Imbalanced SVM-Based Anomaly Detection Algorithm for Imbalanced Training Datasets

GuiPing Wang, JianXi Yang, and Ren Li

Abnormal samples are usually difficult to obtain in production systems, resulting in imbalanced training sample sets. Namely, the number of positive samples is far less than the number of negative samples. Traditional Support Vector Machine (SVM)-based anomaly detection algorithms perform poorly for highly imbalanced datasets: the learned classification hyperplane skews toward the positive samples, resulting in a high false-negative rate. This article proposes a new imbalanced SVM (termed ImSVM)-based anomaly detection algorithm, which assigns a different weight for each positive support vector in the decision function. ImSVM adjusts the learned classification hyperplane to make the decision function achieve a maximum GMean measure value on the dataset. The above problem is converted into an unconstrained optimization problem to search the optimal weight vector. Experiments are carried out on both Cloud datasets and Knowledge Discovery and Data Mining datasets to evaluate ImSVM. Highly imbalanced training sample sets are constructed. The experimental results show that ImSVM outperforms over-sampling techniques and several existing imbalanced SVM-based techniques.

Keywords: Anomaly detection, Decision function, GMean, Imbalanced training sample set, Support vector machine (SVM).

Manuscript received Nov. 30, 2016; revised June 22, 2017; accepted July 3, 2017.

GuiPing Wang (corresponding author, w_guiping@163.com), JianXi Yang (yjx_cqjtu@163.com), and Ren Li (liren902@163.com) are with the College of Information Science and Engineering, Chongqing Jiaotong University, China.

I. Introduction

In order to effectively represent the state of an observed production system, many state variables (referred to as performance metrics) are measured and collected. A vector containing the sampling values of all these variables at one point in time forms a sample of the observed system. Anomaly detection is a function that detects abnormal states characterized by samples of an observed system.

In production systems, it is usually easy to collect normal samples (that is, negative samples). However, although abnormal samples (that is, anomalies or positive samples) occur frequently, their frequency is still far less than that of normal samples. Consequently, abnormal samples usually are not easy to collect. In addition, for a newly deployed production system, the training dataset merely includes normal samples initially. Along with abnormal samples being detected and sent to human operators for confirmation, the detection system gradually accumulates some abnormal samples. In these cases, the number of abnormal samples is far less than that of normal samples in the training sample set.

The former class of samples is usually called the minority class, while the latter is called the majority class. For example, abnormal samples account for < 5%. Such a sample set is often called an *imbalanced dataset*. When training a model for anomaly detection, one challenge is to cope with imbalanced training datasets.

Classification techniques based on Support Vector Machine (SVM) are widely adopted in domains such as anomaly detection [1]. However, traditional SVM-based anomaly detection techniques perform poorly for imbalanced datasets. The underlying reason is that the decision function obtained in SVM is determined only by the positive and negative support vectors. Since the

number of positive support vectors is also far less than that of negative support vectors for a highly imbalanced training set, SVM tends to learn a classification hyperplane that is too close to and skewed toward the positive samples (as detailed in Section IV). Thus, it is easier to misclassify positive samples as negative ones (that is, false negatives) and achieve a high false-negative rate. Moreover, the consequence of misclassifying a positive sample is usually more serious; for example, falsely classifying a true anomaly as a normal state may bring disastrous consequences.

To this end, this article proposes a new imbalanced SVM (ImSVM)-based anomaly detection algorithm. ImSVM assigns a different weight for each positive support vector in the decision function obtained by traditional SVM, and adjusts the learned original classification hyperplane to make the decision function achieve a maximum GMean measure value on the dataset. The optimal weight vector is solved through an unconstrained optimization problem. A series of experiments are carried out on both Cloud datasets and Knowledge Discovery and Data Mining (KDD) datasets to evaluate ImSVM. Highly imbalanced training sample sets are constructed. The experimental results show that ImSVM outperforms over-sampling techniques and several existing imbalanced SVM-based techniques.

II. Related Work

Anomaly detection [1] is an important research problem that has been widely studied within various research areas and application domains, including intrusion detection for network security [2], [3], fraud detection of credit card transactions [4], and fault diagnosis for distributed systems [5]. Chandola and others [1] survey the literature in the anomaly detection domain. They provide an extensive overview on the research in the literature.

Support Vector Machine-based algorithms are widely adopted in anomaly detection. Fu and others [6] put forward a self-evolving framework for anomaly detection to enhance the dependability of Cloud computing platforms. This framework combines two SVMs (that is, one-class, and two-class). Through experiments they verify that SVM starts to perform reasonably well on imbalanced datasets once the percentage of abnormal samples reaches 10%.

Note that, this article focuses only on binary classification. In an imbalanced training sample set, since the number of positive samples (abnormal samples) is far less than that of negative samples (normal samples), the class of positive samples is also referred to as the minority class, while the class of negative samples is referred to as the majority class.

The framework proposed in [6] solves the problem of imbalanced datasets by switching between two SVMs. However, since both of these SVMs perform poorly when the proportion of abnormal samples is $< 10\%$, the framework essentially does not solve the problem.

The techniques presented in the literature to solve the challenge of imbalanced training sample sets can be divided into two categories: *reconstructing the sample sets* and *improving the detection algorithms*.

The first category of techniques reduces the imbalance between samples of different classes through under-sampling or over-sampling techniques. Under-sampling techniques remove samples from the majority class to balance the dataset. The main defect of under-sampling is that it may remove potentially useful data and reduce the size of the training sample set [4]. Over-sampling techniques balance the dataset through replicating or synthesizing the samples of the minority class. Despite not losing any data, over-sampling may introduce noisy artificial samples, lead to over-fitting, as well as introduce additional computational cost [4], [7].

The synthetic minority over-sampling technique (SMOTE) [8] is a typical over-sampling technique. SMOTE selects a sample (denoted as $\boldsymbol{x}$) from the minority class and computes $\boldsymbol{x}$'s K-nearest neighbors. Then, SMOTE randomly selects a neighbor (denoted as $\boldsymbol{x}_1$) and synthesizes a sample randomly lying in the line segment between $\boldsymbol{x}$ and $\boldsymbol{x}_1$ as a new synthetic sample. Yang and others [9] improve SMOTE by adaptively excluding some neighbors far away from $\boldsymbol{x}$, and propose Adaptive SMOTE (ASMOTE). Castro and others [10] combine under-sampling and over-sampling techniques to reduce the imbalance in the dataset by first removing noisy examples of the majority class, and then generating new synthetic examples of the minority class.

