An Efficient Approach for Outlier Detection with Imperfect Data Labels

Bo Liu, Yanshan Xiao, Philip S. Yu, Zhifeng Hao, and Longbing Cao

Abstract—The task of outlier detection is to identify data objects that are markedly different from or inconsistent with the normal set of data. Most existing solutions typically build a model using the normal data and identify outliers that do not fit the represented model very well. However, in addition to normal data, there also exist limited negative examples or outliers in many applications, and data may be corrupted such that the outlier detection data is imperfectly labeled. These make outlier detection far more difficult than the traditional ones. This paper presents a novel outlier detection approach to address data with imperfect labels and incorporate limited abnormal examples into learning. To deal with data with imperfect labels, we introduce likelihood values for each input data which denote the degree of membership of an example toward the normal and abnormal classes respectively. Our proposed approach works in two steps. In the first step, we generate a pseudo training dataset by computing likelihood values of each example based on its local behavior. We present kernel k-means clustering method and kernel LOF-based method to compute the likelihood values. In the second step, we incorporate the generated likelihood values and limited abnormal examples into SVDD-based learning framework to build a more accurate classifier for global outlier detection. By integrating local and global outlier detection, our proposed method explicitly handles data with imperfect labels and enhances the performance of outlier detection. Extensive experiments on real life datasets have demonstrated that our proposed approaches can achieve a better tradeoff between detection rate and false alarm rate as compared to state-of-the-art outlier detection approaches.

Index Terms—Outlier detection, data of uncertainty

1 Introduction

Outlier detection has attracted increasing attention in machine learning, data mining and and statistics literature. Outliers always refer to the data objects that are markedly different from or inconsistent with the normal existing data [1], [2]. A well-known definition of "outlier" is given in [3]: "an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism," which gives the general idea of an outlier and motivates many anomaly detection methods [1], [4]. Practically, outlier detection has been found in wide-ranging applications from fraud detection for credit cards, insurance or health care, intrusion detection for cyber-security, fault detection in safety critical systems, to military surveillance [1].

- *B. Liu is with the Department of Automation, Guangdong University of Technology, Guangzhou 510006, China. E-mail: csbliu@gmail.com.*
- *Y. Xiao and Z. Hao is with the Department of Computer Science, Guangdong University of Technology, Guangzhou 510006, China. E-mail: xiaoyanshan@gmail.com; mazfhao@scut.edu.cn.*
- *P. S. Yu is with the Department of Computer Science, University of Illinois at Chicago, Chicago, IL 60607, and with the Department of Computer Science, King Abdulaziz University Jeddah, Saudi Arabia. E-mail: psyu@uic.edu.*
- *L. Cao is with the Faculty of Engineering and Information Technology, University of Technology, Sydney, NSW 2007, Australia. E-mail: lbcao@it.uts.edu.au.*

Manuscript received 2 Jan. 2013; revised 24 Apr. 2013; accepted 22 May 2013. Date of publication 25 June 2013; date of current version 9 July 2014. Recommended for acceptance by X. He.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier 10.1109/TKDE.2013.108

Many outlier detection methods have been proposed to detect outliers from existing normal data. In general, the previous work on outlier detection can be broadly classified into distribution (statistical)-based, clustering-based, density-based and model-based approaches [5]–[8], all of them with long history. In the model-based approaches [8], they typically use a predictive model to characterize the normal data and then detect outliers as deviations from the model. In this category, the support vector data description (SVDD) [9], [10] has been demonstrated to be capable of detecting outliers in various application domains. In SVDD, a hyper-sphere is constructed to enclose most of the normal example with minimum sphere. The learned hyper-sphere is then utilized as a classifier to separate a test data into normal examples or outliers.

Though much progress has been done in support vector data description for outlier detection, most of the existing works on outlier detection always assume that input training data are perfectly labeled for building the outlier detection model or classifier. However, we may collect the data with imperfect labels due to noise or data of uncertainty [11], [12]. For examples, sensor networks typically generate a large amount of data subject to sampling errors or instrument imperfections. Thus, a normal example may behave like an outlier, even though the example itself may not be an outlier. These kind of uncertain data information might introduce labeling imperfections or errors into the training data, which further limits the accuracy of subsequent outlier detection. Therefore, it is necessary to develop outlier detection algorithms to handle imperfectly labeled data.

In addition, another important observation is that, negative examples or outliers, although very few, do exist in many applications. For example, in the network intrusion domain, in addition to extensive data about the normal traffic conditions in the network, there also exist a small number of cyber attacks that can be collected to facilitate outlier detection. Although these outliers are not sufficient for constructing a binary classifier, they can be incorporated into the training process to refine the decision boundary around the normal data for outlier detection.

In order to handle outlier detection with imperfect labels, we propose a novel approach to outlier detection by generalizing the support vector data description learning framework on imperfectly labeled training dataset. We associate each example in the training dataset not only with a class label but also likelihood values which denotes the degree of membership towards the positive and negative classes. We then incorporate the few labeled negative examples and the generated likelihood values into the learning phase of SVDD to build a more accurate classifier. The main contribution of our work can be summarized as follows.

1) We put forward two likelihood models, called single likelihood model and bi-likelihood model. In the single likelihood model, each input data is associated with one likelihood value which denotes the degree of membership towards its own class label. In the bi-likelihood model, each sample has two likelihood values which denote the degree of membership towards positive and negative class labels respectively. Based on the two likelihood models, we generate pseudo training datasets by computing likelihood values based on the local data behavior in the feature space. We put forward two methods based on the *k*-means clustering [1] and local outlier factor (LOF) [6] approaches respectively, to generate the likelihood values, which are called kernel *k*-means clustering-based method and kernel LOF-based method respectively. After that, we obtain two pseudo training sets for the two likelihood models respectively, in which each sample has likelihood values.
2) In the second step, we construct two global classifiers for outlier detection by generalizing the SVDD-based learning process based on the two likelihood models. The developed model derived from single likelihood model is called soft-SVDD. Another classifier related with bi-likelihood model is called bi-soft-SVDD. For both approaches, we incorporate the generated likelihood values of each sample and limited negative examples into the learning of support vector data description phase to build accurate outlier detection classifiers. In the process, each sample makes different contribution to the learning of the outlier detection decision boundary based on their likelihood values. By integrating local and global outlier detection, our proposed approaches explicitly handle the input data with imperfect labels and include a few labeled outliers into learning.
3) We conduct extensive experiments on real life datasets to investigate the performance of our proposed approaches. The results show that our proposed approaches can offer a better tradeoff between detection rate and false alarm rate and are less sensitive to noise in comparison of the state-of-the-art outlier detection algorithms.

Compared with the previous work on outlier detection, such as Artificial Immune System (AIS) [13], [14], most of them did not explicitly cope with the problem of both outlier detection with very few labeled negative examples and outlier detection on data with imperfect labels. Our proposed approaches first capture local data information by generating likelihood values for input examples, and then incorporate such information into support vector data description framework to build a more accurate outlier detection classifier.

The rest of the paper is organized as follows. Section 2 discusses previous work related to our outlier detection problem. Section 3 presents our proposed approached, called soft-SVDD and bi-soft-SVDD, to outlier detection in detail. Section 4 reports extensive experimental results on real-world datasets. Section 5 concludes the paper and discusses possible directions for future work.

2 Related Work

In this section, we discuss previous work related to our study. Since we focus on outlier detection with limited labeled outliers and data with imperfect labels, we briefly review previous work on outlier detection in section 2.1, and discuss another branch of related work on learning from imbalanced data in section 2.2. Finally, we briefly review support vector data description in section 2.3.

2.1 Outlier Detection

In the past, many outlier detection methods have been proposed [1]. Typically, these existing approaches can be divided into four categories: distribution (statistical)-based clustering-based, density-based and model-based approaches [1], [15]. Statistical approaches [16]–[18] assume that the data follows some standard or predetermined distributions, and this type of approach aims to find the outliers which deviate form such distributions. The methods in this category always assume the normal example follow a certain of data distribution. Nevertheless, we can not always have this kind of priori data distribution knowledge in practice, especially for high dimensional real data sets. [15].

For clustering-based approaches [7], [19], [20], they always conduct clustering-based techniques on the samples of data to characterize the local data behavior. In general, the sub-clusters contain significantly less data points than other clusters, are considered as outliers. For example, clustering techniques has been used to find anomaly in the intrusion detection domain [19]. In the work of [20], the clustering techniques iterative detect outliers to multi-dimensional data analysis in subspace. Since clustering-based approaches are unsupervised without requiring any labeled training data, the performance of unsupervised outlier detection is limited.

In addition, density-based approaches [6], [21]–[25] has been proposed. One of the representatives of this type of approaches are local outlier factor (LOF) and variants [6]. Based on the local density of each data instance, the LOF determines the degree of outlierness, which provides suspicious ranking scores for all samples. The most important property of the LOF is the ability to estimate local data structure via density estimation. The advantage of these approaches is that they do not need to make any assumption for the generative distribution of the data. However, these approaches incur a high computational complexity in the testing phase, since they have to calculate the distance between each test instance and all the other instances to compute nearest neighbors.

Besides the above work, model-based outlier detection approaches have been proposed [9], [10], [26]. Among them, support vector data description (SVDD) [9], [10] has been demonstrated empirically to be capable of detecting outliers in various domains. SVDD conducts a small sphere around the normal data and utilizes the constructed sphere to detect an unknown sample as normal or outlier. The most attractive feature of SVDD is that it can transform the original data into a feature space via kernel function and effectively detect global outliers for high-dimensional data. However, its performance is sensitive to the noise involved in the input data.

Depending on the availability of a training dataset, outlier detection techniques described above operate in two different modes: supervised and unsupervised modes. Among the four types of outlier detection approaches, distribution-based approaches and model-based approaches fall into the category of supervised outlier detection, which assumes the availability of a training dataset that has labeled instances for normal class (as well as anomaly class sometimes). In addition, several techniques [27]–[29] have been proposed that inject artificial anomalies into a normal dataset to obtain a labeled training data set. In addition, the work of [30] presents a new method to detect outliers by utilizing the instability of the output of a classifier built on bootstrapped training data.

Despite much progress on outlier detection, most of the previous work did not explicitly cope with the problem of outlier detection with very few labeled negative examples and data with imperfect label as well. Our proposed approaches capture local data information by generating the likelihood values of each input example towards the positive and negative classes respectively. Such information is then incorporated into the generalized support vector data description framework to enhance a global classifier for outlier detection.

The work in the paper has difference from our previous work about outlier detection [31]. First, the work in [31], called uncertain-SVDD (U-SVDD) here, addresses the outlier detection only using normal data without taking the outlier/negative examples into account. Second, U-SVDD only calculates the degree of membership of an example towards the normal example and takes single membership into learning phase. However, the work in this paper addresses the problem of outlier detection with a the few labeled negative examples, and takes data with imperfect labels into account. Based on the problem, we put forward single likelihood model and bi-likelihood model to assign likelihood values to each examples based on their local behaviors. For single likelihood model, examples including positive and negative classes are assigned likelihood values denoting the degree of membership towards their own class labels. For bi-likelihood model, each example is not only with a class label but also bi-likelihood values which denote the degree of membership towards the positive and negative classes respectively. Based on two likelihood models, we put forward soft-SVDD and bi-soft-SVDD approaches to incorporate the likelihood values together negative examples into SVDD-based learning phase. Therefore, the optimization model (7) called soft-SVDD, and model (12) called bi-soft-SVDD are completely different from the optimization problem (15) in [31]. In addition, the experiments in section 4 have shown that our proposed outlier detection approaches perform better than U-SVDD by incorporating few number of negative examples into the learning phase.

