Efficient Training Support Vector Clustering with Appropriate Boundary Information

YUAN PING1, BIN HAO2, HUINA LI1, YUPING LAI3, CHUN GUO4, HUI MA1, BAOCANG WANG1,5, and XIALI HEI2, (Senior Member, IEEE)

1School of Information Engineering, Xuchang University, Xuchang 461000, China
2School of Computing and Informatics, University of Louisiana at Lafayette, LA 70503, USA
3College of Computer Science and Technology, North China University of Technology, Beijing 100144, China
4College of Computer Science and Technology, Guizhou University, Guiyang 550025, China
5The Sate Key Laboratory of Integrated Service Networks, Xidian University, Xi’an 710071, China

Corresponding author: Bin Hao (e-mail: bin.hao@louisiana.edu).

This work is supported by the National Key R&D Program of China under Grant 2017YFB0802000, the National Natural Science Foundation of China under Grant U1736111, the Plan For Scientific Innovation Talent of Hen’an Province under Grant 18410051002, the Program for Science & Technology Innovation Talents in Universities of He’nan Province under Grant 18HASTT022, Key Technologies R&D Program of He’nan Province under Grant 182102210123, the Foundation of He’nan Educational Committee under Grants 16A520025 and 18A520047, the Foundation for University Key Teacher of He’nan Province under Grant 2016GGJS-141, the Open Project Foundation of Information Technology Research Base of Civil Aviation Administration of China under Grant CAAC-ITRB-201702, and Innovation Scientists and Technicians Troop Construction Projects of He’nan Province.

ABSTRACT Due to the remarkable capability in handling arbitrary cluster shapes, support vector clustering (SVC) benefits data analysis in terms of data description. However, large-scale data such as network traffic frequently makes it suffer from highly intensive pricey computation and storage for solving the dual problem and storing the kernel matrix, respectively. Fortunately, support vectors which describe the clusters, in a sense, are expected in the boundaries. To tackle this issue, we propose an efficient training SVC with appropriate boundary information (ETSCV), which features excellent flexibility and scalability. In ETSVC, we first give a shrinkable boundary selection (SBS) method which collects appropriate boundaries while reducing redundancy and noise. Based on the boundary information, a redefined dual problem is then designed without scarifying the principle of SVC. Finally, we design a reformative solver (RSolver) to reformulate the training phase, which estimates the support vector function by solving the dual problem. Since only a subset of boundaries is employed for model training, theoretical analysis suggests that ETSVC reaches efficiency improvement and consumes much less memory if sacrificing efficiency to reduce storage consumption. Towards grouping P2P flows and large-scale intrusion traffic, as well as other non-traffic data, experimental results confirm that ETSVC could be applied to resources constrained platform while achieving comparable accuracies with the state-of-the-art methods.

INDEX TERMS Support vector clustering, shrinkable boundary selection, dual coordinate descent, traffic analysis, intrusion detection

I. INTRODUCTION

In an unsupervised way, clustering finds natural groupings of data samples that benefits pattern analysis and description of real-world problems. Based on the principle of support vector data description (SVDD), up to now, support vector clustering (SVC) has attracted much attention because the collected support vectors (SVs) profile clusters with arbitrary shapes well [1], [2]. These cluster profiles are frequently suitable for describing either normal or abnormal pattern in traffic analysis. However, before forming these profiles, SVs with their specific coefficients should be collected through excellent model training. It usually requires a large number of valid training samples and a hard iterative analysis under some metrics. This training phase has always been considered as one of the major bottlenecks [2], [3] for its pricey computation and storage about the number of participated data samples. Intuitively, an efficient model training for clear profiles we expect should focus on much more valuable
training samples as well as a faster training method.

Consider a data set \( \mathcal{X} \) with \( N \) data samples \( \{x_1, x_2, \ldots, x_N\} \) where \( x_i \in \mathbb{R}^d (i \in [1, N]) \) in data space, the training phase is pricey because it, generally, has to solve a quadratic programming problem. The runtime usually takes exponential time about \( N \), i.e., from \( O(N^2) \) to \( O(N^3) \) which depends on the actual case [2]. [4]–[6]. Furthermore, the number of iterative analysis is usually uncertain yet great for the final coefficient vector \( \beta \) that exacerbates the practical time-cost. Meanwhile, a space complexity of \( O(N^2) \) is generally required by the kernel matrix of the dual problem solution. Apparently, reducing the number of inevitable and valuable data samples is critical for performance improvement on either runtime or space. However, few works in the literature focus on this. They prefer selecting a subset of data samples according to a random or fixed strategy. As a representative of the prior, [7] uses a sample rate \( \theta \) to control the randomly selected data samples (\( N_\theta = \theta N \)) in model training and generating voronoi cells for the labeling phase. In spite of running fast, it results in an unstable accuracy. For the latter, [8] adopts the outermost boundaries following SVDD to reformulate and solve the dual problem. Since the boundaries are selected fixedly with given parameters, its performance is stable. However, we find that it suffers from too much noise to describe the clusters’ profiles well.

