Efficient Unsupervised Parameter Estimation for One-Class Support Vector Machines

Zahra Ghafoori, Sarah M. Erfani, Sutharshan Rajasegarar, James C. Bezdek, Life Fellow, IEEE, Shanika Karunasekera, and Christopher Leckie

Abstract—One-class support vector machines (OCSVMs) are very effective for semisupervised anomaly detection. However, their performance strongly depends on the settings of their hyperparameters, which has not been well studied. Moreover, unavailability of a clean training set that only comprises normal data in many real-life problems has given rise to the application of OCSVMs in an unsupervised manner. However, it has been shown that if the training set includes anomalies, the normal boundary created by OCSVMs is prone to skew toward the anomalies. This problem decreases the detection rate of anomalies and results in poor performance of the classifier. In this paper, we propose a new technique to set the hyperparameters and clean suspected anomalies from unlabelled training sets. The proposed method removes suspected anomalies based on a \( K \)-nearest neighbors technique, which is then used to directly estimate the hyperparameters. We examine several benchmark data sets with diverse distributions and dimensionality. Our findings suggest that on the examined data sets, the proposed technique is roughly 70 times faster than supervised parameter estimation via grid-search and cross-validation, and one to three orders of magnitude faster than broadly used semisupervised and unsupervised parameter estimation methods for OCSVMs. Moreover, our method statistically outperforms those semisupervised and unsupervised methods and its accuracy is comparable to supervised grid-search and cross-validation.

Index Terms—Anomaly detection, one-class support vector machine (OCSVM), outlier detection, parameter estimation, unsupervised learning.

I. INTRODUCTION

Anomalies patterns in a data set, which are inconsistent with the majority of the data, are often referred to as anomalies or outliers. In many applications, such as intrusion and fraud detection, environmental monitoring, and medical diagnosis, the major task is to detect such instances and control (or avoid) their (usually harmful) impact [1]. In addition, machine learning algorithms are commonly sensitive to the presence of anomalies in the training set, because they can mislead the learning process [2].

Anomaly detection algorithms can be categorized as supervised, semisupervised, and unsupervised learning methods [2], [3]. The supervised case assumes that labels for both negative (anomalies) and positive (target concept) classes are available to train a classifier, while the unsupervised approach deals with situations where no ground truth labels are available. Compared with these two approaches, semisupervised methods assume that only the normal examples are available during training, which makes it possible to build a model of normality that rejects anomalous instances. If the majority of the training data for an anomaly detection technique is normal and its aim is to model the normal profile, it is categorized as a one-class classification method. In this paper, we investigate one-class classification and unsupervised learning for anomaly detection. Interested readers are referred to [2] and [3] for comprehensive surveys on other anomaly detection methods.

The definition of anomalies is application-dependent, but, in general, the two major underlying assumptions are that anomalies are rare and distinctively different from normal data [2]. The difference can be defined based on a distance [4] or density measure [5] or a model-driven score [6]–[9]. The hyperplane one-class support vector machine (OCSVM) [7] and support vector data description (SVDD) [6] algorithms are one-class classification methods that are widely used for anomaly detection [10]–[15]. These methods are density-based and model the normal data as a single class that occupies a dense subset of the input space. A test instance that resides in such a subset is accepted by the model, whereas anomalies are not [6], [7]. Usually the anomalies are not linearly separable from the normal data, and a kernel function is used to (implicitly) project the data to a kernel space that results in linear separation. Assuming two training examples \( x \) and \( y \), if an applied kernel only depends on \( x - y \), i.e., the kernel is stationary, the SVDD and OCSVM algorithms result in the same solution [7]. We focus on the OCSVM in this paper, but because we use a stationary kernel, our findings can be generalized to SVDD as well.

Although OCSVMs are theoretically well-founded, there are some challenges that affect their performance when they are trained in an unsupervised instead of a semisupervised manner. First, OCSVMs have data-dependent parameters that can substantially affect their performance, and setting these parameters in an efficient and unsupervised manner is an open research problem [1], [16]. Supervised grid-search and \( r \)-fold cross validation can be used to find optimal parameters, however, this method is computationally expensive, does not scale well, and to be effective, requires labeled examples from both the normal samples and anomalies. Contributing “good” examples of anomalies, i.e., ones that do not lie in normal regions and surround the normal data [17], [18], can mitigate...
the latter problem, but it is still computationally expensive and sometimes infeasible for high-dimensional data sets.

Second, it has been shown that OCSVMs handle very small fractions of anomalies in the training set [6], [7], but when the fraction of anomalies increases, they generate models that are skewed toward the anomalies [1], [19]. In Section II, we argue that the fraction of anomalies that can be tolerated by OCSVMs such that they generate a reliable normal profile is data set-dependent and unknown. Therefore, if it is not guaranteed that an unlabelled training set represents only the normal concept, a preprocessing step on the training set is required before applying OCSVMs to build a normal profile [19]. To mitigate this problem, Amer et al. [19] proposed two algorithms called robust OCSVM (ROCSVM) and \( \eta \)OCSVM. Each of these algorithms modifies the OCSVM objective function by introducing a weight vector for training samples. ROCVM aims to penalize suspected anomalies to be selected as boundary support vectors, whereas \( \eta \)OCSVM allows some points to be marked as anomalies that do not participate in the final solution by zeroing their weights. In this paper, instead of modifying the OCSVM objective function, we propose a preprocessing step that simultaneously finds a proper hyperparameter setting and removes suspected anomalies. Our contributions can be summarized as follows.

1) We combine the \( K \)-nearest neighbors (\( K \)-NN) and the OCSVM methods in a novel way to estimate the local density of normal data. The hyperparameters are then theoretically driven based on a well-defined local density threshold that differentiates anomalies from normal points.

2) The hyperparameters of the OCSVM can take any real value, so their choice dramatically affects classifier performance. We show that our proposed method reduces the problem of setting such hyperparameters from large domains to setting the integer value \( K \) in \( K \)-NN. The \( K \) parameter has a small range so its choice does not highly affect the OCSVM’s performance.

3) We compare the accuracy and CPU-time of our method to existing parameter estimation methods for OCSVMs. Our numerical experiments show that our scheme statistically outperforms state-of-the-art semisupervised and unsupervised methods for estimating the hyperparameters of OCSVMs.

The main drawback of using \( K \)-NN-based anomaly detection directly instead of training an OCSVM model is its high computational cost during the test phase. This requires keeping all the training samples in memory and computing the distances between any new test point and the training samples to identify anomalies in future data. On the other hand, the OCSVM decision function only relies on a limited subset of training samples, namely support vectors, and is much faster than \( K \)-NN-based techniques.

The rest of this paper is organized as follows. We review the OCSVM [7] algorithm and related literature in Section II, then formulate the problem and propose our methodology in Sections III and IV. We empirically evaluate our proposed technique in Section V and summarize our findings and future directions in Section VI.

II. BACKGROUND AND RELATED WORK

In this section, the hyperplane OCSVM algorithm is briefly explained, followed by a review of related works that have been proposed to set the hyperparameters of the OCSVM.

A. One-Class Support Vector Machines

Schölkopf et al. [7] proposed the one-class classification method that we call OCSVM, for anomaly detection in high-dimensional real-world data sets. For a given training set, the algorithm estimates minimal subsets in the input space that contain a predefined fraction of the data. If a training set is mostly normal and the predefined fraction is high, the algorithm finds a boundary around dense areas comprising normal data [3]. Consequently, any test instance that resides on or inside such a boundary is labeled as a normal point; otherwise, it is labeled as an anomaly.