The second category of techniques adopts certain methods to improve the classifiers in order to make the detection models more advantageous to the classification of the minority class. These methods include adjusting the cost function of each class of samples, one-class classification, and adjusting the classification boundary.

Imam and others [11] propose an extended SVM, z-SVM, for imbalanced datasets. z-SVM assigns the same weight to each positive support vector in the decision function. z-SVM orients the trained decision boundary of SVM to maintain a good margin between the decision boundary and each class of samples. However, since the position, role, and significance of each support vector is

https://doi.org/10.4218/etrij.17.0116.0879

different, assigning the same weight to each positive support vector cannot achieve the desired effect in improving SVM, which is confirmed in Section V. Yang and others [12] propose an extended SVM, μSVM, which is similar to z-SVM.

Introducing different misclassification cost for each class of samples is another kind of important technique for imbalanced datasets, which derives cost-sensitive SVM [13]. However, it is usually difficult (lack of defined strategy or guidelines) to determine the precise misclassification cost for each class of samples in practice.

Along with the advent of advanced machine learning techniques, these techniques are also introduced into SVM to deal with imbalanced sample sets, including scaling kernel-based SVM [14], [15] and ensemble learning of SVM [16]. However, it is usually difficult to implement these techniques. Therefore, this article will not discuss these techniques.

Some research combined these two categories of techniques to cope with imbalanced datasets. For example, Akbani and others [17] combine a variant of SMOTE algorithm (that is, over-sampling) by using different penalty constants for different classes of samples (that is, cost-sensitive SVM).

In addition, since the common evaluation measures (for example, false negative rate, false positive rate, and accuracy rate) are not applicable for evaluating the performance of anomaly detection algorithms on imbalanced datasets, some measures including F_1-measure [15], GMean [11], [18] are introduced in the literature.

Compared with the research in the literature [8], [9], [11], [13] this article proposes a relatively simple but powerful imbalanced SVM (ImSVM)-based anomaly detection algorithm to cope with imbalanced datasets.

III. Preliminaries

1. Anomaly Detection Algorithm Based on C-SVM

Notations: a lowercase letter in italic format (b) indicates a scalar; a lowercase letter in italic and bold format ($\boldsymbol{x}$) indicates a vector; in addition, the vectors are all column ones. Note that, a vector in Hilbert space ($\mathcal{H}$) is represented by a bold and non-italic lowercase letter ($\mathbf{x}$).

SVM [19] is essentially a supervised learning technique. It is widely used in classification [20], regression [21], and other techniques. This article only focuses on anomaly detection problems (that is, classification).

In essence, SVM is a model for binary classification. SVM learns a linear classifier (that is, a classification hyperplane, $(\boldsymbol{w}\cdot\boldsymbol{x}) + b = 0$) with a maximum margin between two support hyperplanes in Euclidean space, as shown in Fig. 1(a). By introducing kernel methods and learning a linear classifier in Hilbert space ($\mathcal{H}$), SVM can also implement nonlinear classification in Euclidean space, R^n. Maximizing the margin is formalized as a convex quadratic programming problem.

If the training dataset can be separated linearly, as illustrated in Fig. 1(a), a linear classifier (that is, a hyperplane) can be learned by maximizing the hard margin, which is formalized as $2/\|\boldsymbol{w}\|$. Further, maximizing $2/\|\boldsymbol{w}\|$ is equivalent to minimizing $\|\boldsymbol{w}\|^2/2$. The obtained linear classifier is referred to as a linear separable SVM, which is also called an SVM with a hard margin.

If the training dataset is linearly separable except for some outliers (that is, approximately linearly separable), as illustrated in Fig. 1(b), a linear classifier can also be learned by introducing the slack variables ξ_i and maximizing the soft margin. The obtained linear classifier is referred to as a linear SVM, which is also called an SVM with soft margin.

If the training dataset cannot be linearly separated in Euclidean space, by applying kernel methods it is possible to map the input space $\boldsymbol{x} \in R^n$ to a Hilbert space $\mathbf{x} \in \mathcal{H}$ and learn a linear SVM in Hilbert space, as shown in Figs. 1(c) and 1(d). Note that, the map function is denoted as Φ. More important, a linear SVM (a hyperplane) in Hilbert space is equivalent to a nonlinear SVM (a hypersurface) in Euclidean space. By applying kernel methods, the basic operations of inner products in SVM are converted into simply computing the kernel function directly without knowing the map function. Namely, the operation process does not need to really map the samples into $\mathcal{H}$ space.

The basic standard form of SVM is C-SVM [22]. The prefix "C-" introduces the penalty parameter C, which is a trade-off between two conflicting goals: maximization of margin and minimization of the training error. The primal optimization problem of C-SVM is

$$\min_{\boldsymbol{w},b,\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + \boldsymbol{C}\sum_{i=1}^{l}\xi_i, \tag{1}$$

$$\text{s.t.}\, y_i((\boldsymbol{w}\cdot\boldsymbol{x}_i) + b) \geq 1 - \xi_i, i = 1, 2, \ldots, l, \tag{2}$$

$$\xi_i \geq 0, i = 1, 2, \ldots, l, \tag{3}$$

where $\{(\boldsymbol{x}_i, y_i)\}$ is the training dataset, $i = 1, 2, \ldots, l$; l is the number of samples in the dataset; $\boldsymbol{w}$ is the normal vector of the classification hyperplane, b (scalar) is the corresponding bias, and ξ_i are slack variables that indicate how far a particular sample is from its correct side of the support hyperplane.