2.2 Difference from Imbalanced Data Classification

The outlier detection problem that we consider in this paper is also related to the problem of imbalanced data classification [32], in which outliers corresponding to the negative class are extremely small in proportion as compared to the normal data corresponding to the positive class.

We briefly review the research on imbalanced data [32]–[34] as follows. In general, previous work on imbalanced data classification falls into two main categories. The first category attempts to modify the class distribution of training data before applying any learning algorithms [35]. This is usually done by over-sampling, which replicates the data in the minority class, or under-sampling, which throws away part of the data in the majority class. The second category focuses on making a particular classifier learner cost sensitive, by setting the false positive and false negative costs very differently and incorporating the cost factors into the learning process [32]. Representative methods include cost-sensitive decision trees [36] and cost-sensitive SVMs [37]–[40]. In cost-sensitive SVMs, the cost factors of two classes are set differently so that the cost factors can affect the decision boundary. When imbalanced data are present, researchers have argued for the use of ranking-based metrics, such as the ROC curve and the area under ROC curve (AUC) [41] instead of using accuracy.

The difference between imbalanced data classification and our outlier detection problem is that: in imbalanced data classification, the examples from one or more minority classes are often self-similar, potentially forming compact clusters, while in outlier detection, the outliers are typically scattered around normal data so that the distribution of the negative class cannot be well represented by the very few negative training examples. To solve our problem, we can exploit cost-sensitive learning algorithms, but the false positive and false negative costs are usually unknown to us in real life applications. Therefore, we exploit a novel one-class classification method for outlier detection, which aims at building decision boundary around the normal data, and utilizes the few negative examples to refine the boundary to build an outlier detection classifier.

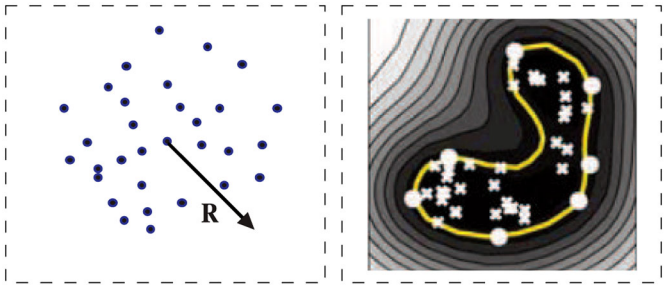


Fig. 1. (a) Illustration of SVDD hyper-sphere in feature space. (b) Illustration of SVDD decision boundary in input space.

2.3 Support Vector Data Description

The support vector data description (SVDD) [9] has been proposed for one-class classification learning. Given a set of target data $\{\mathbf{x}_i\}, i = 1, \ldots, l$, where $\mathbf{x}_i \in R^m$, the basic idea of SVDD is to find a minimum hyper-sphere that contains most of target data in the feature space, as illustrated in Fig. 1(a):

$$\begin{aligned} \min \;\; & F(R, \mathbf{o}, \xi_i) = R^2 + C\sum_{i=1}^{l}\xi_i, \\ s.t. \;\; & \| \phi(\mathbf{x}_i) - \mathbf{o} \|^2 \le R^2 + \xi_i, \\ & \xi_i \ge 0, \end{aligned} \tag{1}$$

where $\phi(.)$ is a mapping function which maps the input data from input space into a feature space, and $\phi(\mathbf{x}_i)$ is the image of $\mathbf{x}_i$ in the feature space, ξ_i are slack variables to allow some data points to lie outside the sphere, and $C > 0$ controls the tradeoff between the volume of the sphere and the number of errors. $\sum_{i=1}^{l}\xi_i$ is the penalty for misclassified samples.

By introducing Lagrange multipliers α_i, the optimization problem (1) is transformed into:

$$\begin{aligned} \max \;\; & \sum_{i=1}^{l}\alpha_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i=1}^{l}\sum_{k=1}^{l}\alpha_i\alpha_k K(\mathbf{x}_i, \mathbf{x}_k) \\ s.t. \;\; & 0 \le \alpha_i \le C, \\ & \sum_i \alpha_i = 1, \end{aligned} \tag{2}$$

in which kernel function $K(,.,)$ is utilized to calculate the inner pairwise product of two vector $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$, that is $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)$. The samples with $\alpha_i > 0$ are support vectors (SVs). For a test point $\mathbf{x}$, it is classified as normal data when this distance is less than or equal to the radius R. Otherwise, it is flagged as an outlier.

The most attractive feature of SVDD is that it can transform the input data into a feature space and detect global outliers effectively. As illustrated in Fig. 1(b), the hyper-sphere in the feature space responds to a decent decision boundary in input space. However, the performance of SVDD is sensitive to the noise involved in the input data. Our proposed method generalizes SVDD to incorporate the likelihood values of samples towards to positive and negative classes, which mitigates the effect of noise on outlier detection.

3 Our Proposed Approach

In this section, we provide a detailed description about our proposed approaches to outlier detection. Given a set of training data S which consists of l normal examples and a small amount of n outlier (or abnormal) examples, the objective is to build a classifier using both normal and abnormal training data and the classifier is thereafter applied to classify unseen test data. However, subject to sampling errors or device imperfections, an normal example may behave like an outlier, even though the example itself may not be an outlier. Such error factors might result in an imperfectly labeled training data, which makes the subsequent outlier detection become grossly inaccurate.

To deal with this problem, we put forward two likelihood models as follows.

Single likelihood model: In the model, we associate each input data with a likelihood value $(\mathbf{x}_i, m(\mathbf{x}_i))$, which indicates degree of membership of an example towards its own class label.

Bi-likelihood model: In the model, each sample is associate with bi-likelihood values, denoted as $(\mathbf{x}_i, m^t(\mathbf{x}_i), m^n(\mathbf{x}_i))$, in which $m^t(\mathbf{x}_i)$ and $m^n(\mathbf{x}_i)$ indicate the degree of an input data $\mathbf{x}_i$ belonging to the positive class and negative class respectively.

The main difference of two models is that, single likelihood model only considers the degree of membership towards its own class label; while bi-likelihood model includes the degree of membership towards its own class and the opposite class.

Such likelihood values information is thereafter incorporated into the construction of a global classifier for outlier detection. Based on this, our proposed approaches work in two steps as follows:

- In the first step, for each likelihood model, we generate a *pseudo training dataset* by computing likelihood values for each input data based on local data behavior in the feature space.
- In the second step, we put forward soft-SVDD and bi-soft-SVDD for single likelihood model and bi-likelihood model respectively, by using both normal and abnormal examples as well as the generated likelihood values.

In the following, we describe the two steps in detail.

3.1 Likelihood Values Generation

The main task of this step is to create a pseudo training dataset by computing likelihood values for each input data. For the single likelihood model, the generated pseudo training data consists of two parts for the l normal examples and n abnormal examples as follows.

$$\begin{aligned} & (\mathbf{x}_1, m^t(\mathbf{x}_1)), \ldots, (\mathbf{x}_l, m^t(\mathbf{x}_l)), (\mathbf{x}_{l+1}, m^n(\mathbf{x}_{l+1})), \ldots, \\ & (\mathbf{x}_{l+n}, m^n(\mathbf{x}_{l+n})), \end{aligned}$$

in which $m^t(\mathbf{x}_i)$ and $m^n(\mathbf{x}_i)$ indicate the likelihood of example $\mathbf{x}_i$ belonging to the the normal class and the abnormal, respectively.

Similarly, the generated pseudo training data for bi-likelihood model is:

$$(\mathbf{x}_1, m^t(\mathbf{x}_1), m^n(\mathbf{x}_1)), \ldots, (\mathbf{x}_l, m^t(\mathbf{x}_l), m^n(\mathbf{x}_l)), (\mathbf{x}_{l+1}, m^t(\mathbf{x}_{l+1}), m^n(\mathbf{x}_{l+1})), \ldots, (\mathbf{x}_{l+n}, m^t(\mathbf{x}_{l+n}), m^n(\mathbf{x}_{l+n})),$$

For each likelihood model, we propose two different schemes to compute likelihood values for each input data, which are inspired by the clustering-based [7] and density-based [6] approaches to outlier detection. The basic idea of both schemes is to capture the local data uncertainty by examining the relative distances of each input data to its local neighbors in the feature space.

For both likelihood models, the likelihood values are generated as follows.

3.1.1 Kernel K-Means Clustering-Based Method

We adopt the kernel k-means clustering algorithm to generate likelihood values for each input data. In kernel-based method, a nonlinear mapping function $\phi(.)$ maps the input samples into a feature space.Kernel k-means clustering minimizes the following objective function:

$$J = \sum_{i=1}^{k} \sum_{j=1}^{l+n} \|\phi(\mathbf{x}_j) - \phi(\mathbf{v}_i)\|^2, \tag{3}$$

where k is the number of clusters and $\mathbf{v}_i$ is the cluster center of the i^{th} cluster.

By solving this optimization problem, k-means clustering returns a set of local clusters, in which data samples belonging to a same cluster are more similar to each other. Intuitively, for a data sample, if most of data samples in the same cluster are normal, it would have a high probability of being normal, and if there is an outlying point that doe not belong to any cluster, it would have a high probability of being an outlier. Therefore, we calculate the likelihood values for single likelihood model and bi-likelihood model as follows. For a given cluster j, assume there exist l_j^p normal examples and l_j^n negative examples.

For the single likelihood model, the likelihood value of a normal example $\mathbf{x}_t$ belonging to the normal class is calculated $m^t(\mathbf{x}_t) = l_j^p/(l_j^p + l_j^n)$. Similarly, the likelihood value of an abnormal example $\mathbf{x}_k$ belonging to the negative class is computed as $m^n(\mathbf{x}_k) = l_j^n/(l_j^p + l_j^n)$.

For the bi-likelihood model, likelihood values of an example towards the normal and abnormal classes are calculated as $m^t(\mathbf{x}_t) = l_j^p/(l_j^p + l_j^n)$ and $m^n(\mathbf{x}_t) = l_j^n/(l_j^p + l_j^n)$ respectively.

Based on the kernel k-means clustering-based method, if a cluster only contains normal examples, the $m^t(\mathbf{x}_i)$ of each sample in the cluster equals to 1; while their corresponding $m^n(\mathbf{x}_i)$ is equivalent to 0. Therefore, the likelihood values generation method considers the local data information of each sample. The advantage of kernel k-means is that it can partition the dataset into a set of local clusters that are nonlinearly separable in the input space. However, the main limitation is that it does not work well on datasets with varying densities by using a global distance function, which causes the generated likelihood values to be inaccurate.

3.1.2 Kernel LOF-Based Method

To cope with datasets with varying densities, we propose a local density-based method to compute likelihood values for each input data. Inspired by the LOF algorithm [6], the basic idea is to examine the relative distance of a point to its local neighbors in feature space. More specifically, we extend the original LOF into the kernel space by using kernel function and generate the likelihood values in the kernel space instead of the input space.