In many cases, the training set distribution is complicated and not linearly separable from areas with low density in the input space. As a result, the OCSVM (implicitly) projects the training set \( \{x_1, x_2, \ldots, x_n\} \) from the input space \( \mathbb{R}^d \) into a higher dimensional feature space \( \mathbb{R}^{d'} \), where \( d' \ll d \), using a feature map \( \phi : \mathbb{R}^d \rightarrow \mathbb{R}^{d'} \). The aim of the mapping is to make the data linearly separable in the new space, which in the simplest case can be represented by an inner product, but usually more sophisticated kernels are used to provide the separation. Using this technique, the OCSVM algorithm finds a hyperplane that separates the projected examples from the origin with the maximum possible margin. Therefore, the problem of fitting a nonlinear boundary around dense areas in the input space is reduced to finding a linear boundary in the feature space \( \mathbb{R}^{d'} \).

The primal quadratic problem that defines the OCSVM classifier is

\[
\begin{align*}
\min_{\omega, \xi, \rho} \quad & \frac{1}{2} \|\omega\|^2 + \frac{1}{n} \sum_{i=1}^{n} \xi_i - \rho \\
\text{s.t.} \quad & \langle \omega, \phi(x_i) \rangle \geq \rho - \xi_i \\
& \xi_i \geq 0, \quad i = 1, 2, \ldots, n
\end{align*}
\]

where \( \omega \in \mathbb{R}^d, \xi = [\xi_1, \ldots, \xi_n] \) are slack variables, \( \rho \) is a bias term, and \( 0 < \nu \leq 1 \). The slack variables relax the problem constraints by allowing some examples to fall on the origin side of the hyperplane. This relaxation generates a solution with a higher margin, i.e., a larger value for \( \rho/\|\omega\| \), and shrinks the estimated volume of the normal data in the input space. The \( \nu \) parameter controls the level of relaxation. If \( \nu \) is set near 0, the penalty term \( \sum_{i=1}^{n} \xi_i \) vanishes, because it forces all the \( \xi_i \rightarrow 0 \), which forces the algorithm to find a hyperplane that separates almost all the training examples from the origin in the feature space \( \mathbb{R}^{d'} \). Conversely, if this parameter is very close to 1, the algorithm has greater freedom to leave more points on the origin side of the hyperplane by increasing the penalty term and selecting larger values for the \( \rho \) variable and the \( \omega \) vector while minimizing (1). In this case, the final solution classifies almost all the examples as anomalies.

Explicit computation of the mapping \( \phi \) from the input space to the feature space \( \mathbb{R}^{d'} \) might be quite expensive (or
impossible), whereas the OCSVM actually uses a similarity matrix and does not require the explicit mapping for each training instance. Instead, a kernel function $k(x_i, x_j)$ is used to compute a positive-definite matrix $K \in \mathbb{R}^{n \times n}$. In this paper, we use a Gaussian kernel $k(x, y) = e^{-\gamma ||x - y||^2}$ called the radial basis function (RBF) kernel. The $\gamma$ parameter is the kernel bandwidth and its role will be clarified in Section II-B.

To avoid computing the explicit mapping, (1) is converted to the following quadratic dual problem using Lagrange multipliers:

$$
\min_{[\alpha_i]} \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j k(x_i, x_j) \\
\text{s.t. } 0 \leq \alpha_i \leq 1/n \delta_i = 1, \quad i = 1, 2, \ldots, n.
$$

When the dual formulation is used, the value of the $\rho$ variable is computed by selecting an arbitrary point $x_i$ that resides on the hyperplane. Based on (2), any point $x_i$ with this property has $0 < \alpha_i < 1/n$, and for any such point $\rho = \sum_j \alpha_j k(x_j, x_i)$.

After the training phase, the label of any unseen data $x$ is simply predicted using $\text{sign}(f(x))$, where the decision function $f(x) = \langle \omega \cdot \phi(x) \rangle - \rho$ for the primal and $f(x) = (\sum_j \alpha_j k(x_j, x)) - \rho$ for the dual form. As a result, a given solution divides the training set into three groups.

1) Non-support Vectors (NSVs): Examples with $\text{sign}(f(x)) > 0$ that fall inside the normal boundary.
2) SVs: Examples with $\text{sign}(f(x)) = 0$ that are located on the boundary.
3) Anomalies or Bounded Support Vectors (BSVs): Examples with $\text{sign}(f(x)) < 0$ that reside outside the boundary.

SVs and BSVs participate in defining the normal boundary, and after the training phase, all NSVs can be discarded. Next, we explain the effect of the hyperparameters $\gamma$ and $\nu$ in the OCSVM model selection.

**B. Sensitivity of OCSVMs to the Hyperparameter Settings**

The kernel bandwidth parameter $\gamma$ and the regularization parameter $\nu$ have a major influence on the accuracy of a model generated by any member of the OCSVM family of algorithms [16], [20]. To illustrate, we have designed an experiment with a toy problem named the Half Kernel that includes 4000 normal samples (green in Fig. 1) and 5% (i.e., 200) anomalies (red in Fig. 1), which were added to the normal data at random using a uniform distribution on the unit square. Therefore, the dataset includes 4200 samples in total. Fig. 1 shows different models that can be generated using the OCSVM algorithm with different values of the $\gamma$ and $\nu$ parameters. The visually best model, i.e., Fig. 1(b), uses the parameter settings $(\nu^*, \gamma^*) = (0.1, 100)$. Fig. 1(a) and (c) shows that choosing $\gamma \ll \gamma^*$ results in an overly general and simple model with a high false positive rate (FPR) with respect to the target class, whereas a larger value results in a poor model with a high false negative rate (FNR). The $\nu$ parameter affects the results in a similar manner as shown in Fig. 1(d) and (e). In Fig. 1(f), we have added another 5% anomalies to the Half Kernel data set and used the same settings as in Fig. 1(b) to illustrate how these parameters are data-dependent. Fig. 1(b) shows how dramatically the model
may skew toward anomalies, if the hyperparameters are not properly set. In summary, the values of the \( \nu \) and \( \gamma \) parameters are data-dependent, and thus, an automatic method that recognizes this dependence is required to select these parameter values from the data. In Section II-C, we briefly review three categories of approaches to this problem.

C. Estimating Parameter Settings for the OCSVM Algorithm

For two-class SVMs, the availability of ground truth labels during the training phase makes it possible to use efficient methods like leave-one-out estimation to tune the hyperparameters [21]–[24]. In the case of unsupervised and semisupervised learning with OCSVMs, no ground truth is available for tuning the parameters this way. In this section, we elaborate on hyperparameter estimation methods that have been proposed for OCSVMs. These methods are divided into three categories: supervised, semisupervised, and unsupervised hyperparameter estimation approaches.

1) Supervised Learning of the Parameters: If ground truth labels are available, it is possible to optimize the choice of values for the \( \nu \) and \( \gamma \) parameters via \( r \)-fold cross validation and searching the parameter space. Zhuang and Dai [25] suggested that the most reliable approach to search the parameter space in this manner is to use grid-search, and due to its considerable computational requirements, they recommend using an efficient parameter search algorithm. To this end, they first applied a coarse-grained search over the entire parameter space and then performed two additional fine-grained searches to reduce the complexity by restricting the search space. Suvorov et al. [26] and Tax and Duin et al. [6] have proposed using samples from the anomaly class directly in the optimization function of the OCSVM or SVDD algorithms to boost their accuracy. However, Tax and Duin [6] have shown that choosing “poor” anomaly examples, i.e., anomalies that fall inside or very close to the target class, reduces the accuracy of the trained model so that it becomes similar to a random classifier. They showed that if only examples from the target class are available, generating synthetic anomalies in low-density regions can help tighten the data description and enhancing the accuracy via parameter setting [18].