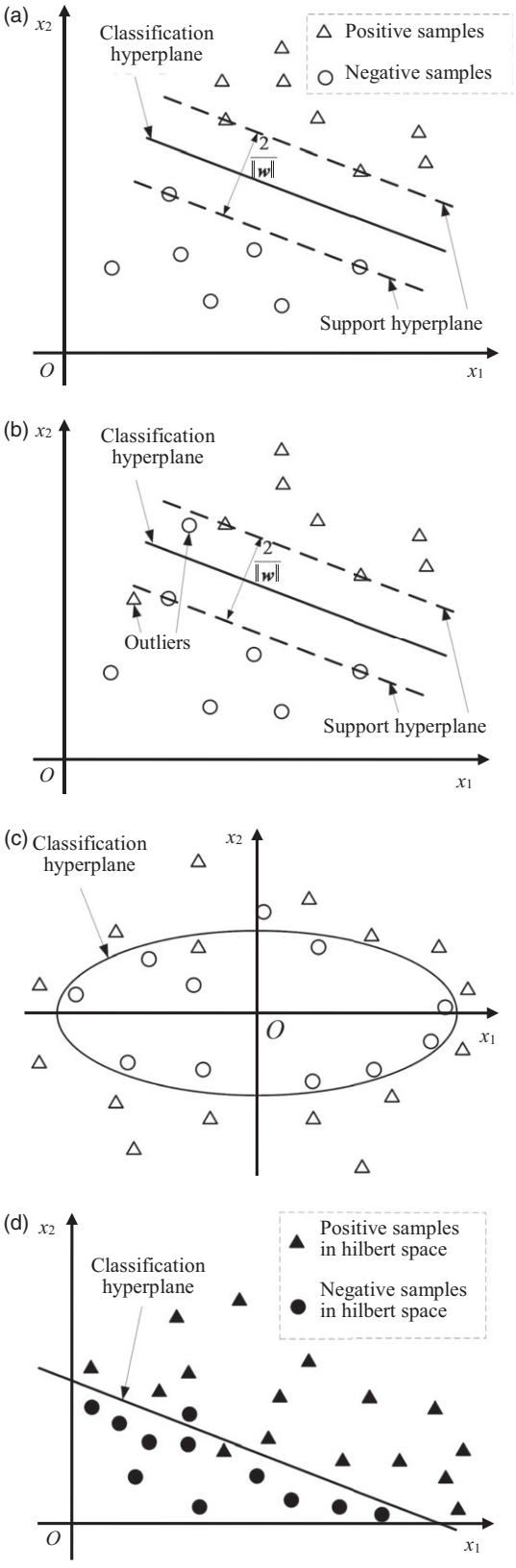

Fig. 1. Basic principles of Support Vector Machine (SVM): (a) linear separable SVM, (b) linear SVM, (c) linear inseparable dataset, and (d) linear SVM in Hilbert space (equivalent to a nonlinear SVM in Euclidean space).

By applying Lagrange duality and introducing kernel methods, the following dual optimization problem is obtained:

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{j=1}^{l} \alpha_j, \tag{4}$$

$$\text{s.t.} \sum_{i=1}^{l} y_i \alpha_i = 0, \tag{5}$$

$$0 \le a_i \le C, i = 1, 2, \ldots, l, \tag{6}$$

where $K(\cdot, \cdot)$ is the kernel function associated with Φ, and $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_l)^T$ is the Lagrange multiplier vector that needs to be solved, where the component α_i is the Lagrange multiplier associated with the sample $(\boldsymbol{x}_i, y_i)$.

The basic idea of the anomaly detection algorithm based on *C*-SVM is to learn a classifier, and then classify a new sample as normal or abnormal. The steps of this algorithm are detailed as follows.

Algorithm 1 The anomaly detection algorithm based on *C*-SVM

Input: the training dataset $T = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\}$, where $\boldsymbol{x}_i \in R^n, y_i \in \mathcal{Y} = \{1, -1\}$ is the label associated with $\boldsymbol{x}_i$, $i = 1, 2, \ldots, l$; the newly collected sample, $\boldsymbol{x}_{\text{new}}$.

Output: the solved decision function $f(\boldsymbol{x})$ and the label y_{new} related to $\boldsymbol{x}_{\text{new}}$.

Step 1: Choose an appropriate kernel $K(\boldsymbol{x}, \boldsymbol{x}')$ (usually a Gaussian kernel function is chosen), and choose a penalty parameter $C > 0$.

Step 2: Construct the convex quadratic programming problem using (4)–(6), solve the problem, and obtain a solution.

$$\boldsymbol{\alpha}^* = [\alpha_1^* \ \alpha_2^* \cdots \alpha_l^*]^T. \tag{7}$$

Step 3: Compute the normal vector $\boldsymbol{w}^*$ according to the following equation:

$$\boldsymbol{w}^* = \sum_{i=1}^{l} \alpha_i^* y_i \Phi(\boldsymbol{x}_i). \tag{8}$$

Note that, since the map function Φ is usually unknown, it may be impossible to compute $\boldsymbol{w}^*$ directly. Fortunately, only the decision function is needed generally. Therefore, the normal vector does not need to be explicitly calculated.

Step 4: Compute b^*. Choose a component (denoted as (α_j^*)) of $\boldsymbol{\alpha}^*$, $\alpha_j^* \in (0, C)$, and compute

$$b^* = y_j - \sum_{i=1}^{l} y_i \alpha_i^* K(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{9}$$

Since the solution of b is not unique, the average value of all solutions is computed in practice.

https://doi.org/10.4218/etrij.17.0116.0879

Step 5: Construct the following decision function

$$f(\boldsymbol{x}) = \text{sgn}(g(\boldsymbol{x})), \tag{10}$$

where

$$g(\boldsymbol{x}) = \sum_{i=1}^{l} y_i \alpha_i^* K(\boldsymbol{x}_i, \boldsymbol{x}) + b^*. \tag{11}$$

Step 6: Compute $y_{\text{new}} = f(\boldsymbol{x}_{\text{new}})$. If y_{new} equals to +1, the new sample $\boldsymbol{x}_{\text{new}}$ is classified as an anomaly (an abnormal state); otherwise, it is classified as a normal state.

If a component (denoted as α_i^*) of $\boldsymbol{\alpha}^*$ is nonzero, then the input vector $\boldsymbol{x}_i$ associated with $(\boldsymbol{x}_i, y_i)$ is referred to as a *support vector*. Otherwise, $\boldsymbol{x}_i$ is a nonsupport vector.

2. Evaluation Measures for Detection Algorithms

For an observed system, an abnormal state detected by an anomaly detection algorithm is called a positive, while a detected normal state is called a negative. Combined with the actual normal or abnormal state of the observed system, the detection results fall into the following four categories: False positive (F_{P}), False negative (F_{N}), True positive (T_{P}), True negative (T_{N}). They constitute the confusion matrix illustrated in Fig. 2.

