T = \eta^T\mathbf{P}. \tag{37}$$

As matrix $\mathbf{P}$ is diagonal and the centered kernel matrix $\mathbf{K}_i^C$ could be decomposed as

$$\mathbf{K}_i^C = \mathbf{A}_i^T\boldsymbol{\Omega}_i\mathbf{A}_i \tag{38}$$

we can easily obtain

$$\mathbf{P} = \frac{1}{|S_i^\Phi|}\boldsymbol{\Omega}_i \tag{39}$$

$$\eta = \lambda\mathbf{A}_i. \tag{40}$$

λ is some diagonal matrix. Because the eigenvectors of the covariance matrix $\boldsymbol{\Sigma}_i^\Phi$ are orthogonal, we have

$$\begin{aligned} \mathbf{I} = \mathbf{B}\mathbf{B}^T &= \eta\left(\mathbf{D}_i^{\Phi T} - \mathbf{E}\mathbf{D}_i^{\Phi T}\right)\left(\mathbf{D}_i^\Phi - \mathbf{D}_i^\Phi\mathbf{E}\right)\eta^T \\ &= \lambda\mathbf{A}_i\mathbf{K}_i^C\mathbf{A}_i^T\lambda \\ &= \lambda\boldsymbol{\Omega}_i\lambda. \end{aligned} \tag{41}$$

From (41), it is obvious that

$$\lambda = \boldsymbol{\Omega}_i^{-\frac{1}{2}}. \tag{42}$$

Multiplying (38) by $\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i$ from the left side and by $\mathbf{E}$ from the right side, respectively, at the same time, we obtain $\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i\mathbf{K}_i^C\mathbf{E} = \mathbf{A}_i\mathbf{E}$. Because $\mathbf{K}_i^C$ is a centralized matrix, we have $\mathbf{K}_i^C\mathbf{E} = 0$. Therefore

$$\mathbf{A}_i\mathbf{E} = 0. \tag{43}$$

From (35), using (40), (42), and (43), we obtain

$$\mathbf{B} = \boldsymbol{\Omega}_i^{-\frac{1}{2}}\mathbf{A}_i\mathbf{D}_i^{\Phi T}. \tag{44}$$

Considering (33), (39), and (44) together, we finally have

$$\boldsymbol{\Sigma}_i^\Phi = \left(\boldsymbol{\Omega}_i^{-\frac{1}{2}}\mathbf{A}_i\mathbf{D}_i^{\Phi T}\right)^T\left(\frac{1}{|S_i^\Phi|}\boldsymbol{\Omega}_i\right)\left(\boldsymbol{\Omega}_i^{-\frac{1}{2}}\mathbf{A}_i\mathbf{D}_i^{\Phi T}\right). \tag{45}$$

APPENDIX IV
DERIVATION OF THE OPTIMIZATION PROBLEM IN KERNELIZED TOCC

By substituting (18) and (21) into the left side of the nonlinear constraints in optimization problem (15), we obtain (46),

$$\sqrt{\left(\Phi(\mathbf{x}_j)-\mathbf{a}_i^{\Phi}\right)^T\boldsymbol{\Sigma}_i^{\Phi-1}\left(\Phi(\mathbf{x}_j)-\mathbf{a}_i^{\Phi}\right)}=\sqrt{\left(\Phi(\mathbf{x}_j)-\mathbf{D}_i^{\Phi}\mathbf{w}_i\right)^T\left|S_i^{\Phi}\right|\mathbf{D}_i^{\Phi}\mathbf{A}_i^T\boldsymbol{\Omega}_i^{-2}\mathbf{A}_i\mathbf{D}_i^{\Phi^T}\left(\Phi(\mathbf{x}_j)-\mathbf{D}_i^{\Phi}\mathbf{w}_i\right)}$$
$$=\left\|\sqrt{\left|S_i^{\Phi}\right|}\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i\mathbf{D}_i^{\Phi^T}\Phi(\mathbf{x}_j)-\sqrt{\left|S_i^{\Phi}\right|}\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i\mathbf{D}_i^{\Phi^T}\mathbf{D}_i^{\Phi}\mathbf{w}_i\right\|$$
$$=\left\|\sqrt{\left|S_i^{\Phi}\right|}\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i\mathbf{K}_i^j-\sqrt{\left|S_i^{\Phi}\right|}\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i\mathbf{K}_i\mathbf{w}_i\right\| \tag{46}$$