2) Semisupervised Learning of the Parameters: These methods were designed for one-class classification and did not consider the case of using OCSVMs for unsupervised anomaly detection. Consequently, their training set only includes the normal data. Tran et al. [27] proposed a method, called \( \xi \)-ap-proximation, which can be used to estimate the generalization error of an OCSVM model. Wang et al. [28] proposed a technique to estimate the \( \gamma \) parameter that starts with a given range specified by an upper and a lower bound \([\gamma_1, \gamma_u]\) and performs a binary search to shrink this range based on a tightness measure, which is defined on the OCSVM model trained using \( \gamma = \frac{\gamma_1 + \gamma_2}{2} \). They assumed that the \( \nu \) parameter is given. Xiao et al. [29], [30] proposed a two-step algorithm called MIES to estimate the \( \gamma \) parameter. In the first step, a set of edge samples are automatically detected. Then, a set of candidate values for the \( \gamma \) parameter are examined, and one that results in mapping the identified edge samples nearest to the model surface and furthest from the interior samples is selected. MIES selects all training instances as edge samples for high-dimensional data sets and fails to estimate the \( \gamma \) parameter, unless sufficient training instances are available. Anaissi et al. [31] extended MIES to work for high-dimensional data sets. This method, called appropriate distance to the enclosing surface (ADES), trains a linear OCSVM model for each instance and its \( K \)-NNs. If a majority of such points are linearly separable from the origin, the corresponding instance is selected as an edge sample.

3) Unsupervised Learning of the Parameters: Several heuristic unsupervised approaches have been proposed to estimate the \( \gamma \) and \( \nu \) parameters, when no ground truth labels are available and the training set includes an unknown fraction of anomalies as well as the normal data. Emmott et al. [32] assume prior knowledge about the value of the \( \nu \) parameter. Then, the value of the \( \gamma \) parameter is increased until a predefined proportion of the data has been rejected. There is usually more than one pair of parameters that result in approximately this predefined proportion of anomalies that yield completely different models. Moreover, it is unclear to us how a priori knowledge for the fraction of anomalies is acquired. In contrast to Emmott et al. [32], Rätsch et al. [20] assumed that the value of the \( \gamma \) parameter is known. They proposed a heuristic to find an appropriate \( \nu \) value. One wonders how the \( \gamma \) parameter is known a priori. They also assumed that anomalies are far from normal samples. The idea is to increase the value of the \( \nu \) parameter over the range \((0, 1)\) to find a value that maximizes the separation distance between the normal class and the rejected samples. The distance is defined by the following equation:

\[
D_\nu = \frac{1}{N_+} \sum_{f(x) \geq 0} (f(x) + \rho) - \frac{1}{N_-} \sum_{f(x) < 0} (f(x) + \rho)
\]

where \( f() \) and \( \rho \) are, respectively, the decision function and the bias term of the OCSVM model for a given value of the \( \nu \) parameter. \( x \) is a training example, and \( N_+ \) and \( N_- \) are the number of samples detected as target and anomaly, respectively, given \( \text{sign}(f()) \). Rätsch et al. [20] have reported that if there is no clear separation between negative and positive samples, the proposed heuristic may come up with extreme solutions, i.e., \( \nu = 0 \) or \( 1 \). Moreover, the choice of \( \gamma \) and its effect on finding a good value for \( \nu \) was not addressed.

Liu et al. [1] estimated \( \gamma \) as \( \frac{1}{\gamma} = 2 \sum_{i,j=1}^{n} ||x_i - x_j||^2/n^2 \), where the norm is Euclidean. This method for estimating \( \gamma \) can be used in combination with Rätsch’s method to estimate both \( \gamma \) and \( \nu \) in a fully unsupervised manner. We call this method Duplex Max-margin Model Selection (DMMS) as it is based on the max-margin principle and maximizes the separation between the two classes.

Tax and Duin [16] proposed a heuristic to estimate \( \gamma \) and \( \nu \) for an SVDD in a fully unsupervised manner. Their proposed heuristic optimizes the estimated FP and FN rates by solving the following minimization problem:

\[
\arg \min_{\nu,\gamma} \frac{|\text{SV}_{(\nu,\gamma)}|}{N} + \lambda |\text{SV}_{(\nu,\gamma)}| \left( \frac{1}{\gamma 0.5s_{\text{max}}} \right)^d
\]
where $s_{\text{max}}$ represents the maximum distance between two vectors in the training set. Given a pair of values $(\nu, \gamma)$, the set of support vectors for the corresponding trained model is denoted by SV$\nu,\gamma$, whereas SV$\nu_b,\gamma$ is the subset of border support vectors (i.e., those with $0 < a_i < C$, where $C = 1/\nu$), and $\lambda$ is a regularizer. The first term ($SV_{\nu,\gamma}(\nu,\gamma)$) is the fraction of training samples selected as support vectors, which estimates the error on the target class, and the second term controls the error on the anomaly class. Since the RBF kernel has been used for this heuristic, it is also possible to use this method to estimate the parameter settings for the OCSVM algorithm [16]. Hereafter, we refer to this heuristic as the \textit{dual} error-minimization model selection (DEMMS) as its objective is to minimize FPR and TPR.

Another work by Liu et al. [1] uses an unsupervised self-guided soft labeling mechanism to train a one-class classifier, different from the OCSVM and SVDD methods, by applying the soft labels directly in the optimization problem of the studied one-class classification algorithm, which is very different from the aim of this paper and so is not discussed here.

The shortcomings of these existing methods are as follows.

1) Even a moderately high-resolution grid-search may incur a substantial number of iterations and high time-complexity. Moreover, the granularity of the search has a major effect on the final result.

2) In many applications, examples from the anomaly class are not available for finding an optimal parameter setting via cross validation. Moreover, it is not assured that an unlabelled training set only includes normal samples.

3) Even if negative and positive examples are available, Tax and Duin [6] have shown that this does not guarantee improved accuracy for the trained model. The examples should be well representative of anomalous and normal patterns.

4) All the existing unsupervised parameter estimation methods (except DEMMS) assume prior knowledge of either $\gamma$ or $\nu$, but both parameters are usually unknown. This makes it impossible to optimize one parameter based on knowing the other one.

5) The DEMMS method is unsupervised, but it requires a mechanism like grid-search over the parameter space and suffers from the time-complexity problem in 1).

In Sections III–V, we propose and evaluate an unsupervised parameter estimation technique for one-class SVMs.

III. PROBLEM STATEMENT

We are given an unlabelled data set $D = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^d$. The data are drawn from an unknown underlying distribution, which results in creating subsets of simple or complicated shapes of arbitrary densities in the input space. The goal of anomaly detection using one-class classification is to identify normal subsets of the input space as a single class that comprises the majority of the data. In the test phase, any test point that resides in this region is labeled by $+1$ as a normal observation, else by $-1$ as an anomaly. The data set $D$ contains unknown anomalies and no ground truth labels are available. Therefore, the hyperparameters $\gamma$ and $\nu$ of the OCSVM algorithm with the RBF kernel cannot be set using supervised grid search and cross validation. The aim is to find a pair of hyperparameter settings denoted by $(\gamma^*, \nu^*)$ without requiring ground truth labels or resorting to exhaustive grid-search. We will use quadratic programming to approximate solutions of the OCSVM optimization problem shown in (2). This will result in (approximately) optimal pairs [denoted here by superscripts (*)] for the unsupervised case.