This article introduces the following four measures to evaluate the performance of the anomaly detection algorithms involved in the experiments:

1) *Sensitivity* is defined as the ratio of the accurately detected anomalies (T_{P}) to the actual anomalies (T_{P} and F_{N}).

$$Sensitivity = T_{\text{P}}/(T_{\text{P}} + F_{\text{N}}). \tag{12}$$

2) *Specificity* is defined as the ratio of the accurately detected normal states (T_{N}) to the actual normal states (F_{P} and T_{N}).

$$Specificity = T_{\text{N}}/(F_{\text{P}} + T_{\text{N}}). \tag{13}$$

3) *Precision* is defined as the ratio of the accurately detected anomalies (T_{P}) to the detected anomalies (T_{P} and F_{P}).

		Actual state	
		Anomaly	Normal
Detection results	Anomaly (positive)	**True positive (T_{p})**	**False positive (F_{p})**
	Normal (negative)	**False negative (F_{N})**	**True negative (T_{N})**

Fig. 2. Confusion matrix.

$$Precision = T_{\text{P}}/(F_{\text{P}} + T_{\text{P}}). \tag{14}$$

4) F_1-*Measure* is defined as the harmonic mean of recall (denoted as R) and precision (abbreviated as P), where recall is just equivalent to sensitivity. Therefore,

$$F_1 = 2PR/(P + R), P = T_{\text{P}}/(T_{\text{P}} + F_{\text{N}}). \tag{15}$$

IV. Proposed IMSVM Algorithm

1. Basic Idea of ImSVM

In the input Euclidean space R^n, the classification hyperplane obtained in SVM is determined by the normal vector ($\boldsymbol{w}$) and the bias (b). It is known from (8) that $\boldsymbol{w}$ is determined only by support vectors, which is also illustrated by the following equation:

$$\boldsymbol{w} = \sum_{\boldsymbol{x}_p \in SV; y_p > 0} \alpha_p y_p \boldsymbol{x}_p + \sum_{\boldsymbol{x}_n \in SV; y_n < 0} \alpha_n y_n \boldsymbol{x}_n, \tag{16}$$

where SV is the set of support vectors. The bias b is determined by (9). Therefore, the obtained decision function can be expressed as $f(\boldsymbol{x}) = \text{sgn}(g(\boldsymbol{x}))$, where

$$\begin{aligned} g(\boldsymbol{x}) = &\sum_{\boldsymbol{x}_p \in SV; y_p > 0} \alpha_p y_p (\boldsymbol{x} \cdot \boldsymbol{x}_p) \\ &+ \sum_{\boldsymbol{x}_n \in SV; y_n < 0} \alpha_n y_n (\boldsymbol{x} \cdot \boldsymbol{x}_n) + b\,, \end{aligned} \tag{17}$$

$(\boldsymbol{x} \cdot \boldsymbol{x}_p)$ and $(\boldsymbol{x} \cdot \boldsymbol{x}_n)$ represent the inner products of two samples.

Note that, the above two equations also hold in Hilbert space, as long as $\boldsymbol{x}$ is replaced with $\Phi(\boldsymbol{x})$, and $(\boldsymbol{x}, \boldsymbol{x}')$ is replaced with $K(\boldsymbol{x}, \boldsymbol{x}')$.

For a newly collected sample $\boldsymbol{x}_{\text{new}}$, if $f(\boldsymbol{x}_{\text{new}}) > 0$, then it is classified as a positive sample; otherwise, it is classified as a negative sample.

In a highly imbalanced training set, the negative samples heavily outnumber the positive samples; the negative support vectors also heavily outnumber the positive support vectors. Therefore, the negative samples and the negative vectors dominate in the decision function (17). Concretely speaking, in the objective function (1) of SVM, the penalty of the negative samples heavily exceeds that of the positive samples. Therefore, the obtained solution tends to maximize the margin between the negative samples and the classification hyperplane, thus making SVM tend to learn a classification hyperplane that is too close to and skewed toward the positive samples [17], as illustrated in Fig. 3(a).

The skewed boundary may enhance the possibility of misclassifying some positive samples as negative ones (that is, classifying a true anomaly as a normal state), thus increasing the false-negative rate. As an illustrative example, no negative sample is falsely classified in Fig. 3(a), while two positive samples are falsely classified as negative ones (false negatives). Moreover, the consequence of misclassifying positive samples is usually more serious. Therefore, an intuitional improvement of SVM is to move the original classification hyperplane to a more proper position.

Note that, although Fig. 3 takes a linear SVM in Euclidean space as an illustrative example, the above conclusions also hold for linear SVMs in Hilbert space, which is equivalent to nonlinear SVMs in Euclidean space.

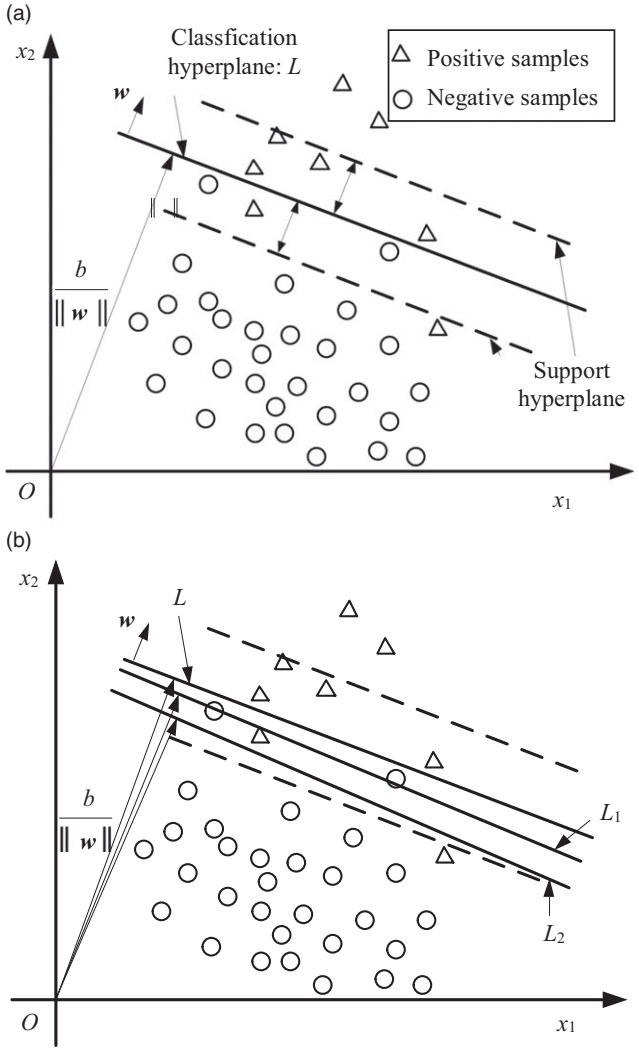

Fig. 3. Geometry interpretation of imbalanced Support Vector Machine (ImSVM): (a) original classification hyperplane solved by C-SVM skews towards the positive samples and (b) adjusting the classification hyperplane.