For each point $\mathbf{x}_i$, we first compute its local reachability density, which is the average reachability distance based on the k-nearest neighbors of $\mathbf{x}_i$.

$$lrd_k(\mathbf{x}_i) = \frac{1}{k} \sum_{\mathbf{x}_j \in N_k(\mathbf{x}_i)} \text{reach-dist}_k(\mathbf{x}_i, \mathbf{x}_j), \tag{4}$$

where $N_k(\mathbf{x}_i)$ is a set of k-nearest neighbors of point $\mathbf{x}_i$. Here, reach-dist$_k(\mathbf{x}_i, \mathbf{x}_j)$ denotes the reachability distance of object $\mathbf{x}_i$ with respect to object $\mathbf{x}_j$ in the feature space, which is defined as reach-dist$_k(\mathbf{x}_i, \mathbf{x}_j) = \max\{\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|, \max_{\mathbf{x}' \in N_k(\mathbf{x}_i)} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}')\|\}$. The Interested readers please refer to [6] for detailed definitions. By considering the definition of reach-dist$_k(\mathbf{x}_i, \mathbf{x}_j)$, Equation (4) is simplified as

$$lrd_k(\mathbf{x}_i) = \max_{\mathbf{x}' \in N_k(\mathbf{x}_i)} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}')\|. \tag{5}$$

After the local reachability density $lrd_k(\mathbf{x}_i)$ is computed, for the point $\mathbf{x}_i$, we find its lrd-neighborhood $N_{lrd}(\mathbf{x}_i) = \{\mathbf{x}_j \in D \mid \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 \leq lrd_k(\mathbf{x}_i)\}$. The distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ in the feature space is computed as

$$\begin{aligned} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 &= (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)) \cdot (\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)) \\ &= \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i) + \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_j) - 2\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \\ &= K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned} \tag{6}$$

For a sample, suppose that there exist l_t examples out of $|N_{lrd}(\mathbf{x}_i)|$ nearest neighbors belonging to the positive class. $|N_{lrd}(\mathbf{x}_i)|$ denotes the number of nearest neighbors in the lrd-neighborhood. Let $l_n = |N_{lrd}(\mathbf{x}_i)| - l_t$.

For single likelihood model, the likelihood value of a normal example $\mathbf{x}_t$ belonging to the normal class is calculated $m^t(\mathbf{x}_t) = l_t/|N_{lrd}(\mathbf{x}_i)|$. Similarly, the likelihood value of an abnormal example $\mathbf{x}_n$ belonging to the negative class is computed $m^n(\mathbf{x}_k) = l_n/|N_{lrd}(\mathbf{x}_i)|$.

For bi-likelihood model, the likelihood value of $\mathbf{x}_t$ towards the positive class and negative class are calculated $m^t(\mathbf{x}_t) = l_t/|N_{lrd}(\mathbf{x}_i)|$ and $m^n(\mathbf{x}_k) = l_n/|N_{lrd}(\mathbf{x}_i)|$.

Based on the above method, the likelihood value $m^t(\mathbf{x}_i)$ of sample $\mathbf{x}_i$ equals to 1 if there is not any abnormal examples in its lrd-neighborhood $N_{lrd}(\mathbf{x}_i)$ and the corresponding $m^n(\mathbf{x}_i)$ is equivalent to 0. In this method, we can calculate the likelihood values based on the local behavior of each sample and cope with the dataset with varying densities.

3.2 Constructing SVDD-Based Classifiers

Above, for the two likelihood models, we put forward kernel k-means clustering-based and kernel LOF-based method to generate likelihood values. We then develop soft-SVDD and bi-soft-SVDD for the single likelihood model and bi-likelihood model respectively. Both developed methods include normal and abnormal data in the learning.

However, soft-SVDD only incorporates single likelihood value of an example towards its own class label in the learning; while bi-soft-SVDD takes bi-likelihood values of an examples towards the positive and negative class labels in the training. The basic idea of our methods are to enclose normal examples inside the sphere and exclude the abnormal examples outside of the sphere and consider the likelihood values in the learning. Below, we present two developed approaches.

3.2.1 Constructing Soft-SVDD Classifiers

For the single likelihood model, we put positive examples into set S_p, in which examples only have $m^t(\mathbf{x}_i)$, and put negative examples into set S_n, where examples are only associated with $m^n(\mathbf{x}_j)$. Since the membership functions $m^t(\mathbf{x}_i)$ and $m^n(\mathbf{x}_j)$ indicate the degree of the membership of data example $\mathbf{x}_i$ toward normal class and negative class, the solution to soft-SVDD can be achieved by solving the following optimization problem:

$$\begin{aligned} \min \;\; & F = R^2 + C_1 \sum m^t(\mathbf{x}_i)\xi_i + C_2 m^n(\mathbf{x}_j)\xi_j \\ s.t. \;\; & \| \mathbf{x}_i - \mathbf{o} \|^2 \le R^2 + \xi_i, \mathbf{x}_i \in S_p \\ & \| \mathbf{x}_j - \mathbf{o} \|^2 \ge R^2 - \xi_j, \mathbf{x}_j \in S_n \\ & \xi_i \ge 0, \;\; \xi_j \ge 0, \end{aligned} \tag{7}$$

Above, Parameters C_1 and C_2 control the tradeoff between the sphere volume and the errors. Parameters ξ_i are ξ_j are defined as a measure of error, as in SVDD. The terms $m^t(\mathbf{x}_i)\xi_i$ and $m^n(\mathbf{x}_j)\xi_j$ can be therefore considered as a measure of error with different weighing factors. Note that a smaller value of $m^t(\mathbf{x}_i)$ could reduce the effect of the parameter ξ_i in Equation (7), such that the corresponding data example $\mathbf{x}_i$ becomes less significant in the training.

Problem Solution

In order to resolve the optimization problem (7), we introduce the Lagrange method [42] and then have Theorem 1.

Theorem 1. *The solution of problem (7) can be resolved by the optimization problem (8):*

$$\begin{aligned} \max \; & \sum_{i=1}^{l+n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}_i) - \sum_{i=1}^{l+n}\sum_{j=1}^{l+n} \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ s.t. \; & 0 \le \alpha_i \le C_i^m \quad i = 1, 2, \ldots, l+n, \\ & \sum_{i=1}^{l+n} \alpha_i = 1, \end{aligned} \tag{8}$$

in which $\alpha_i \ge 0$, $\alpha_j \ge 0$ are Lagrange multipliers, $C_i^m = C_1 m^t(\boldsymbol{x}_i)(i = 1, 2, \ldots, l)$ and $C_i^m = C_2 m^n(\boldsymbol{x}_i)(i = l+1, l+2, \ldots, l+n)$.

Proof. To solve the above optimization problem (7), we introduce Lagrange multipliers $\alpha_i^t \ge 0$, $\alpha_j^n \ge 0$, $\beta_i^t \ge 0$, $\beta_j^n \ge 0$, and convert problem (7) into problem (9).

$$\begin{aligned} L = {} & R^2 + C_1 \sum m^t(\mathbf{x}_i)\xi_i + C_2 \sum m^n(\mathbf{x}_j)\xi_j \\ & - \sum \alpha_i^t (R^2 + \xi_i - \| \phi(\mathbf{x}_i) - \mathbf{o} \|^2) - \sum \beta_j^n \xi_j \\ & - \sum \beta_i^t \xi_i - \sum \alpha_j^n (\| \Phi(\mathbf{x}_j) - \mathbf{o} \|^2 - R^2 - \xi_j). \end{aligned} \tag{9}$$

Setting the partial derivatives of L with respect to $R, \mathbf{o}, \xi_i, \xi_j$ equal to zeros respectively, we can obtain

$$\begin{aligned} & \frac{\partial L}{\partial R} = 0 \longrightarrow \sum \alpha_i^t - \sum \alpha_j^n = 1, \\ & \frac{\partial L}{\partial \mathbf{o}} = 0 \longrightarrow \sum \alpha_i^t (\mathbf{o} - \phi(\mathbf{x}_i)) = \sum \alpha_j^n (\mathbf{o} - \phi(\mathbf{x}_j)), \\ & \frac{\partial L}{\partial \xi_i} = 0 \longrightarrow \alpha_i^t + \beta_i^t = C_1 m^t(\mathbf{x}_i), \\ & \frac{\partial L}{\partial \xi_j} = 0 \longrightarrow \alpha_j^n + \beta_j^n = C_2 m^n(\mathbf{x}_j). \end{aligned}$$

Replacing these into Equation (9), and set $\alpha_i = \alpha_i^t (i = 1, 2, \ldots, l)$, $\alpha_i = \alpha_i^n (i = l+1, l+2, \ldots, l+n)$, $C_i^m = C_1 m^t(\mathbf{x}_i)(i = 1, 2, \ldots, l)$ and $C_i^m = C_2 m^n(\mathbf{x}_i)(i = l+1, l+2, \ldots, l+n)$, we have optimization problem problem (8).

After solving the above dual problem, we obtain the Lagrange multipliers α_i $(1 \le i \le l+n)$, which gives the centroid of the minimum sphere as a linear combination of $\mathbf{x}_i$:

$$\mathbf{o} = \sum_{i=1}^{l+n} \alpha_i \phi(\mathbf{x}_i). \tag{10}$$

Above, we find only the patterns with $\alpha_i \ne 0$ construct the centroid of the minimum sphere, and these pattern are called support vectors.

Decision Boundary Construction

By applying Karush-Kuhn-Tucker conditions [42], we then obtain the radius R of the decision hyperplane. Assume $\mathbf{x}_u$ is one of the patterns lying on the surface of sphere, R can be calculated as follows:

$$\begin{aligned} R^2 = \|\mathbf{x}_u - \mathbf{o}\|^2 &= K(\mathbf{x}_u, \mathbf{x}_u) + K(\mathbf{o}, \mathbf{o}) - 2K(\mathbf{x}_u, \mathbf{o}) \\ &= K(\mathbf{x}_u, \mathbf{x}_u) + \sum_{i=1}^{l+n}\sum_{k=1}^{l+n} \alpha_i \alpha_k K(\mathbf{x}_i, \mathbf{x}_k) - 2\sum_{i=1}^{l+n} \alpha_i (\mathbf{x}_i, \mathbf{x}_u). \end{aligned}$$

To classify a test point $\mathbf{x}$, we just calculate its distance to the centroid of the hypersphere. If this distance is less than or equal to R, i.e.

$$\|\mathbf{x} - \mathbf{o}\|^2 \le R^2, \tag{11}$$

$\mathbf{x}$ is accepted as the normal data. Otherwise, it is detected as an outlier. □

3.2.2 Constructing Bi-Soft-SVDD Classifiers

For the bi-likelihood model, we derive the bi-soft-SVDD as follows.

First of all, based on the generated likelihood values, we split the datasets into three parts S_p, S_b and S_n for the sake of derivation. For the samples in S_p, the likelihood value towards the positive class equals to 1, that is $m^t(\mathbf{x}_i) = 1$ and $m^n(\mathbf{x}_i) = 0$, which means the sample $\mathbf{x}_i$ completely belongs to the positive class. For the samples in S_b; it has non-zero likelihood values towards the positive and negative classes at the same time, that is $m^t(\mathbf{x}_h) \ne 0$ and $m^n(\mathbf{x}_h) \ne 0$. For the samples in S_n, they completely belong to the negative class, that is $m^n(\mathbf{x}_j) = 1$ and $m^t(\mathbf{x}_j) = 0$.