Assumption: Let $D^a \subset D$ denote the unknown anomalies in the training set $D$. We assume that $|D^a| < |D|$, where $|\ast|$ denotes the cardinality (or size) of the set. Moreover, we assume that there is a $K \in \{1, \ldots, n - 1\}$ such that $\delta^i_{avg}(D^a) > \delta^i_{avg}(D)$, where $\delta^i_{avg}(\ast)$ denotes the average distance, relative to any metric $\delta$ on $\mathbb{R}^d \times \mathbb{R}^d$, to the $K$-NN of every point in the set $(\ast)$.

Let us have a closer look at this assumption. There might not be a $K$ that satisfies it. In this case, any anomaly detection that relies only on a distance-based metric to distinguish between anomalies and normal data will fail to detect the anomalies. If the assumption is satisfied, ranking the training samples based on their distance to their $K$-NN and picking the top ranked ones can be used to identify potential members of $D^a$.

This procedure will be discussed in Section IV.

IV. METHODOLOGY

We discuss two unsupervised parameter estimation heuristics, called \textit{quick model selection} 1 and 2, QMS1 and QMS2, both of which consist of two steps: 1) estimating the $\nu$ parameter and 2) estimating the $\gamma$ parameter. The QMS1 algorithm is a technique we recently proposed for OCSVMs [33] and QMS2 is our new technique. We include QMS1 here to provide a benchmark for improvements made to it by the new technique (QMS2). Our numerical experiments will confirm that QMS2 is almost as fast as QMS1 and provides better estimates of $(\gamma^*, \nu^*)$ than QMS1, DEMS [1], [20], and DMMS [16].

A. Estimation of the $\nu$ Parameter

Based on Schölkopf \textit{et al.} [7], the $\nu$ parameter in (1) and (2) controls the fraction of training samples in a given training set $D$ that will be selected as SVs and BSVs (or anomalies). Consequently, identifying the fraction of BSVs and potential SVs in the training set $D$ gives us an estimate of the $\nu$ parameter.

To begin, we define $s_K^i$ of data instance $x_i$ in the input space as follows:

$$s_K^i = \frac{1}{K} \sum_{j \in \text{KNN}_i} \|x_i - x_j\|, \quad i = 1, 2, \ldots, n$$

(5)

where $\|\ast\|$ is a user-defined vector norm on $\mathbb{R}^d$ and $\text{KNN}_i$ is the set of indices of the $K$-NNs of $x_i$. In this paper, we use the $L^2$-norm, but any other norm could be used to cope with challenges in analyzing very high-dimensional data sets. We use the change point detection method proposed by Lavielle [34] to locate the last sudden change in the ordered list $S_K$, which is defined in the following.
H1. Identifying BSVs, $v = \frac{n - [(1 - \eta) \times m]}{n}$: According to Tax and Duin [6], the fraction of support vectors is an upper bound on the FNR of the OCSVM algorithm, which is based on Schölkopf et al. [7], controlled using the $v$ parameter. Let $\eta$ be the upper bound of the FNR on the normal class. Then, the maximum numbers of SVs and BSVs to avoid FNRs larger than $\eta$ are $\eta \times m$ and $n - m$, respectively, where $m$ is the last sudden change point in $S_K$. Consequently, the total number of SVs and BSVs should be less than or equal to $n - [(1 - \eta) \times m]$. This results in the estimate

$$v = \frac{n - [(1 - \eta) \times m]}{n}.$$  \hspace{1cm} (7)

Ghafoori et al. [33] chose $\eta$ in range $[0, 1]$. In this paper, the default value $\eta = 0$ is used for our experiments and we do not consider optimizing this value for QMS1 [33], because it is the baseline and our main contribution in this paper is QMS2.

H2$\nu$. Remove BSVs and Set $v = 0.01$: In view of the fact that the $\{m, m + 1, \ldots, n\}$ indices correspond to BSVs in the training set $D$, we can remove the corresponding samples from $D$ before running the OCSVM algorithm. This step avoids selecting these samples as SVs and prevents generating models skewed toward anomalies [19]. After removing the BSVs, only a small fraction of samples should be selected as SVs in the final solution. Moreover, the upper bound for the FNR should be very small, because almost all the samples of the new training set should be classified as normal. We empirically observed that after this step, setting $v = 0.01$ works for a wide range of data sets. In Section V, we show that this heuristic is robust to the choice of the $v$ parameter.

For example, for the data set in Fig. 2(a), we find that $m = 3950$, so with $\eta = 0$, H1$\nu$, (7) yields $v = 0.059$. Removing the $(n - m) = 250$ BSVs from $D$ per H2$\nu$ reduces the data set to only the green dots in Fig. 2(c), and the new estimate for $v$ is taken to be $v = 0.01$, per our empirical experience. Models corresponding to each of these heuristics are depicted in Fig. 3.

This method for estimation of $v$ computes the $K$-NN distances for each point over all training samples, which requires $O(n(d + K))$ where $n$ is the size of training set, $d$ is the data dimensionality and $K$ is the number of nearest neighbors. Dimension $d$ and the number of NNs $K$ are typically many orders of magnitude less than $n$, so our heuristics are more efficient than grid-search over the parameter space that requires several runs of the OCSVM algorithm. Combining the $K$-NNs and the OCSVM makes it possible to directly estimate the parameter settings of the OCSVM algorithm without running it multiple times. Moreover, unlike $K$-NN-based anomaly detection, OCSVMs are very fast and...
memory-efficient in the test phase, because the number of support vectors is much less than the size of the training set.

Next, we elaborate on our proposed approaches to estimate the RBF kernel bandwidth $\gamma$.

### B. Estimation of the $\gamma$ Parameter

Setting $\gamma$ is a more challenging task as it is strongly data-dependent and is affected by both the density and shape of the input distribution. This parameter acts as a scaling factor to smooth the learned density estimate to reflect the true data density. From Section II-A, a given training instance $x_i$ in the training set $D$ incorporates the following term in (2):

$$\alpha_i^2 + \alpha_i \sum_{j \neq i} \alpha_j K_{ij}$$

where $K_{ij} = k(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$, for a given training sample $x_i$, indicates its similarity to training sample $x_j$. The $\gamma$ parameter comes into play by controlling the rate of decay of the similarity as the distance from $x_i$ increases.

**Proposition 2 ($\gamma^*$-Property):** Let $\alpha = [\alpha_j, j \neq i] \subset \alpha$ be the vector that minimizes (2) for data set $D' = \{x_j, j \neq i\} \subset D$. Let $\gamma^*$ be the optimal value of $\gamma$ for this solution that yields the minimum generalization error, then

1) For any $x_i$ that should be classified as an NSV and $x_i' \in \{SV \cup BSV\} \subset D'$: $\sum_{j \neq i} K_{ij} > \sum_{j \neq i'} K_{i'j}$.

**Proof:** If $\sum_{j \neq i} K_{ij} \leq \sum_{j \neq i'} K_{i'j}$, then $a_i \geq a_i' > 0$ minimizes (2) for the training set $D$. This means $x_i$ will be misclassified, which contradicts the assumption that $\gamma^*$ is optimal.

2) For any $x_i$ that should be selected as an SV and $x_i' \in BSV \subset D'$: $\sum_{j \neq i} K_{ij} > \sum_{j \neq i'} K_{i'j}$. Proof: If $\sum_{j \neq i} K_{ij} \leq \sum_{j \neq i'} K_{i'j}$, then $a_i = a_i' = \frac{1}{\nu n}$ minimizes (2) for the training set $D$. This means $x_i$ will be misclassified as an outlier, which contradicts the assumption that $\gamma^*$ is optimal.