shown at the top of the page, where $\mathbf{K}_i^j$ represents the jth column in kernel Gram matrix $\mathbf{K}_i$. By rewriting the nonlinear constrains in (15) using (46), problem (15) is changed to

$$\min\ r_i+C\sum_{j=1}^{\left|S_i^{\Phi}\right|}\xi_j$$
$$\text{s.t.}\ \left\|\sqrt{\left|S_i^{\Phi}\right|}\boldsymbol{\Omega}_i^{-1}\mathbf{A}_i\left(\mathbf{K}_i^j-\mathbf{K}_i\mathbf{w}_i\right)\right\|$$
$$\leq r_i+\xi_j,\quad \xi_j\geq 0,\ \Phi(\mathbf{x}_j)\in S_i^{\Phi},\ j=1,\ldots,\left|S_i^{\Phi}\right|,\ r_i>0. \tag{47}$$

ACKNOWLEDGMENT

The authors would like to thank the Editor and the anonymous reviewers for their valuable and constructive suggestions that made this paper more complete and convincing.

REFERENCES

[1] M. Moya, M. Koch, and L. Hostetler, "One-class classifier networks for target recognition applications," in *Proc. World Congr. Neural Netw.*, Portland, OR, 1993, pp. 797–801.
[2] H. Yu, "SVMC: Single-class classification with support vector machines," in *Proc. IJCAI*, 2003, pp. 567–572.
[3] G. Ritter and M. Gallegos, "Outliers in statistical pattern recognition and an application to automatic chromosome classification," *Pattern Recognit. Lett.*, vol. 18, no. 6, pp. 525–539, Jun. 1997.
[4] L. Tarassenko, P. Hayton, and M. Brady, "Novelty detection for the identification of masses in mammograms," in *Proc. 4th Int. IEE Conf. Artif. Neural Netw.*, 1995, vol. 409, pp. 442–447.
[5] N. Japkowicz, "Concept-learning in the absence of counterexamples: An autoassociation-based approach to classification," Ph.D. dissertation, State Univ. New Jersey, New Brunswick, 1999.
[6] B. Schölkopf, R. Williamson, A. Smola, and J. Shawe-Taylor, "SV estimation of a distribution's support," in *Proc. NIPS*, 1999, pp. 582–588.
[7] V. Vapnik, *Statistical Learning Theory*. Hoboken, NJ: Wiley, 1998.
[8] D. Tax and R. Duin, "Support vector data description," *Mach. Learn.*, vol. 54, no. 1, pp. 45–66, Jan. 2004.
[9] B. Schölkopf and A. J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2001.
[10] D. Tax and P. Juszczak, "Kernel whitening for one-class classification," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 17, no. 3, pp. 333–347, 2003.
[11] B. Mukherjee, L. Heberlein, and K. Levitt, "Network intrusion detection," *IEEE Network*, vol. 8, no. 3, pp. 26–41, May/Jun. 1994.
[12] C. Christodoulou and C. Pattichis, "Unsupervised pattern recognition for the classification of EMG signals," *IEEE Trans. Biomed. Eng.*, vol. 46, no. 2, pp. 169–178, Feb. 1999.
[13] S. Veeramachaneni and G. Nagy, "Style context with second-order statistics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 1, pp. 14–22, Jan. 2005.
[14] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: An application to face detection," in *Proc. IEEE Conf. CVPR*, 1997, pp. 130–136.
[15] K. Huang, H. Yang, I. King, and M. R. Lyu, "Learning large margin classifiers locally and globally," in *Proc. 21st Int. Conf. Mach. Learn.*, Banff, AB, Canada, 2004, pp. 401–408.
[16] G. R. G. Lanckriet, L. E. Ghaoui, C. Bhattacharyya, and M. I. Jordan, "A robust minimax approach to classification," *J. Mach. Learn. Res.*, vol. 3, no. 3, pp. 555–582, Mar. 2003.
[17] D. Wang, D. S. Yeung, and E. Tsang, "Sample reduction for SVMs via data structure analysis," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Waikoloa, HI, 2005, pp. 1030–1035.
[18] D. S. Yeung, D. Wang, W. Ng, E. Tsang, and X. Wang, "Structured large margin machine: Sensitive to data distribution," *Mach. Learn.,* submitted for publication.
[19] I. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means, spectral clustering and normalized cuts," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 551–556.
[20] A. K. Jain and R. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[21] J. H. Ward, "Hierarchical grouping to optimize an objective function," *J. Amer. Statist. Assoc.*, vol. 58, no. 301, pp. 236–244, Mar. 1963.
[22] A. El-Hamdouchi and P. Willett, "Comparison of hierarchic agglomerative clustering methods of document retrieval," *Comput. J.*, vol. 32, no. 3, pp. 220–227, Jun. 1989.
[23] S. Everitt, S. Landau, and M. Leese, *Cluster Analysis*. London, U.K.: Hodder Arnold, 2001.
[24] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms," in *Proc. 16th IEEE Int. Conf. Tools AI*, 2004, pp. 576–584.
[25] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: The Johns Hopkins Univ. Press, 1983.
[26] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. Philadelphia, PA: SIAM, 1994.
[27] J. Sturm, "Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones," *Optim. Methods Softw.*, vol. 11/12, no. 1–4, pp. 625–653, Aug. 1999.
[28] E. D. Andersen and A. D. Andersen, "The MOSEK interior point optimizer for linear programming: An implementation of the homogeneous algorithm," in *High Performance Optimization*. Norwell, MA: Kluwer, 2001, pp. 197–232.
[29] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra Appl.*, vol. 284, no. 1–3, pp. 193–228, Nov. 1998.
[30] A. Ruiz and P. E. Lopez-de Teruel, "Nonlinear kernel-based statistical pattern analysis," *IEEE Trans. Neural Netw.*, vol. 12, no. 1, pp. 16–32, Jan. 2001.
[31] C. Blake, E. Keogh, and C. Merz. (2006, Jan.). *UCI Repository of Machine Learning Database*. [Online]. Available: http://www.ics.uci.edu/_mlearn/ML-Repository.html
[32] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 6, pp. 1145–1159, 1997.
[33] B. Schölkopf, A. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.
[34] O. Chapelle and V. Vapnik, "Model selection for support vector machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 230–236.
[35] B. Schölkopf, J. Shawe-Taylor, A. Smola, and R. Williamson, "Kernel dependent support vector error bounds," in *Proc. 9th Int. Conf. Artif. Neural Netw.*, 1999, pp. 304–309.