Since the likelihood values $m^t(\mathbf{x}_i)$ and $m^n(\mathbf{x}_i)$ indicate the degree of membership of data example $\mathbf{x}_i$ towards the

positive and negative class respectively, the solution to bi-soft-SVDD can be extended from problem (1) by solving the following optimization problem:

$$\begin{aligned}\min\ & F = R^2 + C_1(\sum \xi_i + \sum m^t(\mathbf{x}_h)\xi_h) \\ & + C_2(\sum \xi_j + \sum m^n(\mathbf{x}_k)\xi_k) \\ s.t.\ & \| \phi(\mathbf{x}_i) - \mathbf{o} \|^2 \le R^2 + \xi_i, \quad \mathbf{x}_i \in S_p \\ & \| \phi(\mathbf{x}_h) - \mathbf{o} \|^2 \le R^2 + \xi_h, \quad \mathbf{x}_h \in S_b \\ & \| \phi(\mathbf{x}_k) - \mathbf{o} \|^2 \ge R^2 - \xi_k, \quad \mathbf{x}_k \in S_b \\ & \| \phi(\mathbf{x}_j) - \mathbf{o} \|^2 \ge R^2 - \xi_j, \quad \mathbf{x}_j \in S_n \\ & \xi_i \ge 0,\ \xi_h \ge 0,\ \xi_k \ge 0,\ \xi_j \ge 0, \end{aligned} \tag{12}$$

Above, parameters C_1, C_2 control the tradeoff between the sphere volume and the errors, which is the same function as C in optimization (1). Parameters ξ_i, ξ_j, ξ_h and ξ_k are defined as measure of error, the same as ξ_i in (1). The terms $m^t(\mathbf{x}_h)\xi_h$ and $m^n(\mathbf{x}_k)\xi_k$ can be therefore considered as measure of error with different weighing factors.

Problem Solution

In order to resolve the optimization problem (12), we introduce the Lagrange method [42] and then have Theorem 2 to resolve the problem as follows.

Theorem 2. *The solution of problem (12) can be resolved by the optimization problem (13)*

$$\begin{aligned}Max\ & \sum \alpha_i^t K(\boldsymbol{x}_i, \boldsymbol{x}_i) - \sum \alpha_j^n K(\boldsymbol{x}_j, \boldsymbol{x}_j) \\ & - \sum\sum \alpha_i^t \alpha_k^t K(\boldsymbol{x}_i, \boldsymbol{x}_k) + 2\sum\sum \alpha_i^t \alpha_j^n K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ & - \sum\sum \alpha_j^n \alpha_v^n K(\boldsymbol{x}_j, \boldsymbol{x}_v) \\ s.t\ & 0 \le \alpha_i^t \le m_i^t(\boldsymbol{x}_i) C_1, \\ & 0 \le \alpha_j^n \le m_i^n(\boldsymbol{x}_i) C_2, \\ & \sum \alpha_i^t - \sum \alpha_j^n = 1, \\ & \boldsymbol{x}_i, \boldsymbol{x}_k \in\ S_p \cup S_b,\ \boldsymbol{x}_j, \boldsymbol{x}_v \in S_b \cup S_n, \end{aligned} \tag{13}$$

in which $\alpha_i^t \ge 0$, $\alpha_j^n \ge 0$ are Lagrange multipliers.

Proof. In order to solve the optimization problem in (13), we introduce Lagrange multipliers $\alpha_i^t \ge 0$, $\alpha_h^b \ge 0$ $\alpha_j^n \ge 0$, $\alpha_k^b \ge 0$, $\beta_i^t \ge 0$, $\beta_h^b \ge 0$, $\beta_j^n \ge 0$, $\beta_k^b \ge 0$, and convert the problem (12) into the following problem (14):

$$\begin{aligned}L = & R^2 + C_1 \sum \xi_i + C_2 \sum \xi_j + C_1 \sum m^t(\mathbf{x}_h)\xi_h \\ & + C_2 \sum \xi_k m^n(\mathbf{x}_k) - \sum \beta_i^t \xi_i - \sum \beta_h^b \xi_h - \sum \beta_k^b \xi_k \\ & - \sum \beta_j^n \xi_j - \sum \alpha_i^t (R^2 + \xi_i - \| \phi(\mathbf{x}_i) - \mathbf{o} \|^2) \\ & - \sum \alpha_j^n (\| \phi(\mathbf{x}_j) - \mathbf{o} \|^2 - R^2 - \xi_j) \\ & - \sum \alpha_h^b (R^2 + \xi_h - \| \phi(\mathbf{x}_h) - \mathbf{o} \|^2) \\ & - \sum \alpha_k^b (\| \phi(\mathbf{x}_k) - \mathbf{o} \|^2 - R^2 - \xi_k). \end{aligned} \tag{14}$$

The parameters must satisfy that: $\frac{\partial L}{\partial R} = 0$, $\frac{\partial L}{\partial \mathbf{o}} = 0$, $\frac{\partial L}{\partial \xi_i} = 0$, $\frac{\partial L}{\partial \xi_j} = 0$, $\frac{\partial L}{\partial \xi_h} = 0$, $\frac{\partial L}{\partial \xi_k} = 0$, we then have the following conditions respectively:

$$\sum \alpha_i^t + \sum \alpha_h^b - \sum \alpha_k^b - \sum \alpha_j^n = 1, \tag{15}$$

$$\begin{aligned}\mathbf{o} = & \sum \alpha_i^t \phi(\mathbf{x}_i) + \sum \alpha_h^b \phi(\mathbf{x}_h) - \sum \alpha_k^b \phi(\mathbf{x}_k) \\ & - \sum \alpha_j^n \phi(\mathbf{x}_j), \end{aligned} \tag{16}$$

$$\alpha_i^t + \beta_i^t = C_1, \tag{17}$$

$$\alpha_j^n + \beta_j^n = C_2, \tag{18}$$

$$\alpha_h^b + \beta_h^b = m^t(\mathbf{x}_h) C_1, \tag{19}$$

$$\alpha_k^b + \beta_k^b = m^n(\mathbf{x}_k) C_2. \tag{20}$$

It is noted that if we substitute (16) into problem (14), it will be complicated. To simplify the process, we rewrite (15)-(20). First of all, let

$$\alpha_i^t = \begin{cases} \alpha_i^t, \ for\ \ \mathbf{x}_i \in S_p, \\ \alpha_h^b, \ for\ \ \mathbf{x}_h \in S_b, \end{cases} \quad \alpha_j^n = \begin{cases} \alpha_k^b \ for\ \ \mathbf{x}_k \in S_b, \\ \alpha_j^n \ for\ \ \mathbf{x}_j \in S_n. \end{cases} \tag{21}$$

then, (15) and (16) can be rewritten as

$$\sum \alpha_i^t - \sum \alpha_j^n = 1, \tag{22}$$

$$\mathbf{o} = \sum \alpha_i^t \phi(\mathbf{x}_i) - \sum \alpha_j^n \phi(\mathbf{x}_j), \tag{23}$$

in which $\mathbf{x}_i \in S_p \cup S_b$ and $\mathbf{x}_j \in S_n \cup S_b$. let

$$m_i^t(\mathbf{x}_i) = \begin{cases} 1 \qquad for \quad \mathbf{x}_i \in S_p, \\ m_i^t(\mathbf{x}_h)\ for \quad \mathbf{x}_h \in S_b, \end{cases} \tag{24}$$

$$m_j^n(\mathbf{x}_j) = \begin{cases} m_k^n(\mathbf{x}_k)\ for \quad \mathbf{x}_k \in S_b, \\ 1 \qquad for \quad \mathbf{x}_j \in S_n, \end{cases} \tag{25}$$

$$\beta_i^t(\mathbf{x}_i) = \begin{cases} \beta_i^t \ for \quad \mathbf{x}_i \in S_p, \\ \beta_h^b \ for \quad \mathbf{x}_h \in S_b, \end{cases} \tag{26}$$

$$\beta_j^n(\mathbf{x}_j) = \begin{cases} \beta_k^b, \ for \quad \mathbf{x}_k \in S_b \\ \beta_j^n \ \ for \quad \mathbf{x}_j \in S_n. \end{cases} \tag{27}$$

Then (17), (18), (19), and (20) are represented as

$$\alpha_i^t + \beta_i^t = m_i^t(\mathbf{x}_i) C_1, \quad for \quad \mathbf{x}_i \in S_p \cup S_b, \tag{28}$$

$$\alpha_j^n + \beta_j^n = m_j^n(\mathbf{x}_j) C_2, \quad for \quad \mathbf{x}_j \in S_n \cup S_b, \tag{29}$$

since $\beta_i^t \ge 0, \beta_j^n \ge 0$, then we have

$$0 \le \alpha_i^t \le m_i^t(\mathbf{x}_i) C_1, \quad for \quad \mathbf{x}_i \in S_p \cup S_b, \tag{30}$$

$$0 \le \alpha_j^n \le m_j^n(\mathbf{x}_j) C_2, \quad for \quad \mathbf{x}_j \in S_n \cup S_b. \tag{31}$$

Based on (23), the inner product of the centroid of the sphere is:

$$\begin{aligned}(\mathbf{o}, \mathbf{o}) = & \sum\sum \alpha_i^t \alpha_k^t K(\mathbf{x}_i, \mathbf{x}_k) - 2\sum\sum \alpha_i^t \alpha_j^n K(\mathbf{x}_i, \mathbf{x}_j) \\ & + \sum\sum \alpha_j^n \alpha_v^n K(\mathbf{x}_j, \mathbf{x}_v). \end{aligned} \tag{32}$$

Based on (22), (28), (29), we have

$$R^2 (1 - \sum \alpha_i^t + \sum \alpha_j^n) = 0 \tag{33}$$

$$C_1 \sum m^t(\mathbf{x}_i)\xi_i - \sum \beta_i^t \xi_i - \sum \alpha_i^t \xi_i = 0 \tag{34}$$

$$C_2 \sum m^n(\mathbf{x}_j)\xi_j - \sum \beta_j^n \xi_j - \sum \alpha_j^n \xi_j = 0 \tag{35}$$

TABLE 1
Confusion Matrix

		Actual Label	
		Target Class	Negative Class
Predicted Label	Target Class	True Positive (TP)	False Negative (FN)
	Negative Class	False Positive (FP)	True Negative (TN)

TABLE 2
Datasets Description

Dataset	Description	# of dataset	# of Features
Abalone	classes 1-8 vs rest	4177	10
Spambase	others vs spam	4601	57
thyroid	class 2 vs rest 3	3428	21
Waveform	class 0 vs rest	900	21
Satellite	Grey soil vs rest	4435	36
Delft pump 5x3	normal situations vs rest	1500	64
Diabetes	present vs rest	768	8
Segment	class 1 vs rest	2310	19
Letter	class 1 vs rest	6238	617
Arrhythmia	normal vs rest	420	278

substitute (23) into (14), and consider (33), (34), (35), we then have

$$L = \sum \alpha_i^t K(\mathbf{x}_i, \mathbf{x}_i) - \sum \alpha_j^n K(\mathbf{x}_j, \mathbf{x}_j) - (\mathbf{o}, \mathbf{o}) \quad (36)$$

substitute (32) into (36), we have the Theorem 2.