With these considerations, in this paper, we propose an efficient training SVC (ETSVC) towards making traffic analysis plausible on a resource-constrained platform. To reduce consumption cost, ETSVC firstly tries to avoid unnecessary computations in feature space by employing data samples on the shrunk boundaries. Since they are candidate SVs, we hence reformulate the dual problem and relax the constraints to reduce its solver’s cost. Meanwhile, the requirement of imprecise coefficients \( \beta \) urges us to control the iteration on demand while keeping the usability of coefficients. For the further benefit of efficiency or storage, ETSVC supports utilizing either the pre-computed kernel matrix or not, respectively. The achieved balance is concerning the trade-off between computation and storage cost, which a platform affords to. The main contributions lie in:

- A shrinkable boundary selection (SBS) method that prefers data samples on the shrunk boundary instead of the outermost edge in [9]. A certain degree of the final outliers and noises are refused before entering the solver. Furthermore, with a preprocess of data partition, the achieved efficiency improvement makes SBS well handle large-scale traffic and supply reasonable boundaries.
- The redefined dual problem follows the principle of minimal enclosing hypersphere. Only the candidate SVs (i.e., the output of SBS) are allowed to construct the hypersphere and estimate the support vector function. Naturally, we can expect a bog amount of time savings since, computations in feature space, the size of candidate SVs is much smaller than that of the whole data set.
- The proposal of a reformative solver (RSolver) for further efficiency improvement in model training. It employs a dual coordinate descent method (DCD) to reformulate the procedure of solving the redefined dual problem. By altering the unsupervised problem into a supervised issue, it adopts a linear method whose iterations can be flexibly controlled while imprecise coefficients meet the requirement. Additionally, unnecessary storage consumptions can be effectively avoided to make an efficient traffic analysis by ETSVC on resource-constrained platform possible.

The remainder of this paper is organized as follows: In Section II, the classic phases of SVC are briefly described. Then we give our review of the related works in Section III. In Section IV, we first present the SBS method, redefine the equivalent dual problem on the basis of SVC whose solver is then reformulated. After integrating with a typical labeling phase, theoretical analysis and experimental results of ETSVC are conducted in Section V. Finally, conclusions are drawn in the last section, as well as the future works to be investigated.

II. PRELIMINARIES OF SVC

A. ESTIMATION OF A TRAINED SUPPORT FUNCTION

Data samples of \( \mathcal{X} \) can be mapped to a high-dimensional feature space from data space through a nonlinear function \( \Phi(\cdot) \). Then, SVC tries to find a sphere with the minimal radius which contains most of the mapped data samples. This sphere, when mapped back to the data space, can be partitioned into several components, each one encloses an isolated cluster of samples. In mathematical formulation, the spherical radius \( R \) subjects to:

\[
\begin{align*}
\min_{R, \alpha, \xi} & \quad R^2 + C \sum_i \xi_i \\
\text{s.t.} & \quad \|\Phi(x_i) - \alpha\|^2 \leq R^2 + \xi_i,
\end{align*}
\]

(1)

where \( \alpha \) is the center of the sphere, \( \xi_i \) is a slack variable, and \( C \) is a constant controlling the penalty of noise. Following [3], [10], the expected sphere is estimated by a support function which is defined as a positive scalar function \( f: \mathbb{R}^n \rightarrow \mathbb{R}^+ \). Since the support function is constructed by SVs, we estimate it by solving a dual problem in Eq.(2) where \( x_i \) corresponds to coefficient \( \beta_i (i = 1, \ldots, N) \) if its \( 0 < \beta_i < C \) is a SV.

\[
\begin{align*}
\max_{\beta_j} & \quad \sum_j K(x_j, x_j) - \sum_{i,j} \beta_i \beta_j K(x_i, x_j) \\
\text{s.t.} & \quad \sum_j \beta_j = 1, \quad 0 \leq \beta_j \leq C, \quad j = 1, \ldots, N.
\end{align*}
\]

(2)

By optimizing Eq.(2) with Gaussian kernel \( K(x_i, x_j) = e^{-\|x_i - x_j\|^2} \), the objective trained support function can be formulated by the squared radial distance of the image of \( x \) from the sphere center \( \alpha \) given by...
f(x) = 1 - 2 \sum_j \beta_j K(x_j, x) + \sum_{i,j} \beta_i \beta_j K(x_i, x_j).\hspace{1cm} (3)

\alpha = \sum_j \beta_j \Phi(x_j).\hspace{1cm} (4)

Theoretically, the radius \( R \) is usually defined by the square root of \( f(x_i) \) where \( x_i \) is one of SVs.