According to Proposition 2, the optimal value $\gamma^*$ scales the similarities between the training samples so that the accumulated similarity for NSVs becomes greater than that for the SVs and in turn, for the SVs, greater than the BSVs. We use our toy example to show that a poor choice of $\gamma$ disrupts this order. To this end, two models are trained using the OCSVM algorithm with the same value of the $\nu$ parameter but different choices of the $\gamma$ parameter. Fig. 4 depicts the data set, and the normal boundary of each trained model. The contour plots of similarities (computed using the RBF kernel) for two points, an anomaly and a boundary point, are chosen to illustrate Proposition 2. A good choice of the $\gamma$ parameter in Fig. 4(a) preserves the similarities among the instances in the transformed space. We see that $\sum_{j \neq i} K_{ij}$ for the given boundary point $x_{\text{boundary}}$ is bigger than this value for the given anomaly $x_{\text{anomaly}}$, because, compared with
\(x_{\text{anomaly}}\), the contour levels for \(x_{\text{boundary}}\), especially the inner ones that represent higher similarities, cover more data points. Consequently, this choice of the \(\gamma\) parameter results in a good fit. Fig. 4(b) shows contour plots of the similarities for the same instances when a poor choice of the \(\gamma\) parameter is used to build a model. Moving toward the last contour level, the accumulated similarity for the anomaly point \(x_{\text{anomaly}}\) becomes greater than the boundary point \(x_{\text{boundary}}\). As a result, \(x_{\text{anomaly}}\) is accepted as a normal point by the model, whereas \(x_{\text{boundary}}\) is misclassified as an anomaly. This toy example shows that a choice of \(\gamma\) that satisfies Proposition 2(2) also satisfies Proposition 2(1). As a result, if the \(\gamma\) parameter is set based on a density “limit” that differentiates data points on the boundary from BSVs, the OCSVM algorithm should be able to filter out anomalies in the optimal solution. Thus, we set \(\gamma\) to reflect the maximum local density of potential SVs that makes it possible to differentiate them from BSVs in the feature space \(\mathbb{R}^d\).

Proposition 2 enables us to estimate the \(\gamma\) parameter another way. This proposition ensures that similarities between pairs of implicit vectors in the feature space \(\mathbb{R}^d\) for training set \(D\) resemble similarities in the input space. Consequently, given any two dissimilarity values \(\delta_1\) and \(\delta_2\) in the input space and their corresponding similarities in the kernel space

\[
\frac{\delta_1}{\delta_2} \approx e^{-\gamma \delta_2^2} e^{-\gamma \delta_1^2} \implies \gamma \approx \frac{-\ln \left(\frac{\delta_1}{\delta_2}\right)}{\delta_2^2 - \delta_1^2}. \tag{10}
\]

These arguments lead us to propose the following heuristic to estimate the \(\gamma\) parameter.

**H2.** \(l_\gamma = \frac{-\ln \left(\frac{\delta_{\text{max}}}{\delta_{\text{min}}}\right)}{\delta_{\text{max}} - \delta_{\text{min}}}:\) Let \(q = \arg \min_{1 \leq i \leq n} \|x_i - x_{1N}\|\). Let \(\delta_{\text{min}} = \|x_q - x_{1N}\|\) and \(\delta_{\text{max}} = \frac{1}{n-1} \sum_{i \neq q} \|x_q - x_i\|\). Based on (10), we set

\[
\gamma = \frac{-\ln \left(\frac{\delta_{\text{max}}}{\delta_{\text{min}}}\right)}{\delta_{\text{max}}^2 - \delta_{\text{min}}^2}. \tag{11}
\]

For example, for the data set in Fig. 2(a), \(\gamma\) is set to 69.3 using H1.\(\gamma\) and is set to 77 using H2.\(\gamma\). Models corresponding to each of these heuristics are depicted in Fig. 3.

We summarize our unsupervised parameter estimation methods in Table I, based on the two proposed heuristics H1 and H2.

### V. EXPERIMENTAL EVALUATION

We compare our QMS2 method to the baseline algorithm QMS1 [33], DEMS [1], [20], DMMS [16], ROC SVM [19], \(\eta\)OCSVM [19], MIES [29] and its recent extension to high-dimensional data sets ADES [31]. All distance calculations were Euclidean, and we used the default parameters specified by the authors of all seven comparison methods. For QMS2, we studied different settings of its two parameters \(\nu\) and \(K\) to show its insensitivity to their values.

We also evaluated a supervised grid-search method (SGS) to estimate the optimal parameters. SGS required a labeled training set and was only evaluated to show the best performance if training labels were available. The SGS method involved two phases of coarse-grained and fine-grained search and a twofold cross validation for each pair of parameters. For the \(\nu\) parameter, its value was changed in the range [0.1, 0.6] with steps of size 0.1 for the coarse-grained search. Let the best estimate after this step be \(\hat{\nu}\), then fine-grained search covered the range \([\hat{\nu} - 0.09, \hat{\nu}]\) with steps of size 0.01. For the \(\gamma\) parameter, fine- and coarse-grained searches proposed by Hsu et al. [35] were used to make sure that the time complexity was kept low. This involved changing the value of \(\gamma\) in the range \([2^{-15}, 3]\), with steps of size 2 in the exponent for coarse-grained search. Given the best exponent \(\hat{\gamma}\), the value of \(\gamma\) was changed in the range \([2^{\hat{\gamma} - 3}, 2^{\hat{\gamma} + 3}]\) with steps of size 0.25 in the exponent for the fine-grained search.

To evaluate accuracy, we used the receiver operating characteristic curve and the corresponding area under the curve (AUC) as it is insensitive to the class imbalance problem. The reported AUC values were averaged over 100 runs. Moreover, we conducted an experiment on our Half Kernel toy example (from Section II-B) to show how sensitive SGS is to the availability of good or bad examples. The experiments were conducted on a machine with an Intel Core i7 CPU at 3.40 GHz and 16-GB RAM. The MATLAB LIBSVM toolbox [36] was used to implement the OCSVM method.

#### A. Data Sets

We ran our experiments on five benchmark anomaly detection data sets from outlier detection data sets [37], namely Cardiotocography (Cardio), SMTP (from KDDCUP99), Musk, Satimage2, and Shuttle. From the UCI Machine Learning Repository [38], we selected Breast Cancer Wisconsin (Cancer), MNIST, human activity recognition (HAR), and daily and sports activities (DSA)
data sets. For the Cancer data set, the aim was to detect malignant breast cytology as anomalies. For the MNIST data set, digit zero was considered normal and other digits were considered as anomalies. The HAR data set included sensor signals of six different activities of a group of 30 volunteers whose ages were in the range [19, 48]. In this data set, we regarded the sitting activity as normal: walking in different ways, standing, and laying were the anomalies. DSA on the other hand comprises sensor signals for 19 activities each of which was performed by four females and four males with ages in the range [20, 30]. The first 11 activities in this data set are activities like sitting, standing, lying, walking, ascending, and descending stairs. The remaining eight activities are exercising. We considered the first 11 activities as normal and the rest as anomalies. We also used the credit card fraud detection (Credit) [39] data set that contains credit card transactions of European cardholders in September 2013. The goal was to detect fraudulent transactions as anomalies. Finally, from NSL-KDD, we used HTTP data, which we refer to it as NSL-KDD-HTTP (NKH). Attacks to HTTP service were regarded as anomalies in NKH. The data descriptions, including number of unique observations and features, are summarized in Table II. For all data sets, we performed downsampling to make sure that the training set included 5% of anomalies. All data sets were rescaled in the range $[0, 1]$ based on maximum and minimum values observed in the training set. Test and training sets were randomly selected from the data with the ratio of 1 to 4. For all techniques except SGS, actual labels were only used for evaluation purposes and were not fed to the algorithms. This collection of data sets enabled us to examine our proposed heuristics on a variety of data distributions of various dimensionality. For the two ratios that appear in our assumption from Section IV-A, we verified that $\gamma_K(D^n) / \gamma_K(D) > 1$ for all the data sets. It also can be verified that $\gamma_K(D^n) / |D| < 1$, because the fraction of anomalies is 0.05.