By solving the optimization problem (13), we can obtain the Lagrange multipliers $\alpha_i^t \geq 0$, $\alpha_j^n \geq 0$.

For the relationship between the location of the samples and their Lagrange multipliers α_i^t, we have the following analysis: For the normal examples $\mathbf{x}_i \in S_p$, for the problem (14), the Karush-Kuhn-Tucker conditions [42] satisfy that

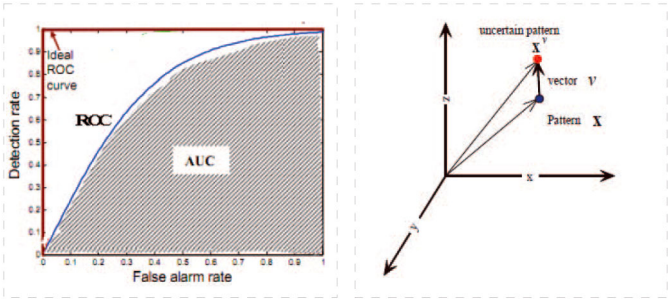
$$\beta_i^t \xi_i = 0 \quad for \quad \mathbf{x}_i \in S_p \quad (37)$$

$$\alpha_i^t (R^2 + \xi_i - \| \mathbf{x}_i - \mathbf{o} \|^2) = 0 \quad for \quad \mathbf{x}_i \in S_p \quad (38)$$

1) If $\mathbf{x}_i$ lies outside the sphere, $\xi_i > 0$ holds, and we have $\beta_i^t = 0$ according to (37), and then $\alpha_i^t = C_1$ from (17).
2) If $0 < \alpha_i^t < C_1$, from (17) and (37) $\beta_i^t \neq 0$ and $\xi_i = 0$; therefore, from the first constraint of (12), lie on the surface of the sphere.
3) For all patterns inside the sphere, we necessarily have $\alpha_i^T = 0$ from (38).

It is noted that, for the normal samples whose $\alpha_i^t \neq 0$ called support vectors (SVs), and the support vectors reside onside or outside of the hyper-sphere. Based on this, the centroid and radius of the hyper-sphere are denoted:

$$\begin{aligned} \mathbf{o} &= \sum \alpha_i^t \phi(\mathbf{x}_i) - \sum \alpha_j^n \phi(\mathbf{x}_j), \\ R^2 &= K(\mathbf{x}_u, \mathbf{x}_u) + \sum \sum \alpha_i^t \alpha_k^t K(\mathbf{x}_i, \mathbf{x}_k) \\ &\quad + \sum \sum \alpha_j^n \alpha_v^n K(\mathbf{x}_j, \mathbf{x}_v) - 2 \sum \sum \alpha_i^t \alpha_j^n K(\mathbf{x}_i, \mathbf{x}_j) \\ &\quad - 2 \sum \alpha_i^t K(\mathbf{x}_i, \mathbf{x}_u) + 2 \sum \alpha_j^n K(\mathbf{x}_j, \mathbf{x}_u) \end{aligned} \quad (39)$$



(a) (b)

Fig. 2. (a) Illustration of ROC curve and the AUC. (b) Illustration of the method used to add the noise to a data example: $\mathbf{x}$ is an original data example, $\mathbf{v}$ is a noise vector, $\mathbf{x^v}$ is the new data example with added noise. Here we have $\mathbf{x^v} = \mathbf{x} + \mathbf{v}$.

in which $\mathbf{x}_i, \mathbf{x}_k \in S_p \cup S_b, \mathbf{x}_j, \mathbf{x}_v \in S_n \cup S_b$, and $\mathbf{x}_u \in S_p$ is the labeled normal example whose $0 < \alpha_i^t < C_1$, that is it resides on the surface of the hyper-sphere. From above formulations, we know that it is only the support vectors that determine the centroid and radius of the hyper-sphere.

Decision Boundary Determination

After obtaining the centroid and radius of the hyper-sphere, we have the decision boundary of classifier. To classify a test sample $\mathbf{x}$, we calculate its distance to the centroid of the hyper-sphere. If the distance is less than or equals to R, i.e.

$$\begin{aligned} \|\mathbf{x} - \mathbf{o}\|^2 &= K(\mathbf{x}, \mathbf{x}) + \sum \sum \alpha_i^t \alpha_k^t K(\mathbf{x}_i, \mathbf{x}_k) \\ &\quad + \sum \sum \alpha_j^n \alpha_v^n K(\mathbf{x}_j, \mathbf{x}_v) - 2 \sum \sum \alpha_i^t \alpha_j^n \\ &\quad K(\mathbf{x}_i, \mathbf{x}_j) - 2 \sum \alpha_i^t K(\mathbf{x}_i, \mathbf{x}) \\ &\quad + 2 \sum \alpha_j^n K(\mathbf{x}_j, \mathbf{x}) \leq R^2, \end{aligned} \quad (40)$$

$\mathbf{x}$ is accepted as the normal data. Otherwise, it is classified as an outlier. □

3.2.3 Discussion

For the single likelihood model, it only considers the degree of membership towards its own class label. For the bi-likelihood model, it includes the degree of membership towards its own class and the opposite class.

In the likelihood values generation procedure, although we have $m^t(\mathbf{x}_i) + m^n(\mathbf{x}_i) = 1$ for the kernel K-Means clustering-based and kernel LOF-based methods, the two likelihood models have different contribution on the subsequent outlier detection classifiers construction. The soft-SVDD classifier based on single likelihood model, i.e. optimization problem (7), only considers the degree of membership towards its own class label. For the example $\mathbf{x}_i$ in the normal class, it only incorporates $m^t(\mathbf{x}_i)$ in the learning, but discarding $m^n(\mathbf{x}_i)$, i.e. the degree of membership towards the negative class, in the training. For the example $\mathbf{x}_j$ in the negative class, it only considers $m^n(\mathbf{x}_j)$ in the optimization problem, but discarding $m^t(\mathbf{x}_j)$, i.e. the degree of membership towards the normal class, in the training phase. However, the bi-soft-SVDD classifier built on bi-likelihood model incorporates the degree of membership towards its own class and the opposite class in the training, as shown in optimization problem (12). Compared with soft-SVDD, bi-soft-SVDD incorporates more data information in the training phase; as a result, bi-soft-SVDD delivers higher performance than soft-SVDD.

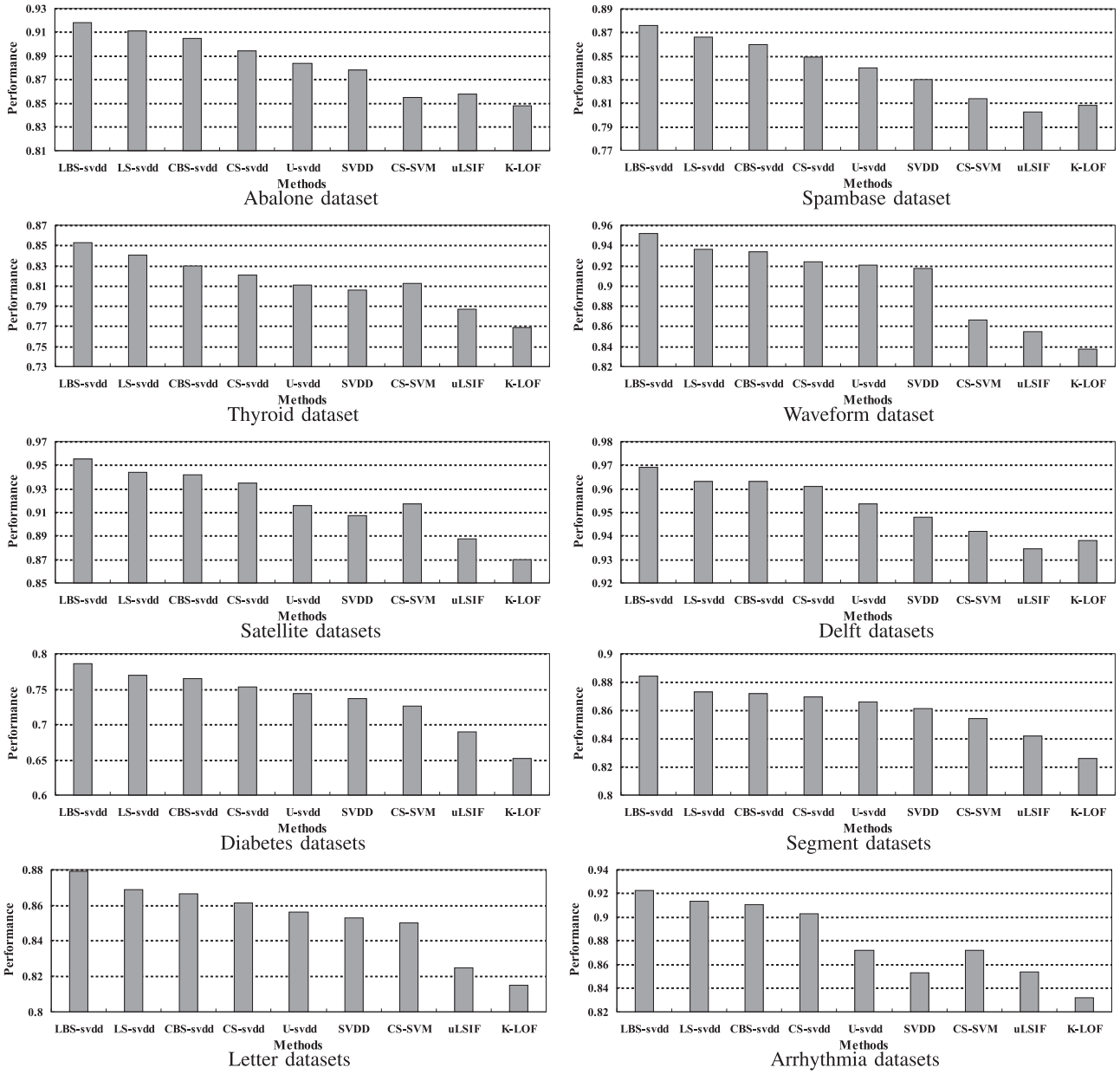


Fig. 3. Performance of outlier detection approaches on ten data sets.

4 Experimental Evaluation

In this section, we conduct extensive experiments to investigate the performance of our proposed approach on real life datasets. For all reported results, the test platform is a Dual 2.4GHz Intel Core2 T9600 laptop with 4GB RAM.

4.1 Baselines and Metrics

4.1.1 Baselines

For the single likelihood and bi-likelihood models, we put forward kernel k-means clustering-based and kernel LOF-based methods to generate likelihood values for them. After that, we develop soft-SVDD and bi-soft-SVDD methods to incorporate negative examples and likelihood values into learning.

We then have four variants of our proposed approaches, which are called *k*-means clustering-based soft-SVDD (CS-SVDD), LOF-based soft-SVDD (LS-SVDD), *k*-means clustering-based bi-soft-SVDD (CBS-SVDD) and LOF-based bi-soft-SVDD (LBS-SVDD) respectively. For comparison, another five state-of-the-art outlier detection algorithms are used as baselines.