The basic idea of ImSVM is that for the original classification hyperplane solved by C-SVM, ImSVM adjusts the position of the hyperplane by adjusting the parameters $\boldsymbol{w}$ and b to make a certain evaluation measure (usually more applicable for imbalanced datasets) of the detection model achieve an optimal value on a training sample set.

As illustrated in Figs. 3(a) and (b), for the original classification hyperplane (denoted as L) solved by C-SVM, a new hyperplane (L_1) is obtained after adjusting $\boldsymbol{w}$; another new hyperplane (L_2) is obtained after adjusting b. Note that, $b/\|\boldsymbol{w}\|$ represents the distance between the origin and the hyperplane. Further, adjusting b can be implemented by adjusting $\boldsymbol{w}$. According to (16), the adjustment of $\boldsymbol{w}$ can be implemented only by adjusting the weights of the positive support vectors in (16). Note that, the support vectors lie in different regions; their roles and significance are also different [22]. Therefore, each positive support vector in the decision function should be assigned a different weight, which results in a new decision function, $f(\boldsymbol{x}, \boldsymbol{\lambda}) = \mathrm{sgn}(g(\boldsymbol{x}, \boldsymbol{\lambda}))$, where

$$g(\boldsymbol{x}, \boldsymbol{\lambda}) = \sum_{\boldsymbol{x}_p \in SV; y_p > 0} \lambda_p \alpha_p y_p K(\boldsymbol{x}, \boldsymbol{x}_p) + \sum_{\boldsymbol{x}_n \in SV; y_n < 0} \alpha_n y_n K(\boldsymbol{x}, \boldsymbol{x}_n) + b, \tag{18}$$

$\boldsymbol{\lambda} = [\lambda_1\ \lambda_2\ \cdots\ \lambda_{np}]^T$ are parameters to be determined, and np is the number of support vectors in the minority class.

2. Determination of $\boldsymbol{\lambda}$

In ImSVM, the principle of determining $\boldsymbol{\lambda}$ is to make the decision function (18) achieve a maximum GMean value on a training sample set $T = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\}$.

The GMean measure is defined as the geometric mean of the accuracy rates of positive and negative samples [11], [18].

$$GMean = \sqrt{Acc_+ \times Acc_-}, \tag{19}$$

where $Acc_+ = T_P/(T_P + F_N)$ is the accuracy rate of positive samples, while $ACC_- = T_N/(T_N + F_P)$ is the accuracy rate of negative samples.

In the expression of GMean, the denominator of Acc_+ is the number of actual positive samples in T_t, while the denominator of Acc_- is the number of actual negative samples in T. Both these denominators are fixed values. Therefore, maximizing GMean is equivalent to maximizing $T_P \times T_N$, while T_P and T_P can be formalized as the first and second items of the following

https://doi.org/10.4218/etrij.17.0116.0879

equation respectively, thus obtaining the objective function for searching the optimal $\boldsymbol{\lambda}$ value.

$$\max_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda}) = \sum_{(\boldsymbol{x}_i, y_i)\in T; y_i>0} I(y_i g(\boldsymbol{x}_i, \boldsymbol{\lambda})) \times \sum_{(\boldsymbol{x}_i, y_i)\in T; y_i<0} I(y_i g(\boldsymbol{x}_i, \boldsymbol{\lambda})), \quad (20)$$

where $I(u)$ is an indicator function defined as

$$I(u) = \begin{cases} 1, & u \geq 0 \\ 0, & u < 0. \end{cases} \quad (21)$$

To be concrete, the first term in (20) calculates the number of correctly classified positive samples (T_P), while the second one calculates the number of correctly classified negative samples (T_N).

Equation (20) is an unconstrained optimization problem: the number of variables just equals to the number of positive support vectors, that is, *np*. The value of *np* is usually small. This optimization problem can be solved by the gradient descent method or Newton method [23].

After obtaining the optimal solution, $\boldsymbol{\lambda}^*$, it can be substituted into $g(\boldsymbol{x}, \boldsymbol{\lambda})$, thus obtaining a new decision function, $f(\boldsymbol{x}, \boldsymbol{\lambda}^*) = \text{sgn}(g(\boldsymbol{x}, \boldsymbol{\lambda}^*))$. The detection model represented by this new decision function is just the anomaly detection model produced by ImSVM.

3. Proposed ImSVM-Based Anomaly Detection Algorithm

The first key step in ImSVM is to solve the original classification hyperplane by using *C*-SVM-based anomaly detection algorithm (Algorithm 1). The second key step is to adjust the hyperplane, which is finally converted into an unconstrained optimization problem to search the optimal weight vector $\boldsymbol{\lambda}^*$.

The ImSVM algorithm is detailed in the following steps:

Algorithm 2 Anomaly detection algorithm based on ImSVM

Input: the training dataset $T = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l)\}$, where $\boldsymbol{x}_i \in R^n, y_i \in \mathcal{Y} = \{1, -1\}$, $i, i = 1, 2, \ldots, l$; the newly collected sample, $\boldsymbol{x}_{\text{new}}$.

Output: the optimal weight vector $\boldsymbol{\lambda}^*$, the solved decision function $f(\boldsymbol{x}, \boldsymbol{\lambda}^*)$, and the label y_{new} related to $\boldsymbol{x}_{\text{new}}$.

Step 1: Learn a detection model on T by *C*-SVM, and obtain a decision function and the original classification hyperplane represented by $(\boldsymbol{w}, b)$; obtain the set of support vectors *SV*.

Step 2: Solve the unconstrained optimization problem (20) on the training sample set; take the $\boldsymbol{\lambda}$ value when $J(\boldsymbol{\lambda})$ is maximum as the optimal solution $\boldsymbol{\lambda}^*$.

Step 3: Substitute $\boldsymbol{\lambda}^*$ into (18), and obtain a new decision function $f(\boldsymbol{x}, \boldsymbol{\lambda}^*)$, and a new classification hyperplane represented by $(\boldsymbol{w}, b, \boldsymbol{\lambda}^*)$.