### B. CLUSTER ASSIGNMENTS

Since SVs locate on the border of clusters, a simple graphical connected-component method can be used for cluster labeling. For any two samples, \( x_i \) and \( x_j \), we check \( m \) segments on the line segment connecting them by traveling their images in the hypersphere. According to Eq.(3), \( x_i \) and \( x_j \) should be labeled the same cluster index while all the \( m \) segments are always lying in the hypersphere, i.e., \( f(x_{\hat{m}}) \leq R^2 \) for \( \hat{m} \in [1, m] \). Otherwise, they will be in two different clusters.

### III. RELATED WORKS

Despite well discovering arbitrary patterns, the utilization of SVC frequently suffers from pricey computation and memory. Towards making efficiency improvement never hurt accuracy, the training phase and labeling phases are considered, respectively.

In the training phase, the core is solving the dual problem (2) which consists of three typical resource consumption tasks, i.e., complex operations and huge iterations in optimization methods reducing the efficiency, and the precomputed kernel matrix requiring more than the affordable memory. To tackle these issues, besides using generic optimization algorithms such as gradient descent and sequential minimal optimization [2], researches pay attention to rewriting the dual problem by introducing the Jaynes maximum entropy [11], the position-based weight [12] and the relationship amongst SVs [8]. However, to the best of our knowledge, few of them in SVC consider restricting the iteration number in a reasonable range for practical use. Furthermore, building the final model with the full dataset suffers from huge consumption of \( O(N^2) \) for the kernel matrix when \( N \) is too large. Thus, the faster and reformulated SVC (FRSVC) [1] employs a linear method which calculates the required items of the kernel matrix on demand. It provides a balance between efficiency and memory cost, yet its training phase may be delayed. To avoid the full data set, the voronoi cell-based clustering (VCC) [7] randomly selects \( \theta N (\theta \in (0, 1]) \) samples while the fast and scalable SVC (FSSVC) in [8] prefers the boundaries collected by a border-edge pattern selection (BEPS) method [9]. Consequently, they have to face unstable results due to employing a nondeterministic subset and unavoidable noisy data samples, respectively. Others related to reducing the working set and divide-and-conquer strategy were surveyed in [2]. However, bottlenecks still easily appear. Moreover, the performance of the training phases is closely related to the choice of labeling strategy, light or heavy. Unfortunately, it does not receive much attention in the literature.

In the labeling phase, the preferred component for connectivity analysis and the sampling check strategy are critical for both efficiency and accuracy. In [3], the first work of SVC, all the data samples are components employed to construct a complete graph (CG) which adopt a linear sampling strategy. CG is classical and good at accuracy, but its efficiency is too low to analyze large traffic. Reducing the number of sample pairs thus becomes the first consideration, e.g., using a part of data samples. For instance, position regularized support vector clustering (PSVC) [12] uses SVs, reduced complete graph (RCG) [13] extracts SEVs and equilibrium-based SVC (E-SVC) [14], [15] finds the transition points (TS). However, along with the reduced number of sample pairs, a side-effect of additional iterations becomes in seeking SEVs or TS. Another strategy is to make each component consist of a group of data samples. For instance, convex decomposition based clustering (CDCL) [16] suggests constructing convex hulls (CHs) by SVs to substitute SEVs. For efficiency, CDCL also reduces the average sample rate by a nonlinear sampling strategy. Later, FSSVC [8] and FRSVC [1] make further improvements by reducing the average sample \( m \) close to 1. Besides, [17] introduces a cell growth strategy which starts at any data sphere, expands by absorbing new neighboring spheres and splits if its density is reduced to a certain degree. Later, cone based cluster labeling (CCL) [18] checks the connectivity of two SVs through a single distance calculation. But too strict constraints emphasized on the solver degrade its applicability. In fact, for these methods, the other pricey consumption is the adjacent matrix, which usually ranges from \( O(N^2_{SV}) \) to \( O(N^2) \).

### IV. THE PROPOSED ETsvc

#### A. SHRINKABLE BOUNDARY SELECTION

In geometrical, cluster boundary consists of edge and border. Generally, a cluster has independent edges while any two clusters with overlapping region share the same border. For the latter, we usually consider them as two components of a cluster unless they have different labels. Similar to SVDD, the edge is what we expected to describe cluster in unsupervised learning. Consequently, without the border, the edge should be the most informative samples which can do the most accurate description of the distribution structure. In perspective, it can be the superset of SVs. Towards collecting it, Li and Maguire [9] designed BEPS whose performance is confirmed by both clustering [8] and classification [9]. However, we find that BEPS suffers from too many noises along with the un-shrunk edges.