1) The first one is the kernel-LOF algorithm, which generalizes the LOF algorithm [6] by computing the outlier factor in the feature space. This baseline is used to show the improvement of our proposed method over unsupervised outlier detection approach.
2) The second one is SVDD [9], which builds a one-class classifier solely based on the normal data. This baseline is used to test the ability of our proposed method over original SVDD classifier.
3) The third one is uncertain-SVDD (U-SVDD) [31], which assigns single membership towards normal data and constructs a classifier only based on the normal data.

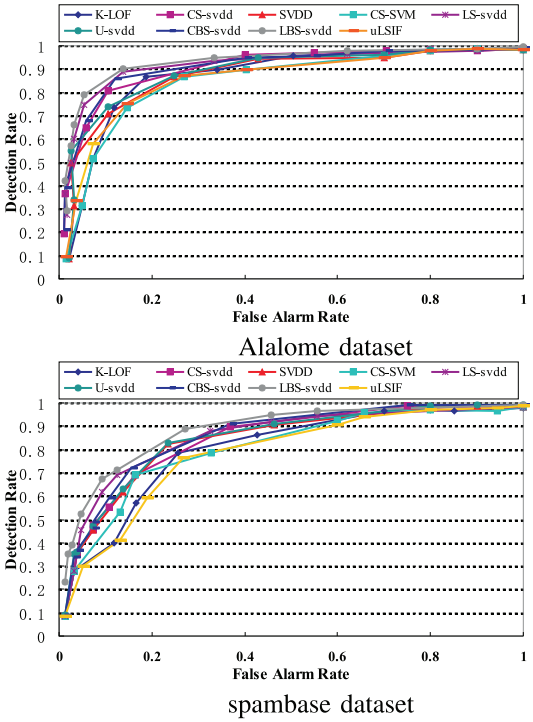


Fig. 4. Comparison of ROC curves with respect to the detection rate and false alarm rate.

4) The forth one is uLSIF [18], which uses the ratio of training and test data densities to determine outliers.
5) The fifth baseline is the cost-sensitive SVM (CS-SVM) [39] for imbalanced classification, which assigns different costs to the normal data and abnormal data so as to learn a binary classifier for outlier detection. This baseline is used to test the effectiveness of our proposed method when very few labeled negative examples are available for training.

4.1.2 Metrics

The performance of outlier detection algorithms can be evaluated based on two error rates: *detection rate* and *false alarm rate*. Detection rate gives information about the number of correctly identified outliers, while the false alarm rate reports the number of outliers misclassified as normal data records. Based on the confusion matrix in Table. 1. The detection rate and the false alarm are computed as follows: $Detection\ rate = TP/TP + FN$, $False\ alarm\ rate = FP/FP + TN$.

The ROC (receiver operating characteristic) curve represents the trade-off between the detection rate and the false alarm rate and is typically shown on a 2-D graph (Fig. 2(a), where false alarm rate and detection rate are plotted on x-axis, and y-axis respectively. In general, the area under the curve (AUC) is also used to measure the performance of outlier detection algorithm. The AUC of specific algorithm is defined as the surface area under its ROC curve, as illustrated in Fig. 2(a). The AUC for the ideal ROC curve is typically closer to one, while AUCs of ąřless than perfectąś outlier detection algorithms are less than 1. We also explicitly compute the AUC values [41] to compare the algorithms.

4.2 Datasets and Parameter Settings

In our experiments, we used 10 real life datasets that have been used earlier by other researchers for outlier detection [43], [44]. These datasets include Abalone, Spambase,thyroid, Waveform,Satellite, Delft pump, Diabetes, Segment, Letter recognition, Arrhythmia, which are available from UCI datasets [45] and [46]. The information of these datasets is listed in Table. 2. To perform outlier detection with very few abnormal data, we randomly selected 50% of positive data and a small number of abnormal data for training, such that 95 percent of the training data belong to the positive class and only 5 percent belong to the negative class. All the remaining data are used for testing.

For all the algorithms, the Gaussian RBF kernel was used in the experiments

$$K(\mathbf{x}, \mathbf{x}_i) = \exp(-\| \mathbf{x} - \mathbf{x}_i \|^2 / 2\sigma^2). \quad (41)$$

We use cross-validation on the training data to tune the parameters for LBS-SVDD, CBS-SVDD, LS-SVDD, CS-SVDD, CS-SVM, and SVDD. The parameter σ in the RBF kernel is searched in the range from 2^{-3} to 2^4. In addition, the parameter C in SVDD, as well as C_1, C_2 in LBS-SVDD, CBS-SVDD, LS-SVDD, CS-SVDD is selected from 2^0 to 2^4. All the reported AUC results are based on this setting.

For CBS-SVDD, CS-SVDD, the number of k in kernel k-means is varied from 2 to $\frac{l+n}{2}$ and obtain the optimal number of clusters k^* by minimizing the external criteria in [47]. For LBS-SVDD and LS-SVDD, we set the number of nearest neighbors k used for computing confidence values to the number of negative samples in the training set. For kernel LOF, we follow the experimental setting in [6] to compute the maximum LOF by varying k in the range from 30 to 50.

4.3 Performance Comparison

We first perform experiments to compare the classification accuracy of the eight algorithms. For each dataset, we generate the training data by randomly selecting positive examples and negative examples at the ratio of 95% to 5%, and apply the supervised outlier detection algorithms to the training data and evaluate the performance on the remaining test data. To avoid sampling bias, we repeat the above process for 10 times, and report the average AUC values for the 10 datasets in Fig. 3.

As we can see from the figure, our proposed methods, i.e., CS-SVDD, LS-SVDD, CBS-SVDD and LBS-SVDD can consistently outperform the other four baselines on all the 10 datasets. We also discover that, CBS-SVDD and LBS-SVDD outperform CS-SVDD and LS-SVDD respectively, this shows that the likelihood values used in bi-soft-SVDD can contribute to performance than the single likelihood value used in soft-SVDD. It is worth noting that LBS-SVDD and LS-SVDD can yield better accuracy than CBS-SVDD and CS-SVDD respectively on most of the datasets. This is because, the likelihood values computed by the kernel LOF-based method can better capture the local distribution of data, in particular, when the data has varying densities. As a result, the performance of bi-soft-SVDD and soft-SVDD can be better enhanced. In addition, we discover

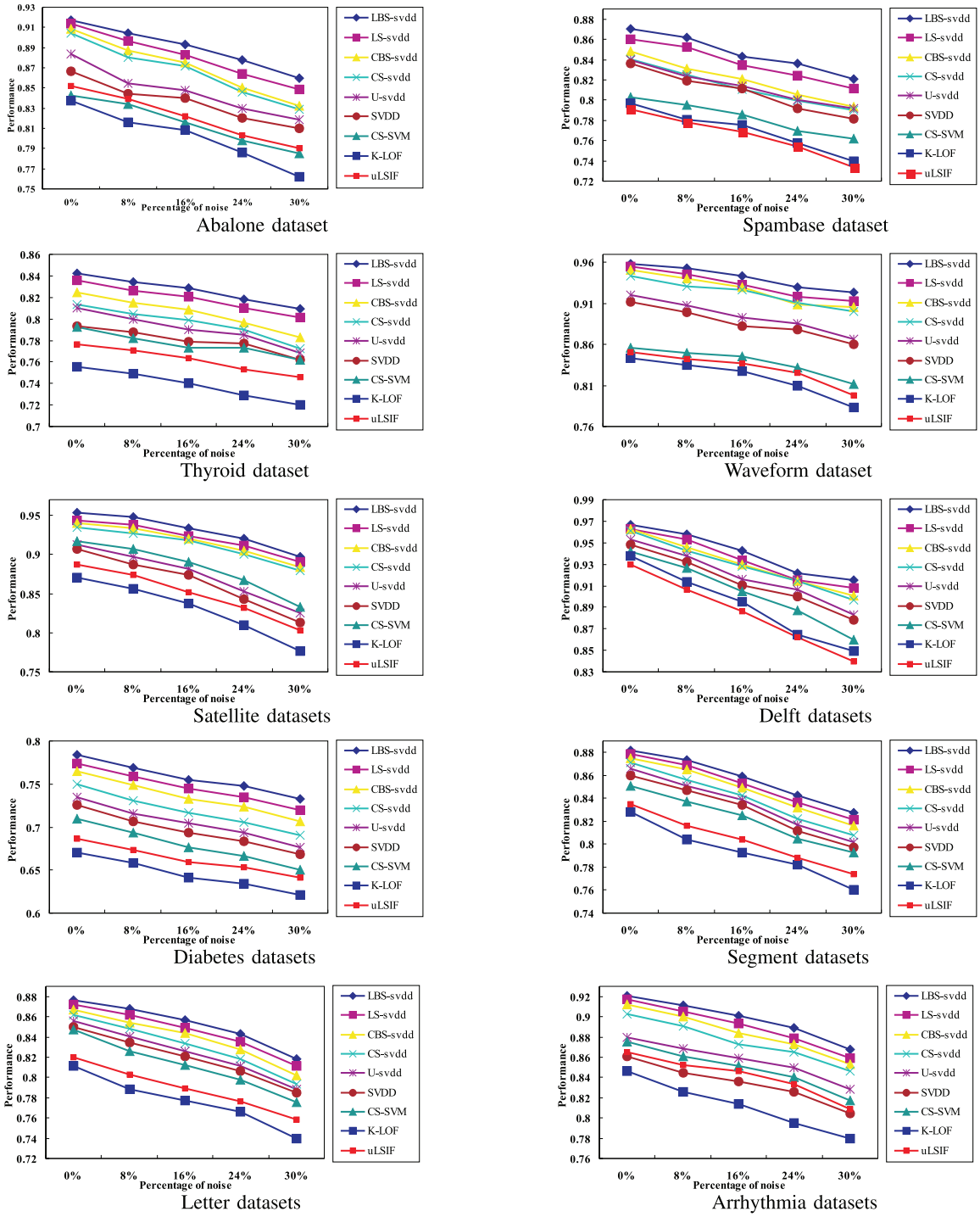


Fig. 5. Comparison of AUC values with respect to different percents of training data corrupted by noise.

that LS-SVDD performs better than CBS-SVDD, as the LOF-based likelihood values can capture the local distribution of data compared with the clustering-based likelihood values.

Above, we have illustrated that, the AUC values of CS-SVDD, LS-SVDD, CBS-SVDD and LBS-SVDD are higher than other outlier detection algorithms. Taking the Abalone and Spambase data sets for examples, we illustrate the ROC curves of the nine outlier detection algorithms in Fig. 4 for one out of ten groups of data. It is noted that, CS-SVDD, LS-SVDD, CBS-SVDD and LBS-SVDD can consistently outperform the other baselines. In addition, CBS-SVDD outperforms CS-SVDD, U-SVDD, SVDD, CS-SVM, uLSIF and kernel LOF. For the other datasets, we find similar phenomenon.