### B. Results and Discussion

To show the sensitivity of the SGS method to bad examples of anomalies that reside in normal regions [6], we used our toy example from Section II-B; first, with good examples of anomalies that are far enough from the normal region, and second, with bad examples of anomalies that may lie inside the normal region. Fig. 5 shows that for a labeled data set that comprises bad examples of anomalies, the SGS method chooses poor parameter settings that result in building a model with a high FNR regarding the target (normal) class. However, our QMS2 technique in both cases selects a good setting and filters anomalies effectively. This example shows that if mislabelling in ground truth happens or synthetic anomalies are blindly added to a training set to tighten the normal boundary, the SGS method does not necessarily find the optimal parameter settings.

Table III reports the average AUC values and their standard deviations over 100 runs of the examined parameter estimation methods. Bold entries in Table III indicate that the corresponding method provides a statistically significant improvement compared with the regular entries. We used a Wilcoxon signed-rank test over 100 runs per data set to evaluate the statistical significance of the results. This test is explained later in this section. As the SGS method is computationally expensive, we set an upper bound of 10,000 data points for the size of each data set in all the experiments reported in Table III. The $\nu$ and $K$ parameters for QMS2, respectively, were set to 0.01 and $\lceil 0.05 \times n \rceil$, where $n$ is the size of training set. The same $K$ was used for QMS1. Later in this section, we show that QMS2 is not sensitive to other choices of the $K$ and $\nu$ parameters. For MIES, ADES, and $\eta$OCSVM, the $\nu$
AVERAGE AUC VALUES AND THEIR STANDARD DEVIATIONS OVER 100 RUNS OF QMS2 COMPARED WITH THE OTHER METHODS

<table>
<thead>
<tr>
<th>Dataset</th>
<th>QMS2</th>
<th>QMS1</th>
<th>MIES</th>
<th>ADES</th>
<th>DMMS</th>
<th>DEMS</th>
<th>ROCSVM</th>
<th>(\eta\text{OCSVM} )</th>
<th>Supervised</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSA</td>
<td>1.000</td>
<td>0.965</td>
<td>0.879</td>
<td>0.817</td>
<td>0.909</td>
<td>0.675</td>
<td>0.967</td>
<td>0.989</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>±0.000</td>
<td>±0.007</td>
<td>±0.058</td>
<td>±0.042</td>
<td>±0.006</td>
<td>±0.023</td>
<td>±0.011</td>
<td>±0.009</td>
<td>±0.000</td>
</tr>
<tr>
<td>HAR</td>
<td>0.894</td>
<td>0.889</td>
<td>-</td>
<td>0.821</td>
<td>0.889</td>
<td>0.856</td>
<td>0.878</td>
<td>0.884</td>
<td>0.893</td>
</tr>
<tr>
<td></td>
<td>±0.013</td>
<td>±0.013</td>
<td></td>
<td>±0.043</td>
<td>±0.013</td>
<td>±0.018</td>
<td>±0.014</td>
<td>±0.014</td>
<td>±0.013</td>
</tr>
<tr>
<td>MNIST</td>
<td>0.996</td>
<td>0.997</td>
<td>-</td>
<td>0.933</td>
<td>0.992</td>
<td>0.996</td>
<td>0.994</td>
<td>0.996</td>
<td>0.997</td>
</tr>
<tr>
<td></td>
<td>±0.001</td>
<td>±0.001</td>
<td></td>
<td>±0.016</td>
<td>±0.001</td>
<td>±0.001</td>
<td>±0.001</td>
<td>±0.001</td>
<td>±0.001</td>
</tr>
<tr>
<td>Cancer</td>
<td>0.995</td>
<td>0.970</td>
<td>0.954</td>
<td>0.755</td>
<td>0.995</td>
<td>0.985</td>
<td>0.884</td>
<td>0.881</td>
<td>0.949</td>
</tr>
<tr>
<td></td>
<td>±0.003</td>
<td>±0.013</td>
<td>±0.095</td>
<td>±0.103</td>
<td>±0.003</td>
<td>±0.008</td>
<td>±0.023</td>
<td>±0.023</td>
<td>±0.021</td>
</tr>
<tr>
<td>Cardio</td>
<td>0.944</td>
<td>0.917</td>
<td>0.818</td>
<td>0.874</td>
<td>0.934</td>
<td>0.799</td>
<td>0.649</td>
<td>0.649</td>
<td>0.952</td>
</tr>
<tr>
<td></td>
<td>±0.01</td>
<td>±0.013</td>
<td>±0.064</td>
<td>±0.019</td>
<td>±0.009</td>
<td>±0.024</td>
<td>±0.028</td>
<td>±0.028</td>
<td>±0.008</td>
</tr>
<tr>
<td>Credit</td>
<td>0.943</td>
<td>0.851</td>
<td>0.812</td>
<td>0.708</td>
<td>0.911</td>
<td>0.898</td>
<td>0.903</td>
<td>0.906</td>
<td>0.943</td>
</tr>
<tr>
<td></td>
<td>±0.015</td>
<td>±0.022</td>
<td>±0.088</td>
<td>±0.079</td>
<td>±0.018</td>
<td>±0.016</td>
<td>±0.015</td>
<td>±0.016</td>
<td>±0.014</td>
</tr>
<tr>
<td>NKH</td>
<td>0.989</td>
<td>0.970</td>
<td>0.863</td>
<td>0.850</td>
<td>0.988</td>
<td>0.614</td>
<td>0.796</td>
<td>0.802</td>
<td>0.963</td>
</tr>
<tr>
<td></td>
<td>±0.006</td>
<td>±0.012</td>
<td>±0.083</td>
<td>±0.077</td>
<td>±0.006</td>
<td>±0.124</td>
<td>±0.076</td>
<td>±0.072</td>
<td>±0.035</td>
</tr>
<tr>
<td>Musk</td>
<td>1.000</td>
<td>0.659</td>
<td>-</td>
<td>0.743</td>
<td>0.821</td>
<td>0.718</td>
<td>0.713</td>
<td>0.720</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>±0.000</td>
<td>±0.055</td>
<td></td>
<td>±0.044</td>
<td>±0.033</td>
<td>±0.049</td>
<td>±0.05</td>
<td>±0.054</td>
<td>±0.000</td>
</tr>
<tr>
<td>Satimage2</td>
<td>0.999</td>
<td>0.886</td>
<td>0.844</td>
<td>0.884</td>
<td>0.980</td>
<td>0.846</td>
<td>0.857</td>
<td>0.862</td>
<td>0.994</td>
</tr>
<tr>
<td></td>
<td>±0.002</td>
<td>±0.037</td>
<td>±0.06</td>
<td>±0.047</td>
<td>±0.025</td>
<td>±0.049</td>
<td>±0.046</td>
<td>±0.045</td>
<td>±0.004</td>
</tr>
<tr>
<td>Shuttle</td>
<td>0.988</td>
<td>0.954</td>
<td>0.637</td>
<td>0.748</td>
<td>0.987</td>
<td>0.848</td>
<td>0.589</td>
<td>0.590</td>
<td>0.990</td>
</tr>
<tr>
<td></td>
<td>±0.002</td>
<td>±0.005</td>
<td>±0.046</td>
<td>±0.124</td>
<td>±0.001</td>
<td>±0.013</td>
<td>±0.043</td>
<td>±0.047</td>
<td>±0.001</td>
</tr>
<tr>
<td>SMTP</td>
<td>0.920</td>
<td>0.840</td>
<td>0.735</td>
<td>0.786</td>
<td>0.836</td>
<td>0.836</td>
<td>0.654</td>
<td>0.671</td>
<td>0.812</td>
</tr>
<tr>
<td></td>
<td>±0.095</td>
<td>±0.123</td>
<td>±0.132</td>
<td>±0.134</td>
<td>±0.132</td>
<td>±0.108</td>
<td>±0.118</td>
<td>±0.12</td>
<td>±0.124</td>
</tr>
<tr>
<td>Average</td>
<td>0.970</td>
<td>0.900</td>
<td>0.727</td>
<td>0.811</td>
<td>0.931</td>
<td>0.825</td>
<td>0.808</td>
<td>0.814</td>
<td>0.954</td>
</tr>
<tr>
<td></td>
<td>±0.013</td>
<td>±0.027</td>
<td>±0.078</td>
<td>±0.066</td>
<td>±0.022</td>
<td>±0.039</td>
<td>±0.039</td>
<td>±0.039</td>
<td>±0.02</td>
</tr>
</tbody>
</table>