As depicted by Fig.1, differences between SVs-based boundary and the collected edge for fire_Gaussians [10] which has five clusters can be captured. Apparently, excluding pricey time consumption, clear boundaries described by SVs in Fig.1 (a) are what we expect. But from Fig.1 (b), BEPS can not get shrunken edges and reserve much more data samples locating in the outermost of clusters. Does all of
them benefit describing cluster boundaries or estimating the support vector function? By introducing the noise definition of [19], we find that most of the edges collected by BEPS are coincide exactly with noises marked by black circles in Fig.1 (c). That means a certain ratio of edges makes no sense for the following dual problem solver.

To avoid noises while obtaining shrunken edges, in this section, we design a shrinkable boundary selection (SBS) method illustrated in Algorithm 1. We measure the imbalanced degree $\ell_i$ of the $k$ nearest neighbors’ locations of $x_i$ in line 4. Here $\cdot$ means inner product and $g(x)$ returns 1 if $x \geq 0$; otherwise, it returns 0. Since the outermost data sample is not what we want to reserve, it is eliminated by setting $\ell_i \leq \gamma_u$, while noise features sparsity also should be removed by keeping $\ell_i \geq \gamma_u$. Fig.1 (d) gives the evidence.

**Algorithm 1** Shrinkable Boundary Selection

**Require:** Dataset $\mathcal{X}$, thresholds $\gamma_l, \gamma_u$, and an integer $k$

**Ensure:** Edges $\mathcal{X}_e$ with $M$ samples

1. for a given data sample $x_i$ in $\mathcal{X}$ do
2. find the $k$ nearest neighbors $x_j$ of $x_i$
3. generate $n_i = \sum_{j=1}^{k} v_{ji}$ where $v_{ji} = x_j - x_i$
4. calculate $\ell_i = 1 - \frac{1}{k} \sum_{j=1}^{k} g(n_i^T \cdot v_{ji})$
5. if $\ell_i \in [\gamma_l, \gamma_u]$ then
6. return $x_i$ as an edge point
7. end if
8. end for

**B. HYPERSPHERE CONSTRUCTION**

Generally, SBS gets a subset $\mathcal{X}_e = \{x_{e_1}, x_{e_2}, \ldots, x_{e_M}\} \subseteq \mathcal{X}$ which contains the most informative samples for hypersphere

FIGURE 1: Comparative analysis of SVs based boundaries, the selected boundaries and noises. On five Gaussians with 1000 data samples equally distributed in fire clusters, (a) CG obtains the cluster boundaries described by SVs before labeling with $q = 2.9744$ and $C = 0.015$, (b) BEPS collects edge data with $k = 30$ and $\gamma = 0.2$, (c) [19] finds noises in edge data with $k = 20$, and (d) SBS obtains edges with $k = 30$, $\gamma_l = 0.78$, $\gamma_u = 0.82$. 
construction. That is, we can expect that each \( x_{ei} \) \((i = 1, \ldots, M)\) is a candidate SV though it is not accurate.

1) Estimation of the coefficients for boundaries

In view of Eq.(3) and the definition of SVs, in theoretical, we have the equal distances from \( x_{ei} \) \((i = 1, \ldots, M)\) to the center of the hypersphere in feature space. However, in practical, the distances should be approximately equal. Thus, we have

\[
R^2 \approx f(x_{ei}) \approx f(x_{e2}) \approx \cdots \approx f(x_{em}).
\]

Since \( K(x_{ei}, x_{e2}) \) with Gaussian kernel and \( \sum_{j} \beta_j K(x_{ei}, x_{ej}) \)
for \( i \in [1, M] \) are respectively equal in Eq.(5), it is easy to check that Eq.(5) has the following expression:

\[
\begin{aligned}
\sum_j \beta_j[K(x_{e1}, x_{ej}) - K(x_{e2}, x_{ej})] & \leq \xi_1 \\
\sum_j \beta_j[K(x_{e2}, x_{ej}) - K(x_{e1}, x_{ej})] & \leq \xi_2 \\
& \ldots \\
\sum_j \beta_j[K(x_{ej}, x_{ei}) - K(x_{ej}, x_{em})] & \leq \xi_{M-1} \\
\end{aligned}
\]

where \( j \in [1, M] \) and \( \sum_j \beta_j = 1 \). Let \( \beta = [\beta_1, \beta_2, \cdots, \beta_M]^T, \xi = [\xi_1, \xi_2, \cdots, \xi_{M-1}]^T \) and \( Q = [Q_1, Q_2, \cdots, Q_{M-1}]^T \) where