Step 4: Compute $y_{\text{new}} = f(\boldsymbol{x}_{\text{new}}, \boldsymbol{\lambda}^*)$ if the obtained label $y_{\text{new}} = +1$, $\boldsymbol{x}_{\text{new}}$, is classified as an anomaly. Otherwise, $(y_{\text{new}} = -1)$, $\boldsymbol{x}_{\text{new}}$, is classified as a normal state.

V. Experiments and Analyses

1. Datasets Adopted in Experiments

A. Cloud Datasets

This article collects samples from an institute-wide Cloud platform, as illustrated in Fig. 4. The Cloud platform consists of 60 physical computing nodes, which are grouped into several clusters. A total of 0–4 VMs are deployed on each node. This number is dynamically changed according to the deployment assigned by the management server. Each sample contains 53 performance metrics, which indicate the heath state of a virtual machine (VM). These 53 performance metrics fall into the following categories: computation, storage, disk I/O, process, and network.

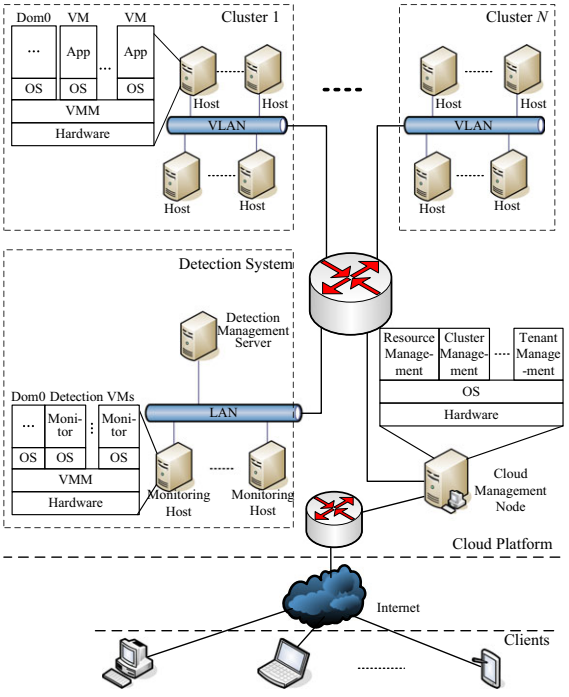

Fig. 4. Topological structure of an institute-wide Cloud platform.

The detection system learns a detection model from the training sample set and classifies a new sample as normal or abnormal. A Cloud dataset contains both normal and abnormal samples (anomalies). The anomalies may be abnormal consumption of computation resources, abnormal consumption of memory resources, abnormal disk I/O operation, or abnormal network access. The anomalies are confirmed by human operators.

B. KDD Datasets

The benchmark dataset, KDD dataset [24], is issued by the 5th Knowledge Discovery and Data Mining (KDD) Conference. It is derived from a dataset released by MIT Lincoln Laboratory [25]. This dataset simulates various intrusions (including DOS, Probe, U2R, and R2L), which are collected from an actual military network. It contains both a training sample set and a test one. The former includes 4,898,431 connection records (network traffic lasting for 7 weeks). The latter includes 2,984,154 connection records (network traffic lasting for 2 weeks). Each sample is associated with a label indicating either a normal connection or an attack.

2. Experiments

This article compares ImSVM with four other techniques for imbalanced datasets: *C*-SVM [22], ASMOTE [9] plus SVM, z-SVM [11], and cost-sensitive SVM [13].

Parameter setting:

1) *C*-SVM: $C \in [2^{-8}, 2^{+8}]$; Gaussian Radial Basis kernel function, $K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\gamma \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2)$, is adopted; the optimal parameters, C and γ, are solved by a 2D grid search method. Note that, the same parameter setting is adopted in the following SVMs.

2) ASMOTE plus SVM: the most important parameter, the percentage of over-sampling (or over-sampling rate), is set as 100% to balance the dataset.

3) z-SVM: the initial value of z is $z_0 = 1$; the optimal solution, z^*, is solved by a Golden section search algorithm.

4) Cost-sensitive SVM: the misclassification cost ratio r is defined as the ratio of the cost of false negative to that of false positive; the initial value is $r_0 = 1$; the optimal ratio r^* is solved by the method presented in [13].

5) ImSVM: the initial value of $\boldsymbol{\lambda}$ is $\boldsymbol{\lambda}_0 = (1, 1, \ldots,)^T$, the optimal solution $\boldsymbol{\lambda}^*$ is solved by a gradient descent method.

For the Cloud dataset, this article first constructs a training dataset that contains 5,000 samples. Abnormal samples account for only 1% (50 samples are abnormal ones). The involved five techniques train their respective detection models on this training dataset. Then, this article constructs a testing dataset that includes 2,000 samples. These five detection models are evaluated on the testing dataset. Table 1 lists the detection results. Table 2 lists the evaluation measures that are calculated by (12)–(15).

The following four conclusions are derived from the above experimental results:

1) *C*-SVM performs the worst. Specifically, the number of false negatives (F_N) in *C*-SVM is the highest, which produces low sensitivity. The underlying reason is that the learned original classification hyperplane is too close to and skewed toward the positive samples, which makes it is more possible to misclassify an anomaly as a normal state (false negative). The constructed Cloud dataset is a highly imbalanced dataset (abnormal samples account for only 1%). Therefore, the defect of *C*-SVM is evidently exposed.

2) z-SVM performs better than ASMOTE plus SVM. The main reason is that SMOTE may generate noisy artificial samples, thus causing a high rate of false alarms (false positives). As shown in Table 1, ASMOTE plus SVM causes three more false positives compared with z-SVM. Another reason is that the performance of ASMOTE is highly dependent on the proper determination of the percentage of over-sampling. If this parameter is set as a high value, too many noisy artificial samples are introduced. By contrast, a low value does not substantially improve the imbalance between samples of different classes. No defined guideline is reported to solve an optimal value.

3) Cost-sensitive SVM does not achieve the desired performance on the Cloud dataset. The underlying reason is that the performance of cost-sensitive SVM is highly dependent on the precise misclassification cost for each class of samples. Unfortunately, a defined guideline is lacking. Usually, empirical knowledge and an exhaustive search are needed to solve for the optimal costs.

4) ImSVM outperforms the other four algorithms in terms of all the four evaluation measures. Specifically, by

Table 1. Detection results on cloud datasets.