* The method failed to find the setting.

CPU-TIMES (IN SECONDS) FOR ESTIMATING PARAMETERS USING QMS2 COMPARED WITH THE OTHER METHODS

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSA</td>
<td>QMS2</td>
</tr>
<tr>
<td></td>
<td>QMS1</td>
</tr>
<tr>
<td>MNIST</td>
<td>C An cer</td>
</tr>
<tr>
<td></td>
<td>Cardio</td>
</tr>
<tr>
<td></td>
<td>Credit</td>
</tr>
<tr>
<td></td>
<td>N KH</td>
</tr>
<tr>
<td></td>
<td>Musk</td>
</tr>
<tr>
<td></td>
<td>Satimage2</td>
</tr>
<tr>
<td></td>
<td>Shuttle</td>
</tr>
<tr>
<td></td>
<td>SMTP</td>
</tr>
<tr>
<td></td>
<td>Average</td>
</tr>
</tbody>
</table>

* The method failed to find the setting.

The parameter is user-defined, so we set it to 0.05 during the experiments. The results in Table III show that our proposed method QMS2 outperforms the semisupervised (MIES and ADES) and unsupervised (QMS1, DEMS, DMMS, ROCSVM, and \(\eta\text{OCSVM} \) parameter estimation methods. For the high-dimensional data sets, namely DSA, HAR, and Mask, MIES...
fails to find the $\gamma$ parameter, because it detects all training observations as edge samples. QMS2 on average works better than SGS and provides a statistically significant improvement in the Cancer, NKH, and SMTP data sets. In terms of variation of accuracy over 100 runs, again our technique outperforms the other techniques. These findings support our arguments regarding the toy example in Fig. 5 and show that QMS2 is less sensitive to the quality of the training set compared with the other techniques. The bottom row of Table III makes the case for QMS2, whose average performance (0.97) over 11 data sets exceeds the next best method, SGS (0.95). Although this positive difference seems slight, we remind you that SGS uses cross validation and requires a labeled data set to find the optimal parameter settings, whereas QMS2 is unsupervised, i.e., learning the best parameters is performed without having access to the labels (labels were used only for evaluation purposes).

Table IV lists CPU-times for the nine techniques. QMS2 is one to three orders of magnitude faster than all of the other methods except QMS1. QMS1 is similar to QMS2 and both of them are roughly 70 times faster than SGS. QMS2 has a slightly higher CPU-time than QMS1 because of the time

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Train Time (sec)</th>
<th>Test Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSA</td>
<td>0.15647</td>
<td>0.00001</td>
</tr>
<tr>
<td>HAR</td>
<td>0.11279</td>
<td>0.00005</td>
</tr>
<tr>
<td>MNIST</td>
<td>0.45396</td>
<td>0.00013</td>
</tr>
<tr>
<td>Cancer</td>
<td>0.00994</td>
<td>0.00000</td>
</tr>
<tr>
<td>Cardio</td>
<td>0.01841</td>
<td>0.00001</td>
</tr>
<tr>
<td>Credit</td>
<td>0.17425</td>
<td>0.00002</td>
</tr>
<tr>
<td>NKH</td>
<td>0.00218</td>
<td>0.00000</td>
</tr>
<tr>
<td>Musk</td>
<td>0.31996</td>
<td>0.00028</td>
</tr>
<tr>
<td>Satimage2</td>
<td>0.00967</td>
<td>0.00001</td>
</tr>
<tr>
<td>Shuttle</td>
<td>0.04446</td>
<td>0.00000</td>
</tr>
<tr>
<td>SMTP</td>
<td>0.00016</td>
<td>0.00000</td>
</tr>
<tr>
<td>Average</td>
<td>0.11757</td>
<td>0.00005</td>
</tr>
</tbody>
</table>

* The method failed to find the setting.

Table VI

<table>
<thead>
<tr>
<th>Method</th>
<th>QMS2</th>
<th>QMS1</th>
<th>MIES</th>
<th>ADES</th>
<th>DMMS</th>
<th>DEMS</th>
<th>ROCSVM</th>
<th>$\eta$OCSVM</th>
<th>SGS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train CPU-time</td>
<td>0.55490</td>
<td>2.98664</td>
<td>0.19048</td>
<td>5.15022</td>
<td>5.02370</td>
<td>0.20623</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test CPU-time</td>
<td>0.00003</td>
<td>0.00001</td>
<td>0.00003</td>
<td>0.00003</td>
<td>0.00001</td>
<td>0.00003</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table V

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Time (sec)</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSA</td>
<td>0.15647</td>
<td>QMS2</td>
</tr>
<tr>
<td>HAR</td>
<td>0.11279</td>
<td>QMS1</td>
</tr>
<tr>
<td>MNIST</td>
<td>0.45396</td>
<td>MIES</td>
</tr>
<tr>
<td>Cancer</td>
<td>0.00994</td>
<td>ADES</td>
</tr>
<tr>
<td>Cardio</td>
<td>0.01841</td>
<td>DMMS</td>
</tr>
<tr>
<td>Credit</td>
<td>0.17425</td>
<td>DEMS</td>
</tr>
<tr>
<td>NKH</td>
<td>0.00218</td>
<td>ROCSVM</td>
</tr>
<tr>
<td>Musk</td>
<td>0.31996</td>
<td>$\eta$OCSVM</td>
</tr>
<tr>
<td>Satimage2</td>
<td>0.00967</td>
<td>SGS</td>
</tr>
</tbody>
</table>

Table VI

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (AUC)</th>
<th>Parameter estimation CPU-time</th>
<th>Train CPU-time</th>
<th>Test CPU-time</th>
</tr>
</thead>
<tbody>
<tr>
<td>QMS1</td>
<td>0.65</td>
<td>0.004</td>
<td>0.004</td>
<td>0.004</td>
</tr>
<tr>
<td>MIES</td>
<td>0.60</td>
<td>0.003</td>
<td>0.003</td>
<td>0.003</td>
</tr>
<tr>
<td>ADES</td>
<td>0.66</td>
<td>0.003</td>
<td>0.003</td>
<td>0.003</td>
</tr>
<tr>
<td>DMMS</td>
<td>0.61</td>
<td>0.004</td>
<td>0.004</td>
<td>0.004</td>
</tr>
<tr>
<td>DEMS</td>
<td>0.65</td>
<td>0.002</td>
<td>0.002</td>
<td>0.002</td>
</tr>
<tr>
<td>ROCSVM</td>
<td>0.60</td>
<td>0.003</td>
<td>0.003</td>
<td>0.003</td>
</tr>
<tr>
<td>$\eta$OCSVM</td>
<td>0.60</td>
<td>0.003</td>
<td>0.003</td>
<td>0.003</td>
</tr>
<tr>
<td>SGS</td>
<td>0.61</td>
<td>0.004</td>
<td>0.004</td>
<td>0.004</td>
</tr>
</tbody>
</table>

* The method failed to find the setting.