\[
Q_t = [1 - K(x_{e1}, x_{et+1}), K(x_{e2}, x_{ei}) - K(x_{e2}, x_{et+1}), \cdots, K(x_{em}, x_{et+1})],
\]

and \( t \in [1, M-1] \), then Eq.(6) can be further written as \( Q\beta \leq \xi \) with constraints of \( \sum_j \beta_j = 1 \) and \( \beta_j \geq 0 \) \((j \in [1, M])\). Since \( Q \) is determined by the collected edges, the vector \( \beta \) thus can be found by solving a linear system of equations with inequality constraint. Using \( \beta \) and Eq.(3), the required hypersphere with radius \( R \) can be constructed. Then cluster labeling will be carried out successfully. Towards this objective, we convert it into a quadratic programming problem for finding an efficient solution.

Consider the linear system of equation,

\[
Q\beta = [Q_1, Q_2, \cdots, Q_{M-1}]^T \times \beta = [Q_1\beta, Q_2\beta, \cdots, Q_{M-1}\beta]^T \leq \xi,
\]

we get

\[
[(Q_1\beta)^2, (Q_2\beta)^2, \cdots, (Q_{M-1}\beta)^2]^T \leq \xi,
\]

where \( Q_t \) is either positive or negative and each element of the matrix \( Q_t\beta \) or \((Q_i\beta)^2\) is close to 0(\( \leq \xi \)). Naturally, the accumulation of the matrix elements is expected to be as follows:

\[
\sum_{i} (Q_i\beta)^2 = \begin{bmatrix} Q_1\beta \\ Q_2\beta \\ \vdots \\ Q_{M-1}\beta \end{bmatrix}^T \begin{bmatrix} Q_1\beta \\ Q_2\beta \\ \vdots \\ Q_{M-1}\beta \end{bmatrix} \leq \xi
\]

Therefore, the linear system of equation can be approximated by

\[
\begin{aligned}
\min_{\beta} \beta^T H \beta \\
\text{s.t.} \sum_j \beta_j = 1, \forall j \in [1, M], \beta_j \geq 0
\end{aligned}
\]

where \( H = Q^T Q \) is a Hessian matrix in \( R^{M \times M} \). Note that it is a standard convex quadratic program, its global optimal solution can be obtained easily by any learning algorithms (e.g., the interior-point-convex algorithm [20],) and the value of the object function can be guaranteed very close to 0 for \((Q_i\beta)^2 \geq 0\). Additionally, differing from problem (2), this program is independent of the penalty factor \( C \).

After solving problem (11), the obtained coefficient vector \( \beta \) is equivalent to which got by Eq.(2) with penalty \( C \) close to 1 because \( C = 1 \) will not introduce outliers [3]. Due to \( M \ll N \), we can expect ETSVC behaving with less time and memory for training.

2) Reformative Solver for Dual Problem

Due to simplicity and efficiency, DCD is employed in solving large-scale linear SVM [21], nonparallel SVMs [22] and it reaches an \( \epsilon \)-accurate solution in \( O(\log(1/\epsilon)) \) iterations. However, the problem (11) cannot be solved by DCD because of its nonlinear model. Therefore, we should reformulate it with a linear model.

Let \( Q \) is the kernel matrix with element \( Q_{ij} = H_{ij}(i, j \in [1, M]) \) and \( \beta = [\beta_1, \beta_2, \cdots, \beta_M]^T \). Since the penalty \( C \) is close to 1 for this problem, apparently, the dual problem of Eq.(11) is equivalent to

\[
\begin{aligned}
\min_{\beta} \beta^T H \beta \\
\text{s.t.} \sum_j \beta_j = 1, \ 0 \leq \beta_j \leq 1, \ \forall j \in [1, M].
\end{aligned}
\]

Let \( \tilde{H} \) is a variant of matrix \( H \), and its element \( \tilde{H}_{ij} = 2 \times H_{ij} = 2 \times K(x_i', x_j')(i, j \in [1, M]) \). Here, we suppose that \( x_i' \) and \( x_j' \) are fictional data samples in dataset \( \mathcal{X}' \) which make \( 
\]

\[
\begin{aligned}
\min_{\beta} \frac{1}{2} \beta^T \tilde{H} \beta \\
\text{s.t.} \sum_j \beta_j = 1, \ 0 \leq \beta_j \leq 1, \ \forall j \in [1, M].
\end{aligned}
\]