Algorithm	T_P	F_P	F_N	T_N
C-SVM	927	61	78	934
ASMOTE + SVM	938	57	67	938
z-SVM	946	54	59	941
Cost-sensitive SVM	954	51	51	944
ImSVM	961	47	44	948

https://doi.org/10.4218/etrij.17.0116.0879

Table 2. Evaluation measures on cloud datasets.

Algorithm	Sensitivity	Specificity	Precision	F_1-Measure
C-SVM	0.922	0.939	0.938	0.930
ASMOTE + SVM	0.933	0.943	0.943	0.938
z-SVM	0.941	0.946	0.946	0.944
Cost-sensitive SVM	0.949	0.949	0.949	0.949
ImSVM	**0.956**	**0.953**	**0.953**	**0.955**

rectifying the skewness of the original classification hyperplane toward the positive samples, the number of false negatives (F_N) in ImSVM is evidently reduced, thus enhancing the sensitivity. In addition, ImSVM does not simply translate the classification hyperplane toward the negative samples. In fact, it moves the classification hyperplane to a more proper position to maximize the GMean value on the training sample set. Therefore, the number of false positives (F_P) is also reduced at the same time, as shown in Table 1.

In addition, ImSVM performs well when the training sample set is highly imbalanced. In order to test the performance of these five techniques under datasets with different degrees of imbalance, a series of experiments are conducted where the percentage of positive samples in the training sample set is controlled as 5%, 10%, and 20%. A variation plot of a chosen evaluation measure (F_1-Measure) with the percentage of positive samples is given in Fig. 5.

The variation plot shows that the advantage of ImSVM compared with the other four algorithms is gradually weakened along with a decrease in the imbalance between samples of different classes. When the percentage of positive samples in the training sample set increases from 1% to 20%, the performances of all five algorithms increase, as shown in Fig. 5. This is because when the percentage of positive samples increases, the dominance of the penalty (in the objective function of SVM) of the negative samples decreases. When the positive samples account for 20%, no significant performance difference exists among these five algorithms. The underlying reason is that these five algorithms are all based on SVMs. When the percentage of positive samples is increased to 20%, SVM starts to perform reasonably well on the dataset. Therefore, the improvement of either ImSVM, ASMOTE, z-SVM, or cost-sensitive SVM, plays a minor role compared with standard *C*-SVM.

For the KDD dataset, this article constructs a training dataset containing 20,000 samples. Similarly, abnormal samples account for only 1%. Then, this article constructs a testing dataset that includes 5,000 samples. Table 3 lists the detection results. Table 4 lists the evaluation measures.

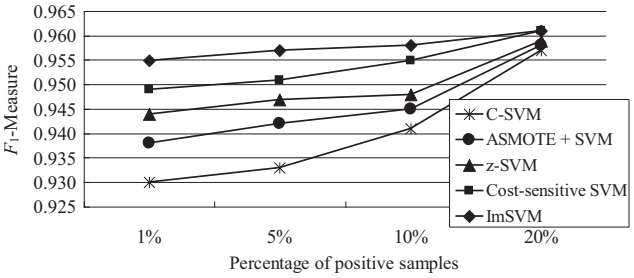

Fig. 5. Variation plot of F_1-Measure with the percentage of positive samples in the training sample set (Cloud datasets).

Table 3. Detection results on KDD cup datasets.

Algorithm	T_P	F_P	F_N	T_N
C-SVM	2,299	168	203	2,330
ASMOTE + SVM	2,325	160	177	2,338
z-SVM	2,348	153	154	2,345
Cost-sensitive SVM	2,365	149	137	2,349
ImSVM	2,383	141	119	2,357

Table 4. Evaluation measures on KDD Cup datasets.

Algorithm	Sensitivity	Specificity	Precision	F_1-Measure
C-SVM	0.919	0.933	0.932	0.925
ASMOTE + SVM	0.929	0.936	0.936	0.932
z-SVM	0.938	0.939	0.939	0.939
Cost-sensitive SVM	0.945	0.940	0.941	0.943
ImSVM	**0.952**	**0.944**	**0.944**	**0.948**

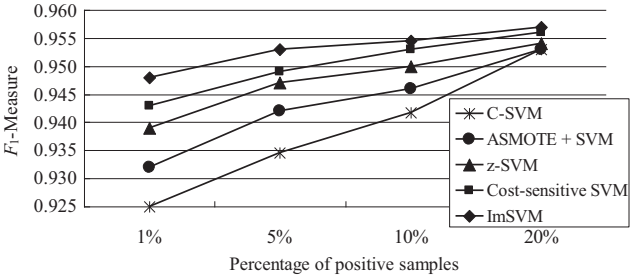

Fig. 6. Variation plot of F_1-Measure with the percentage of positive samples in the training sample set (KDD Cup datasets).

Similar conclusions can be reached from the experimental results on the KDD dataset: ImSVM performs the best among the involved five anomaly detection techniques; in particular, ImSVM evidently reduces the F_N value and therefore enhances sensitivity.

For the KDD dataset, the variation plot of the F_1-Measure with the percentage of positive samples is given in Fig. 6. It is also shown that when the percentage of positive samples in the training sample set increases from 1% to 20%, the performances of all five algorithms increase, but the advantage of ImSVM compared with the other four algorithms is gradually weakened.

VI. Conclusion

Aiming at solving imbalanced training datasets in the anomaly detection domain, this article proposes a new imbalanced SVM termed ImSVM. ImSVM adjusts the learned classification hyperplane by assigning a different weight for each positive support vector in the decision function. The weight vector is solved until a maximum GMean measure value is achieved. ImSVM decreases the possibility of misclassifying some positive samples as negative ones, thus reducing the false-negative rate and enhancing sensitivity. This article conducts a series of experiments on both Cloud and KDD datasets. The results verify that ImSVM outperforms four other SVM-based techniques when the training sample set is highly imbalanced.

This article is confined to binary classification. ImSVM is expected to be extended to multiclass anomaly detection. The comparison between ImSVM in a multiclass situation with other popular multiclass SVM-based anomaly detection algorithms is expected in the future.

Acknowledgments

The authors are grateful to the editor and anonymous reviewers for their valuable comments on this article. The work in this article is supported by the National Natural Science Foundation of China (Grant No. 61272399 and No. 61572090), the Chongqing Research Program of Basic Research and Frontier Technology (Grant No. cstc2015jcyjBX0014 and cstc2016jcyjA0304), and the Scientific and Technological Research Program of Chongqing Municipal Education Commission (Grant No. KJ1500538 and KJ1600521).