Defeng Wang (S'03) received the B.Eng. degree in computer applications from Jilin University, Changchun, China, in 2000, and the M.Eng. degree in computer applications from Xidian University, Shaanxi, China, in 2003. He is currently working toward the Ph.D. degree at Hong Kong Polytechnic University, Hong Kong.

His recent research is focused on kernel methods, large-margin learning, and their applications in computational life science.

Daniel S. Yeung (M'89–SM'99–F'04) received the Ph.D. degree in applied mathematics from Case Western Reserve University, Cleveland, OH, in 1974.

He was an Assistant Professor of mathematics and computer science at the Rochester Institute of Technology, a Research Scientist with the General Electric Corporate Research Center, and a System Integration Engineer with TRW. He was the Chairman of the Department of Computing, The Hong Kong Polytechnic University, Kowloon. He is currently a Chair Professor at The Hong Kong Polytechnic University. He leads a group of researchers in Hong Kong and China who are actively engaged in research on computational intelligence and data mining. His current research interests include neural-network sensitivity analysis, data mining, Chinese computing, and fuzzy systems.

Dr. Yeung was the President of the IEEE Hong Kong Computer Chapter, and an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS and IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B. He is a member of the Board of Governor for the IEEE SMC Society, and has been elected the Vice President for Technical Activities for the same society. He served as the General Co-Chair of the 2002–2004 International Conference on Machine Learning and Cybernetics held annually in China, and a keynote speaker for the same conference. His IEEE Fellow citation makes reference to his "contribution in the area of sensitivity analysis of neural networks and fuzzy expert systems."

Eric C. C. Tsang (M'04–A'04) received the B.Sc. degree in computer studies from the City University of Hong Kong, Kowloon, in 1990, and the Ph.D. degree in computing from the Hong Kong Polytechnic University, Kowloon, in 1996.

He is currently an Assistant Professor in the Department of Computing, Hong Kong Polytechnic University. His main research interests are in the area of fuzzy expert systems, fuzzy neural networks, machine learning, genetic algorithms, fuzzy support vector machines, and multiple classifier systems.