Eq.(13) is an obvious convex optimization problem, thus, it can always reach the globally optimal solution in region of constraint \( D_{\text{svc}} = \{\beta | \sum_j \beta_j = 1, 0 \leq \beta_j \leq 1, j = 1, \ldots, M\} \). Similar with Refs. [13], [23]–[25], we can also find an equivalent globally optimal solution if we relax \( D_{\text{svc}} \) by \( D_{\text{svc}}' = \{\beta | 0 \leq \beta_j \leq 1, j = 1, \ldots, M\} \), where the strict condition of \( \sum_j \beta_j = 1 \) is removed. So, the dual problem (2) can be reformulated by a similar form of linear SVM (see Ref. [21] for details), i.e.,

\[
\begin{aligned}
\min_{\beta} \frac{1}{2} \beta^T \tilde{H} \beta \\
\text{s.t.} \ 0 \leq \beta_j \leq 1, \ j = 1, \ldots, M,
\end{aligned}
\]

where the expected diagonal matrix part \( D \) in \( \tilde{H} \) is removed or considered as 0 (i.e., \( D_{ii} = 0 \)) for simplicity.

In spite of having similar form with L1-SVM, we cannot solve Eq.(14) by using DCD method. Because, essentially, it is an unsupervised model. Therefore, we firstly convert it into
a supervised issue by extending $\lambda_i^t$ to $\lambda_i^j$ whose instance-label pairs are $\{(x_1^t, y_1), (x_2^t, y_2), \cdots, (x_M^t, y_M)\}$. Here $y_i$ is fixed to $+1$ or $-1$ for $i = 1, \cdots, M$. Thus, we present the specific DCD method for it following the works of Ref. [21].

The optimization process of Eq.(14) starts from an initial point $\beta^0 \in R^M$ and generates a sequences of vectors $\{\beta^k\}_{k=1}^{\infty}$. We refer to the process from $\beta^k$ to $\beta^{k+1}$ as an outer iteration. In each outer iteration we have $M$ inner iterations, so that sequentially $\beta_1, \beta_2, \cdots, \beta_M$ are updated.

Each outer iteration thus generates vectors $\beta^{k,i} \in R^M$, $i = 1, 2, \cdots, M$, such that $\beta^{k,1} = \beta^k$, $\beta^{k,M+1} = \beta^{k+1}$, and $\beta^{k,i} = [\beta^{k+1}_1, \cdots, \beta^{k+1}_{i-1}, \beta^{k+1}_i, \cdots, \beta^{k+1}_M]^T$ for $i = 2, \cdots, M$.

To update $\beta^{k,i}$ to $\beta^{k+1,i}$, we fix the other variable and then solve the following one-variable sub-problem:

$$\min_d f(\beta^{k,i} + \theta e_i)$$

s.t. $0 \leq \beta^{k,i} + \theta \leq 1$.

(15)

where $e_i = [0, \ldots, 0, 1, 0, \ldots, 0]^T$. Then, the objective function of Eq.(15) is simple quadratic function of $\theta$:

$$f(\beta^{k,i} + \theta e_i) = \frac{1}{2} \tilde{H}_{ii} \theta^2 + \nabla f(\beta^{k,i}) \theta + \text{constant},$$

where $\nabla f$ is the $i$-th component of the gradient $\nabla f$. Apparently, Eq.(15) has an optimum at $\theta = 0$ if and only if $\nabla^T f(\beta^{k,i}) = 0$, where $\nabla^T f(\beta)$ means the projected gradient

$$\nabla^T f(\beta) = \begin{cases} \nabla_i f(\beta) & \text{if } 0 < \beta_i < 1 \\ \min(\nabla_i f(\beta),0) & \text{if } \beta_i = 0 \\ \max(-\nabla_i f(\beta),0) & \text{if } \beta_i = 1. \end{cases}$$

(17)

If $\nabla_i f(\beta^{k,i}) = 0$, we move to the index $i + 1$ without updating $\beta^{k,i}$. Otherwise, we must find the optimum solution of Eq.(15). If $\tilde{H}_{ii} > 0$, the solution is

$$\beta^{k,i} = \min \left( \max \left( \tilde{H}_{ii}, \frac{\beta^{k,i} \cdot \nabla_i f(\beta^{k,i})}{\nabla_i f(\beta^{k,i})} \right), 1 \right),$$

(18)

and then continue finding in the current index without moving to $i + 1$.

Up to now, we can easily find that calculating $\tilde{H}_{ii}$ and $\nabla_i f(\beta^{k,i})$ are critical for the optimal solution. First, according to the definition, $\tilde{H}_{ii}$ can be calculated by

$$\tilde{H}_{ii} = 2 \times K(x_i,x_i) + D_{ii} = 2.$$  

(19)

It is a constant in the iteration. Second, to evaluate $\nabla_i f(\beta^{k,i})$, we have

$$\nabla_i f(\beta) = (\tilde{H} \beta)_i = \sum_{j=1}^{M} \tilde{H}_{ij} \beta_j.$$  

(20)