4.4 Sensitivity to Input Data Noise

We also conduct experiments to investigate the sensitivity of the nine algorithms to the noise added into the input data. Following the method used in [48], we generate the

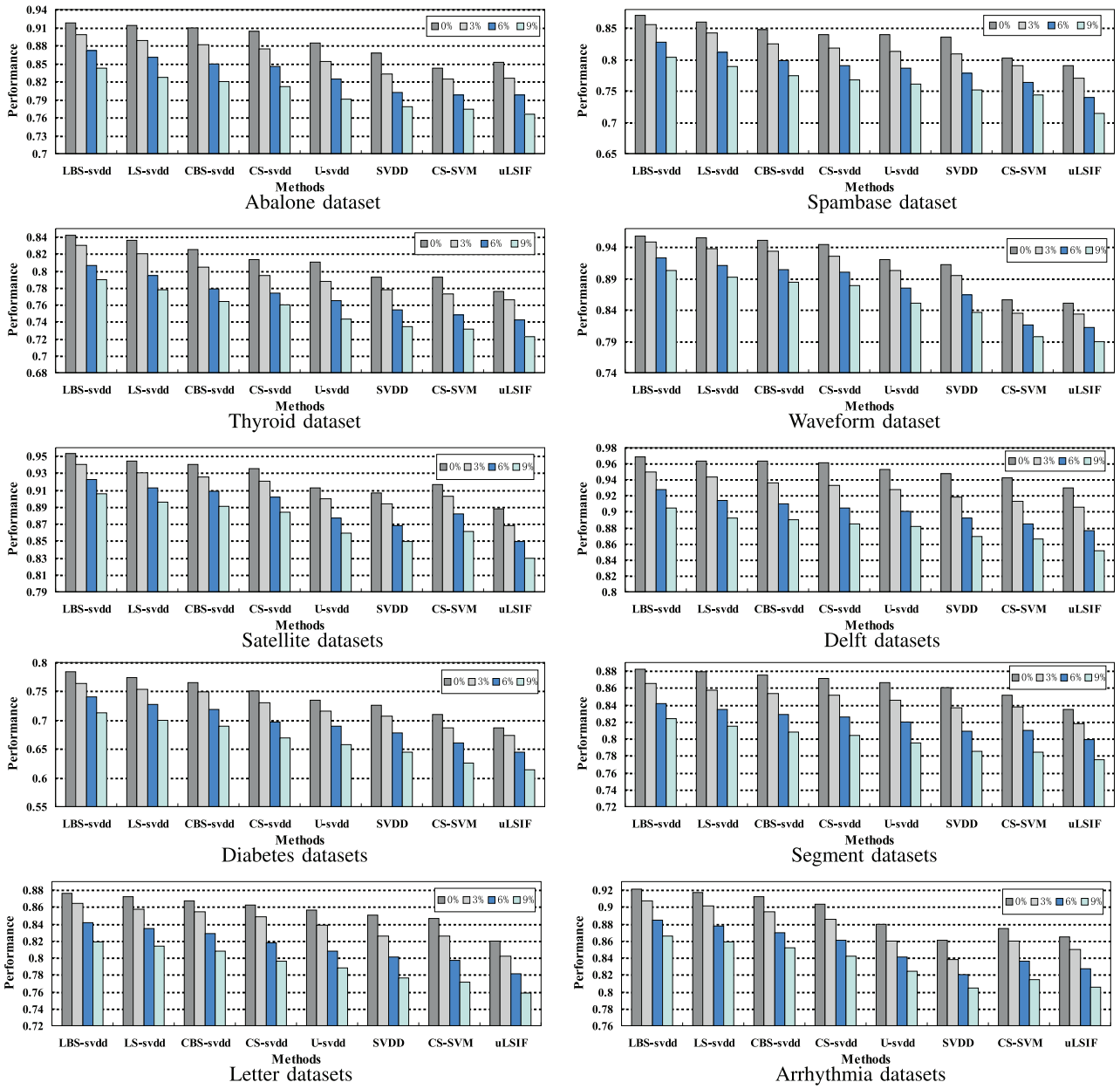


Fig. 6. Performance comparison under different percentage of data with error labels.

noise using a Gaussian distribution with zero mean and standard deviation determined as follows. For each dataset, we first calculate the standard deviation σ_i^0 of the entire data along the ith dimension, and then obtain the standard deviation of the Gaussian noise σ_i randomly from the range $[0, 2 \cdot \sigma_i^0]$. In this way, noise can be added to the positive class as a vector having the same dimension as the original dataset.

Fig. 2(b) illustrates the basic idea of the method used to add the noise to data examples. Specifically, the standard deviation σ_i^0 of the entire data along the ith dimension is first obtained. In order to model the difference in noise on different dimensions, we define the standard deviation σ_i along the ith dimension, whose value is randomly drawn from the range $[0, 2 \cdot \sigma_i^0]$. Then, for the ith dimension, we add noise from a random distribution with standard deviation σ_i. In this way, a data example $\mathbf{x}_j$ is added with the noise, which can be presented as a vector

$$\sigma^{\mathbf{x}_j} = [\sigma_1^{\mathbf{x}_j}, \sigma_2^{\mathbf{x}_j}, \ldots, \sigma_{n-1}^{\mathbf{x}_j}, \sigma_n^{\mathbf{x}_j}]. \quad (42)$$

Here, n denotes the number of dimensions for a data example $\mathbf{x}_j$, and $\sigma_i^{\mathbf{x}_j}$, $i = 1, \cdots n$ represents the noise added into the ith dimension of the data example.

In our experiments, we make the percentage of the data corrupted by noise vary from 0% to 30%, and apply the nine methods on these datasets. Fig. 5 shows the AUC values achieved by the nine algorithms with respect to different percentages of training data corrupted by noise. It is easily discovered that, as more noise is added into the training data, the overall performance of the nine methods degrades. This occurs because, when more noise is involved, target class becomes more indistinguishable from negative class. However, we can clearly see that, our four

TABLE 3
Comparison of AUC Values with Respect to Different Ratios of Normal Data Size to Abnormal Data Size in the Training Dataset

Datasets	Ratio	CS-SVDD	CBS-SVDD	LS-SVDD	LBS-SVDD	CS-SVM
Abalone	98:2	0.887	0.889	0.892	**0.898**	0.765
	95:5	0.904	0.909	0.913	**0.917**	0.842
	90:10	0.913	0.913	0.918	**0.925**	0.915
Spambase	98:2	0.835	0.835	0.843	**0.850**	0.746
	95:5	0.840	0.848	0.850	**0.862**	0.803
	90:10	0.844	0.850	0.858	**0.868**	0.853
Thyroid	98:2	0.804	0.809	0.813	**0.832**	0.786
	95:5	0.813	0.825	0.836	**0.842**	0.812
	90:10	0.825	0.840	0.840	**0.852**	0.836
Waveform	98:2	0.934	0.934	0.939	**0.944**	0.706
	95:5	0.944	0.951	0.955	**0.958**	0.856
	90:10	0.957	0.962	0.966	**0.969**	0.953
Delft Pump	98:2	0.946	0.946	0.951	**0.953**	0.826
	95:5	0.961	0.963	0.963	**0.968**	0.942
	95:10	0.968	0.968	0.968	**0.974**	0.961
Satellite Grey soil	95:2	0.924	0.93	0.935	**0.938**	0.807
	95:5	0.935	0.94	0.944	**0.953**	0.917
	90:10	0.938	0.944	0.948	**0.958**	0.944

methods, CS-SVDD, LS-SVDD, CBS-SVDD and LBS-SVDD, can still consistently yield higher accuracy than kernel LOF, SVDD, U-SVDD, uLSIF, and CS-SVM. This concludes that, our proposed soft-SVDD and bi-soft-SVDD is not affected by noise more than the methods used for comparison are. We can discover that CBS-SVDD and LBS-SVDD using bi-likelihood values can contribute more to the classifier contribution than CS-SVDD, LS-SVDD using single likelihood value.

4.5 Performance to Error Labels

In section 4.4, we add noise into the input data to make the labels of data imperfect. This is because the original labels of data are assigned based on the input data, when the input data are added with noise, the corresponding labels are not correct any more. The experiments have shown that our proposed approaches can consistently obtain higher performance.

In this set of experiment, we flip the labels of the data, i.e., label the data with the labels of the opposite class. This kind of operation has been performed in the previous work [49], [50]. Since K-LOF calculates the outlier factors only based on the test data, without involving the training data, we omit this baseline at this experiment. In our experiments, we mislabel the percentage of the data from 0% to 9%, and apply the eight methods on these datasets. Fig. 6 illustrates the AUC values of eight methods with respect different percentages of training data mislabeled. We can find that, the overall performance of the eight methods degrades when more data are mislabeled; however, our four methods, CS-SVDD, LS-SVDD, CBS-SVDD and LBS-SVDD, can still consistently obtain better performance than SVDD, U-SVDD, uLSIF, and CS-SVM.

4.6 Impact of Imbalanced Data Distribution

We have demonstrated that the SVDD-based approaches: CBS-SVDD, LBS-SVDD, CS-SVDD and LS-SVDD cam consistently outperform CS-SVM when the number of abnormal data is much smaller than the number of normal data.

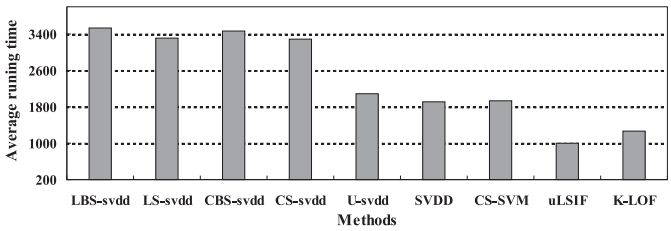


Fig. 7. Average running time of each outlier detection approach.

However, it is still interesting to see how the performance of the three algorithms would be affected when changing the number of abnormal data in the training.

Table 3 shows the AUC values with respect to different ratios of normal data size to abnormal data size in the training data of the first six data sets. It is noted that as more abnormal examples are added into the training dataset, CS-SVM offers increasing accuracy. This is because more negative examples can offer more information from negative class to build a more accurate SVM. However, when the ratio of normal data size to abnormal data size are 98:2 and 95:5 for which the number of abnormal examples are very few, CBS-SVDD, LBS-SVDD, CS-SVDD and LS-SVDD can remarkably outperform CS-SVM. This is because, based on insufficient abnormal data, CS-SVM cannot construct an accurate decision boundary to distinguish two classes. This indicates that, our proposed method can yield higher accuracy in real-world applications where abnormal data are very scarce.

4.7 Average Running Time Comparison

So far, we have compared our proposed approaches with other outlier detection approaches with respect of performance, sensitivity to noise, impact of imbalanced data distribution, it is still interesting to know the average of running time of each outlier detection approach.

Fig. 7 illustrates the average running time of each outlier detection approach over the data sets. It is easy to discover that, CBS-SVDD, LBS-SVDD, CS-SVDD and LS-SVDD take much more running time than the original SVDD and U-SVDD since the former four approaches calculate the likelihood values or confidence values based on kernel k-means and kernel-LOF methods and take the limited abnormal samples into the learning phase; while SVDD and U-SVDD train the outlier detection classifiers only on the normal examples. We further find that, CBS-SVDD and LBS-SVDD take little more time than CS-SVDD and LS-SVDD, thought CBS-SVDD and LBS-SVDD utilize bi-likelihood values in the training phase. In addition, SVDD takes similar time as CS-SVM since both methods resolve the quadratic optimization problem.