* The number of unequal observations is not enough to compute the p-value, but given the rank, the difference is not statistically significant.
required to find $\delta_{\text{max}}$, but the difference is negligible. For MNIST, which has the highest number of features, QMS1 and QMS2 are much faster than the other techniques, all of which require multiple runs of the OCSVM solver. Since $n \gg d$ in our experiments, the time complexity of SGS using traditional matrix inversion is $O(n^3)$ [40]. MIES and ADES also have time complexity equal to $O(n^3)$ [29]. In contrast, the most expensive part of QMS1 and QMS2, i.e., finding the $K$-NN, requires $O(n(d+k))$.

Table V shows that the average running-time (over the 100 runs) of training and testing a model given the parameters found by each of the nine techniques. Table V demonstrates that building a model using the settings estimated by QMS2 can be done roughly 2 to 50 times faster than training a model using the settings estimated via the rest of the techniques. Thus, preprocessing of the data sets and setting the values of hyperparameters using QMS2 speed up the training phase. Moreover, QMS2 results in models with fewer support vectors than the other techniques, which speed up the test-time reported per test sample in Table V.

To identify the statistical significance of the results between QMS2 and the rest of the methods in terms of accuracy and running-time, we conducted a Wilcoxon signed-rank test with a level of significance of $\alpha$. In each comparison, the aim is to investigate to what extent the null hypothesis $H_0$ can be rejected; therefore, the improvement is significant at the level $\alpha = 0.05$. Table VI shows that in terms of accuracy, our QMS2 method provides a statistically significant improvement over all the other methods except SGS. The returned $p$-values less than $\alpha$ indicates that the null hypothesis $H_0$ can be rejected; therefore, the improvement is significant at the level $\alpha = 0.05$. Table VI shows that in terms of the time required for parameter estimation, Table VI shows that QMS2 provides a statistically significant improvement over all the other techniques except QMS1. But again, the difference between QMS1 and QMS2 is negligible. In terms of the training time of models generated using the hyperparameters estimated by the examined techniques, QMS2 provides a statistically significant improvement over all except ADES, DEMS, and SGS, while for the testing time, it outperforms MIES, DMMS, and SGS.

Now, we discuss the effect of choosing different values for the $\nu$ and $K$ parameters for QMS2. We explored values for $K$ in the range $(\lfloor 0.05 \times n \rfloor, \lfloor 0.3 \times n \rfloor]$ with steps of size 0.05 and $\nu$ in the range $[0.0025, 0.02]$ with steps of size 0.0025. We ran
the same experiments on the same data sets. Fig. 6 shows that the choice of these two parameters does not affect the findings (the vertical scale, floor to ceiling, in all ten graphs is 0.02), which confirms that QMS2 is insensitive to the choice of these parameters. If the data set has a dense cluster of anomalies instead of point anomalies, the $K$ parameter should be large enough to be able filter anomalies in the preprocessing step. However, the choice of $K$ is very straightforward, e.g., if the aim is to identify clusters of anomalies populated less than $f\%$ of the data, then $K = \lceil f \times n \rceil$. We should also consider the cost of increasing the value of $K$ in terms of running-time, which is reported in Fig. 7. Fig. 7 shows that this cost is negligible.

VI. CONCLUSION

We proposed the QMS2 algorithm to estimate optimal parameter settings for OCSVMs without the need for ground truth labels or exhaustive grid-search over the parameter space. QMS2 combines $K$-NNs and OCSVMs in a novel way that inherits the best features of both techniques and overcomes their shortcomings by using both density and distance measures to filter-out anomalies. Our experimental evaluation showed that QMS2 outperformed eight other existing supervised, semisupervised, and unsupervised parameter estimation approaches in terms of accuracy and run time. QMS2 had similar or better accuracy than the supervised grid-search method, while being 70 times faster on average on the examined data sets. The QMS2 method was robust to the variety of different types of anomalies. In the future, we aim to use our heuristic $H_1.v$ to speed up the training of OCSVMs by providing a better initialization of the $\alpha$ vector in (2); this would result in a faster convergence toward the global minimum.

REFERENCES


Zahra Ghafoori is currently pursuing the Ph.D. degree with the Department of Computing and Information Systems, University of Melbourne, Melbourne, VIC, Australia.

She is currently involved in developing robust and efficient techniques for unsupervised anomaly detection in complex and dynamic environments. Her current research interests include data mining, pattern recognition, cyber-security, and anomaly detection.

Sarah M. Erfani received the master’s and Ph.D. degrees from the School of Computing and Information Systems, University of Melbourne, Melbourne, VIC, Australia.

She is currently a Lecturer with the School of Computing and Information Systems, University of Melbourne. Her current research interests include machine learning, large-scale data mining, cyber-security, and data privacy.

Sutharshan Rajasegarar received the Ph.D. degree from the University of Melbourne, Melbourne, VIC, Australia, in 2009.

He is currently a Lecturer with the School of Information Technology, Deakin University, Geelong, VIC, Australia. His current research interests include anomaly/outlier detection, distributed machine learning, pattern recognition, signal processing, Internet of Things, and wireless communications.

James C. Bezdek (LF’10) received the Ph.D. degree in applied mathematics from Cornell University, Ithaca, NY, USA, in 1973.

Dr. Bezdek is a past President of North American Fuzzy Information Processing Society, International Fuzzy Systems Association (IFSA), and the IEEE Computational Intelligence Society (CIS), a Founding Editor of the International Journal of Approximate Reasoning and the IEEE TRANSACTIONS ON FUZZY SYSTEMS, a Life Fellow of IFSA, and a recipient of the IEEE 3rd Millennium, an IEEE CIS Fuzzy Systems Pioneer Award, IEEE Rosenblatt technical field medals, and the Kampe de Feriet Award.

Shanika Karunasekera received the B.Sc. degree in electronics and telecommunications engineering from the University of Moratuwa, Moratuwa, Sri Lanka, in 1990, and the Ph.D. degree in electrical engineering from the University of Cambridge, Cambridge, U.K., in 1995.

From 1995 to 2002, she was a Software Engineer and a Distinguished Member of Technical Staff with Lucent Technologies, Bell Labs Innovations, Murray Hill, NJ, USA. In 2003, she joined the University of Melbourne, Melbourne, VIC, Australia, where she is currently a Professor with the Department of Computing and Information Systems. Her current research interests include distributed system engineering, distributed data-mining, and social media analytics.

Christopher Leckie has over two decades of research experience in artificial intelligence and data mining, having led research teams at Telstra Research Laboratories, National ICT Australia, Melbourne, VIC, Australia, and the University of Melbourne, Melbourne. He is currently a Professor with the Department of Computing and Information Systems, University of Melbourne. He has led to a range of operational systems used in industry. He has published over 200 articles in leading international conferences and journals. His current research interests include using data mining for anomaly detection, fault diagnosis, cyber-security, and the life sciences.