Since SVC is well known as a variant of one-class SVM, thus we can, in feature space, define a decision function by its boundary, i.e., $y = w^T \Phi(x_i) + b$. Then, we get

$$w = \sum_{i=1}^{M} \beta_i y_i \Phi(x_i) = \sum_{i=1}^{M} \beta_i \Phi(x_i).$$  

(21)

So, by introducing the definition of $\tilde{H}$, Eq.(20) becomes

$$\nabla_i f(\beta) = y_i w^T \Phi(x_i) + D_{ii} \beta_i = w^T \Phi(x_i).$$  

(22)

To apply Eq.(22), $w$ should be maintained throughout the coordinate descent procedure. Thus, along with the update of $\beta_i$, we can maintain $w$ by

$$w \leftarrow w + (\beta_i - \hat{\beta}_i) y_i \Phi(x_i) = w + (\beta_i - \hat{\beta}_i) \Phi(x_i),$$  

(23)

where $\hat{\beta}_i$ is the temporary coefficient value obtained in the previous iteration. Unfortunately, we cannot use an exactly nonlinear function $\Phi(\cdot)$ to map data sample $x_i$ from the input space to the feature space. So we take Eq.(23) into Eq.(22), alternatively, to achieve an one-time update, i.e.,

$$\nabla_i f(\beta^{k,i}) \leftarrow [w + (\beta_i - \hat{\beta}_i) \Phi(x_i)]^T \Phi(x_i) = w^T \Phi(x_i) + (\beta_i - \hat{\beta}_i) \Phi(x_i) = \sum_j \hat{\beta}_j K(x_j,x_i) + (\beta_i - \hat{\beta}_i).$$  

(24)

Therefore, to update $\nabla_i f(\beta)$ for finding the optimal solution, the current iterative value strongly depends on the weighted summation of the $i$-th row of $\tilde{H}$ (i.e., $\tilde{H}_{ii}$) in the previous iteration and the updated $\beta_i$.

Briefly, to solve the dual problem (2), our algorithm uses Eq.(24) to compute $\nabla_i f(\beta^{k,i})$, checks the optimality of the sub-problem (15) by $\nabla^T f(\beta^{k,i}) \geq 0$, updates $\beta_i$ by Eq.(18).

The procedure of RSolver is detailed in Algorithm 2. The cost per iteration from $\beta^{k}$ to $\beta^{k+1}$ is $O(M)$ with $H$.

Algorithm 2 RSolver for the Dual Problem (14)

Require: Hessian matrix $H$ in Eq.(11)

Ensure: Coefficient vector $\beta$

1. Randomly initialize the coefficient vector $\beta$
2. While $\beta$ is not the optimal
3. for $i = 1, 2, \cdots, M$ do
4. $\hat{\beta}_i \leftarrow \beta_i$
5. $\hat{G} \leftarrow 2 \times \sum_{j=1}^{N} \hat{\beta}_j H_{ji}$
6. $G \leftarrow \hat{G} + (\beta_i - \hat{\beta}_i)$
7. $PG = \left\{ \begin{array}{ll} 0 & \text{if } 0 < \beta_i < 1 \\ \min(G,0) & \text{if } \beta_i = 0 \\ \max(G,0) & \text{if } \beta_i = 1. \end{array} \right.$
8. if $|PG| \neq 0$ then
9. $\beta_i \leftarrow \min(\max(\beta_i - \frac{1}{2} G,0),1)$
10. end if
11. end for
3) Removal of useless boundary samples
In reality, we cannot expect all of the selected boundary samples to be SVs. From the perspective of hypersphere construction, some of them are useless or less informative. Even though SBS cannot altogether avoid them, we can remove them as many as possible by measuring their coefficient values extracted by Algorithm 2. Following the work of [8], we remove those boundary samples with coefficient values lower than a predefined threshold \( \beta_s \). A small \( \beta_s \) allows more redundant data to draw cluster boundaries accurately, but sometimes go along with overfitting; whereas a greater \( \beta_s \) would generate a smoother profile at the risk of producing much more overlapped regions between clusters. A large number of experiments suggest that an appropriate threshold \( \beta_s \) for removing the less informative points should be \( 10^{-\gamma} M^2 \) more or less.

4) Estimation of the radius of the hypersphere
After removing the useless boundary samples, we get a reduced boundary sample set \( \Sigma_R = \{x_{r_1}, x_{r_2}, \ldots, x_{r_L}\} \subseteq \Sigma_e \) with \( L \leq M \). Generally, data samples in \( \Sigma_R \) are considered as SVs whose distances to the center of the hypersphere are approximately equal. However, the removal of useless boundary samples with small yet positive coefficients reduces \( \alpha \) that makes the radius of the hypersphere greater. Thus, we construct the final hypersphere whose radius \( R \) is the distance from any \( x_{r_l} (l \in [1, L]) \) to its center, i.e.,

\[
R = \text{any } f(x_{r_l})^{1/2}, \quad l = 1, 2, \ldots, L
\]

C. THE FRAMEWORK OF ETSVC
By integrating methods in Section IV-A and IV-B, we present a basic framework of ETSVC in Algorithm 3 following a similar form with the classic SVC.