5 Conclusion and Future Work

In this paper, we propose new model-based approaches to outlier detection by introducing likelihood values to each input data into the SVDD training phase. Our proposed method first captures the local uncertainty by computing likelihood values for each example based on its local data behavior in the feature space, and then builds global

classifiers for outlier detection by incorporating the negative examples and the likelihood values in the SVDD-based learning framework. We have proposed four variants of approaches to address the problem of data with imperfect label in outlier detection. Extensive experiments on ten real life data sets have shown that our proposed approaches can achieve a better tradeoff between detection rate and false alarm rate for outlier detection in comparison to state-of-the-art outlier detection approaches.

We plan to extend our work in several directions. First, we would like to investigate how to design better mechanisms to generate likelihood values based on the data characteristics in a given application domain. Second, we will look into how to use an online process to learn the hyper-sphere boundary of soft-SVDD in streaming environments.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their very useful comments and suggestions. This work is supported in part by the US National Science Foundation through grants IIS-0905215, CNS-1115234, IIS-0914934, DBI-0960443, and OISE-1129076, US Department of Army through grant W911NF-12-1-0066, Google Mobile 2014 Program and KAU grant, Natural Science Foundation of China (61070033, 61203280, 61202270), Guangdong Natural Science Funds for Distinguished Young Scholar (S2013050014133), Natural Science Foundation of Guangdong province (9251009001000005, S2011040004187, S2012040007078), Specialized Research Fund for the Doctoral Program of Higher Education (20124420120004), Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, Overseas Outstanding Doctoral Fund (405120095), Australian Research Council Discovery Grant (DP1096218, DP130102691), and ARC Linkage Grant (LP100200774, LP120100566).

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM CSUR*, vol. 41, no. 3, Article 15, 2009.
[2] V. J. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 85–126, 2004.
[3] D. M. Hawkins, *Identification of Outliers.* Chapman and Hall, Springer, 1980.
[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 823–839, May 2012.
[5] V. Barnett and T. Lewis, *Outliers in Statistical Data.* Chichester, U.K.: Wiley, 1994.
[6] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, New York, NY, USA, 2000, pp. 93–104.
[7] S. Y. Jiang and Q. B. An, "Clustering-based outlier detection method," in *Proc. ICFSKD*, Shandong, China, 2008, pp. 429–433.
[8] C. Li and W. H. Wong, "Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection," in *Proc. Natl. Acad. Sci. USA*, 2001, pp. 31–36.
[9] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Mach. Learn.*, vol. 54, no. 1, pp. 45–66, 2004.
[10] D. M. J. Tax, A. Ypma, and R. P. W. Duin, "Support vector data description applied to machine vibration analysis," in *Proc. ASCI*, 1999, pp. 398–405.
[11] C. C. Aggarwal and P. S. Yu, "A survey of uncertain data algorithms and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 5, pp. 609–623, May 2009.
[12] L. Chen and C. Wang, "Continuous subgraph pattern search over certain and uncertain graph streams," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 8, pp. 1093–1109, Aug. 2010.
[13] A. Boukerche, R. B. Machado, K. R. L Juca, J. B. M. Sobral, and M. S. M. A. Notare, "An agent based and biological inspired real-time intrusion detection and security model for computer network operations," *Comput. Commun.*, vol. 30, no. 16, pp. 49–60, 2007.
[14] A. O. Tarakanov, "Immunocomputing for intelligent intrusion detection," *IEEE Comput. Intell. Mag.*, vol. 3, no. 2, pp. 22–30, May 2008.
[15] Y. J. Lee, Y. R. Yeh, and Y. C. F. Wang, "Anomaly detection via online over-sampling principal component analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 7, pp. 1460–1470, May 2012.
[16] E. Eskin, "Anomaly detection over noisy data using learned probability distributions," in *Proc. ICML*, San Francisco, CA, USA, 2000, pp. 255–262.
[17] F. Chen, C. T. Lu, and A. P. Boedihardjo, "GLS-SOD: A generalized local statistical approach for spatial outlier detection," in *Proc. ACM SIGKDD Int. Conf. KDD*, New York, NY, USA, 2010, pp. 1069–1078.
[18] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, "Statistical outlier detection using direct density ratio estimation," *Knowl. Inform. Syst.*, vol. 26, no. 2, pp. 309–336, 2011.
[19] R. Smith, A. Bivens, M. Embrechts, C. Palagiri, and B. Szymanski, "Clustering approaches for anomaly based intrusion detection," in *Proc. Intell. Eng. Syst. Artif. Neural Netw.*, 2002, pp. 579–584.
[20] Y. Shi and L. Zhang, "COID: A cluster-outlier iterative detection approach to multi-dimensional data analysis," *Knowl. Inform. Syst.*, vol. 28, no. 3, pp. 709–733, 2011.
[21] A. Ghoting, S. Parthasarathy, and M. E. Otey, "Fast mining of distance-based outliers in high-dimensional datasets," *Data Min. Knowl. Discov.*, vol. 16, no. 3, pp. 349–364, 2008.
[22] A. Ghoting, S. Parthasarathy, and M. Otey, "Fast mining of distance-based outliers in high-dimensional datasets," *Data Min. Knowl. Discov.*, vol. 16, no. 3, pp. 349–364, 2008.
[23] F. Angiulli and F. Fassetti, "Dolphin: An efficient algorithm for mining distance-based outliers in very large datasets," *ACM Trans. Knowl. Discov. Data*, vol. 3, no. 4, pp. 1–57, 2009.
[24] V. Niennattrakul, E. J. Keogh, and C. A. Ratanamahatana, "Data editing techniques to allow the application of distance-based outlier detection to streams," in *Proc. IEEE ICDM*, Sydney, NSW, USA, 2010, pp. 947–952.
[25] K. Bhaduri, B. L. Matthews, and C. Giannella, "Algorithms for speeding up distance-based outlier detection," in *Proc. ACM SIGKDD Int. Conf. KDD*, New York, NY, USA, 2011, pp. 859–867.
[26] E. M. Jordaan and G. F. Smits, "Robust outlier detection using SVM regression," in *Proc. IJCNN*, 2004, pp. 1098–1105.
[27] N. Abe, B. Zadrozny, and J. Langford, "Outlier detection by active learning," in *Proc. ACM SIGKDD Int. Conf. KDD*, New York, NY, USA, 2006, pp. 504–509.
[28] I. Steinwart, D. Hush, and C. Scovel, "A classification framework for anomaly detection," *J. Mach. Learn. Res.*, vol. 6, pp. 211–232, Dec. 2005.
[29] J. Theller and D.M. Cai, "Resampling approach for anomaly detection in multispectral images," in *Proc. SPIE*, Orlando, FL, USA, 2003, pp. 230–240.
[30] D. Tax and R. Duin, "Outlier detection using classifier instability," in *Proc. Adv. Pattern Recognit.*, London, U.K., 1998, pp. 593–601, LNCS.
[31] B. Liu, Y. Xiao, L. Cao, Z. Hao, and F. Deng, "Svdd-based outlier detection on uncertain data," *Knowl. Inform. Syst.*, vol. 34, no. 3, pp. 597–618, 2013.
[32] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. IJCAI*, San Francisco, CA, USA, 2001, pp. 973–978.
[33] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Editorial: Special issue on learning from imbalanced data sets," *SIGKDD Explorations*, vol. 6, no. 1, pp. 1–6, 2004.
[34] M. V. Joshi and V. Kumar, "CREDOS: Classification using ripple down structure (a case for rare classes)," in *Proc. SIAM Conf. Data Min.*, 2004.

[35] P. Chan and S. Stolfo, "Toward scalable learning with non-uniform class and cost distributions," in *Proc. ACM SIGKDD Int. Conf. KDD*, 1998, pp. 164–168.
[36] G. Nakhaeizadeh, U. Knoll, and B. Tausend, "Cost-sensitive pruning of decision trees," in *Proc. ECML*, Catania, Italy, 1994, pp. 383–386.
[37] G. Fumera and F. Roli, "Cost-sensitive learning in support vector machines," in *Proc. Workshop Mach. Learn. Meth. Appl.*, 2002.
[38] Y. Lin, Y. Lee, and G. Wahba, "Support vector machine for classification in nonstandard situations," *Mach. Learn.*, vol. 46, no. 1–3, pp. 191–202, 2002.
[39] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proc. ECML*, Pisa, Italy, 2004, pp. 39–50.
[40] B. X. Wang and N. Japkowicz, "Boosting support vector machines for imbalanced data sets," *Knowl. Inform. Syst.*, vol. 25, no. 1, pp. 1–20, 2010.
[41] C. X. Ling, J. Huang, and H. Zhang, "AUC: A statistically consistent and more discriminating measure than accuracy," in *Proc. IJCAI*, San Francisco, CA, USA, 2003, pp. 519–526.
[42] V. Vapnik, *Statistical Learning Theory*. London, U.K.: Springer-Verlag, 1998.
[43] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *Proc. ACM SIGKDD Int. Conf. KDD*, New York, NY, USA, 2005, pp. 157–166.
[44] M. Wu and J. Ye, "A small sphere and large margin approach for novelty detection using training data with outliers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 2088–2092, Nov. 2009.
[45] *UCI Machine Learning Repository* [Online]. Available: http://archive.ics.uci.edu/ml/datasets.html
[46] D. M. J. Tax. *Outlier Detection Datasets* [Online]. Available: http://homepage.tudelft.nl/n9d04/occ/index.html
[47] Y. Batistakis, M. Halkidi, and M. Vazirgiannis, "Cluster validity methods: Part i," in *Proc. ACM SIGMOD Rec.*, vol. 31. New York, NY, USA, pp. 40–45, 2002.
[48] C. C. Aggarwal and P. S. Yu, "Outlier detection with uncertain data," in *Proc. SIAM Conf. Data Min.*, 2008, pp. 483–493.
[49] T. Yang, M. Mahdavi, R. Jin, L. Zhang, and Y. Zhou, "Multiple kernel learning from noisy labels by stochastic programming," in *Proc. 29th ICML*, Edinburgh, U.K., 2012.
[50] W. Li and D. Yeung, "MILD: Multiple-instance learning via disambiguation," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 1, pp. 76–89, Jan. 2010.

Bo Liu is with the Guangdong University of Technology, Guangzhou, China, and University of Illinois at Chicago, Chicago, IL, USA. His current research interests include machine learning and data mining. He has published papers in *IEEE Transactions on Neural Networks*, *IEEE Transactions on Knowledge and Data Engineering*, *Knowledge and Information Systems*, *IJCAI*, *ICDM*, *SDM*, and *CIKM*.

Yanshan Xiao is with the Department of Computer Science, Guangdong University of Technology, Guangzhou, China. Her current research interests include multi-instance learning and data mining.

Philip S. Yu received the B.S. degree in electrical engineering from National Taiwan University, the M.S. and Ph.D. degrees in electrical engineering from Stanford University, and the MBA degree from New York University. He is a professor in the Department of Computer Science, University of Illinois, Chicago, IL, USA, and also holds the Wexler Chair in Information and Technology. He is a fellow of the ACM and the IEEE. His research interests include data mining, Internet applications and technologies, database systems, multimedia systems, parallel and distributed processing, and performance modeling.

Zhifeng Hao is with the Faculty of Computer, Guangdong University of Technology, Guangzhou, China. His current research interests include design and analysis of algorithm, mathematical modeling, and combinatorial optimization.

Longbing Cao is a Professor with the Faculty of Information Technology, University of Technology, Sydney, NSW, Australia. His current research interests include data mining, multi-agent technology, and agent and data mining integration.