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," *ACM Comput. Surv.*, vol. 41, no. 3, July 2009, pp. 15:1–15:58.

[2] H. Lee et al., "Anomaly Intrusion Detection Based on Hyper-Ellipsoid in the Kernel Feature Space," *KSII Trans. Internet Inform. Syst.*, vol. 9, no. 3, 2015, pp. 1173–1192.

[3] J.C. Liu et al., "Anomaly Detection Using LibSVM Training Tools," *Int. J. Secur. Appl.*, vol. 2, no. 4, 2008, pp. 89–98.

[4] M. Hejazi and Y.P. Singh, "One-Class Support Vector Machines Approach to Anomaly Detection," *Appl. Artif. Intell.*, vol. 27, no. 5, 2013, pp. 351–366.

[5] D. Li, S.L. Liu, and H.L. Zhang, "A Method of Anomaly Detection and Fault Diagnosis with Online Adaptive Learning Under Small Training Samples," *Pattern Recogn.*, vol. 64, 2017, pp. 374–385.

[6] S. Fu, J.G. Liu, and H. Pannu, "A Hybrid Anomaly Detection Framework in Cloud Computing Using One-Class and Two-Class Support Vector Machines," *Proc. Int. Conf. Adv. Data Mining Applicat.*, Nanjing, China, Dec. 15–18, 2012, pp. 726–738.

[7] S.K. Tezel and L.J. Latecki, "Improving SVM Classification on Imbalanced Time Series Data Sets With Ghost Points," *Knowl. Inf. Syst.*, vol. 28, no. 1, Jan. 2011, pp. 1–23.

[8] N.V. Chawla et al., "SMOTE: Synthetic Minority Over-Sampling Technique," *J. Artif. Intell. Res.*, vol. 16, June 2002, pp. 321–357.

[9] Z.M. Yang, L.Y. Qiao, and X.Y. Peng, "Research on Data Mining Method for Imbalanced Dataset Based on Improved SMOTE," *Acta Electronica Sinica*, vol. 36, no. s2, Dec. 2007, pp. 22–26.

[10] C.L. Castro, M.A. Carvalho, and A.P. Braga, "An Improved Algorithm for SVMs Classification of Imbalanced Data Sets," *Proc. Int. Conf. Eng. Appicat. Neural Netw.*, London, UK, Aug. 7–29, 2009, pp. 108–118.

[11] T. Imam, K.M. Ting, and J. Kamruzzaman, "z-SVM: An SVM for Improved Classification of Imbalanced Data," *Proc. Australian Joint Conf. Artif. Intell.*, Hobart, Australia, Dec. 4–8, 2006, pp. 264–273.

[12] Z.M. Yang and X.Y. Peng, "μSVM - A New Method for Solving the Problem of Imbalanced Dataset Classification," *Chin. J. Sci. Instrum.*, vol. 29, no. S2, Aug. 2008, pp. 117–122.

[13] N. Thai-Nghe, Z. Gantner, and L. Schmidt-Thieme, "Cost-Sensitive Learning Methods for Imbalanced Data," *Proc. Int. Joint Conf. neural Netw.*, Barocelona, Spain, July 18–23, 2010, pp. 1–8.

[14] Y. Zhang et al., "Imbalanced Data Classification Based on Scaling Kernel-Based Support Vector Machine," *Neural Comput. Appl.*, vol. 25, no. 3–4, Apr. 2014, pp. 927–935.

[15] A. Maratea, A. Petrosino, and M. Manzo, "Adjusted F-Measure and Kernel Scaling for Imbalanced Data Learning," *Inform. Sci.*, vol. 257, Feb. 2014, pp. 331–341.
[16] M. Claesen et al., "EnsembleSVM: A Library for Ensemble Learning Using Support Vector Machines," *J. Mach. Learn. Res.*, vol. 15, Jan. 2014, pp. 141–145.
[17] R. Akbani, S. Kwek, and N. Japkowicz, "Applying Support Vector Machines to Imbalanced Datasets," *Proc. Eur. Conf. Mach. Learning*, Pisa, Italy, Sept. 20–24, 2004, pp. 39–50.
[18] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection," *Proc. Int. Conf. Mach. Learning*, 1997, pp. 179–186.
[19] C. Cortes and V.N. Vapnik, "Support-Vector Networks," *Mach. Learn.*, vol. 20, no. 3, 1995, pp. 273–279.
[20] A. Sun, E.-P. Lim, and Y. Liu, "On Strategies for Imbalanced Text Classification Using SVM: A Comparative Study," *Decis. Support Syst.*, vol. 48, no. 1, 2009, pp. 191–201.
[21] P. Branco, L. Torgo, and R.P. Ribeiro, "A Survey of Predictive Modeling on Imbalanced Domains," *ACM Comput. Surv.*, vol. 49, no. 2, 2016, pp. 31:1–31:50.
[22] N. Cristianini and J. Shawe-Taylor, "*An Introduction to Support Vector Machines and Other Kernel-Based Learning Methode*," New York, USA: Cambridge University Press, 2000.
[23] J. Nocedal and S.J. Wright, "*Numerical Optimization*", 2nd edn. New York, Springer, 2006.
[24] S. Hettich and S.D. Bay, "The UCI KDD Archive," 1999, Accessed 2016. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[25] R.P. Lippmann et al., "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation," *Proc. DARPA Inform. Survivability Conf. Expo.*, Hilton Head, SC, USA, 25–27, Jan. 2000, pp. 12–26.

GuiPing Wang received his BS degree in Automation, MS degree in Pattern Recognition and Intelligent System, and PhD degree in Computer Science at Chongqing University, China, in 2000, 2003, and 2015, respectively. Currently, he is an associate professor in the College of Information Science and Engineering, Chongqing Jiaotong University. His research interests include machine learning, dependability analysis and design of distributed systems, and cloud computing.

JianXi Yang received his PhD degree in Computer Science at Chongqing Jiaotong University, China, in 2011. Currently, he is a professor in the College of Information Science and Engineering, Chongqing Jiaotong University, China. His research interests include state monitoring, fault diagnosis, big data processing, and cloud computing.

Ren Li received his BS degree in Software Engineering, MS degree in Software Engineering, and PhD degree in Computer Science at Chongqing University, P. R. China, in 2007, 2009, and 2013, respectively. Currently, he is an associate professor in the College of Information Science and Engineering, Chongqing Jiaotong University, China. His research interests include cloud computing, big data processing, and semantic web techniques.