**Algorithm 3 Description of ETSVC**

**Require:** Dataset \( \Sigma \), integer \( k \), thresholds \( \gamma_l, \gamma_u \) and kernel width \( q \)

**Ensure:** Clustering labels for all the data samples

1. \( \{\Sigma_e, M\} \leftarrow \text{SBS}(\Sigma, k, \gamma_l, \gamma_u) \)
2. \( \Sigma \leftarrow Q^\dagger Q \) following Eq. (7)
3. \( \beta \leftarrow \text{RSolver}(\Sigma) \)
4. \( \Sigma_R \leftarrow \text{removing } x_l \text{ with } \beta_l < 10^{-\gamma} M \gamma \text{ from } \Sigma_e \)
5. Re-estimating \( \Sigma \) following Eq. (25)
6. \( \mathcal{P} \leftarrow \text{finding cluster prototypes for } \Sigma_R \)
7. \( \mathcal{A} \leftarrow \text{sampling for connectivity analysis with } \mathcal{P} \)
8. Labels \( \leftarrow \text{finding connected components using } \mathcal{A} \)
9. for each \( x \in \Sigma \setminus \mathcal{A} \) do
10. \( \text{index} \leftarrow \text{find the nearest SV from } x \)
11. Labels[\( x \)] \( \leftarrow \text{Labels}[x_{\text{index}}] \)
12. end for
13. return Labels

In Algorithm 3, lines 1-5 constitute the training phase while lines 6-13 correspond to the labeling phase. Given parameters \( k \), \( \gamma_l \), and \( \gamma_u \), SBS(\( \cdot \)) in line 1 collects boundaries \( \Sigma_e \) with \( M \) samples. Following Eq. (7), line 2 constructs the essential Hessian matrix \( \Sigma \) which is the input of RSolver(\( \cdot \)) for evaluating coefficient vector \( \beta \). After removing unnecessary data samples, \( \Sigma_R \) is obtained in line 4. Line 5 thus re-estimates the radius \( R \) of the hypersphere which is frequently used in labeling strategies. Step into the labeling phase, we can choose any cluster prototypes for \( \Sigma_R \) and any sampling strategy to match it for connectivity analysis in lines 6-7.

Table 3 gives our considerations in this study, and much more discussions can be found in [2]. After that, line 8 decides labels of the chosen prototypes, which will be employed in lines 9-12 to label all the remaining data.

V. PERFORMANCE ANALYSIS

A. COMPLEXITY ANALYSIS

To measure the time complexity of ETSVC, let \( N \) be the number of samples in a data set, \( N_{SV} \) be the number of SVs, \( \ell \) be the average number of iterations for each data sample to locate its corresponding local minimum via steepest descent process [13], \( \ell_{etsvc} \) be the self-defined iteration number for ETSVC, \( N_e \) be the final number of CHs, \( M \) be the size of the selected cluster boundaries by FSSVC [8], \( L \) be the size of \( \Sigma_R \) obtained by ETSVC, and \( m \) be the average sample rate. Since the selected data samples and the way of problem construction frequently influence the labeling performance, more or less, we take the labeling phase into account even though efficient training is a major concern.

For the training phase, the whole procedure is described in Algorithm 2. The proposed SBS algorithm consumes \( O(N^2) \) in data space to get a reduced set of boundaries. Considering \( L \leq M \ll N \), we prefer a pre-computed \( \Sigma \) for simplicity. Thus, the training phase of ETSVC consumes \( O(N^2 + L^2) \). It is much lower than \( O(N^3) \) required by the conventional methods [2]. For the labeling phase, ETSVC constructs CHs and makes connectivity analysis between them. The prior decomposition uses a self-defined \( \ell_{etsvc} \) iterations from \( L \) data samples while the latter analysis is flexible to any sampling strategy based on CHs, e.g., strategy from FSSVC or FRSVC [1]. Thus, we denote it by a linear function \( f(\zeta) \) where \( \zeta \) can be either \( N_e \) or \( N_{SV} \).

**Table 1:** Time complexity analysis of the state-of-the-art methods.

<table>
<thead>
<tr>
<th>Index</th>
<th>Method</th>
<th>SVC training</th>
<th>Labeling</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FSSVC</td>
<td>( O(N^2) )</td>
<td>( O(N_{SV} + gN) )</td>
</tr>
<tr>
<td>2</td>
<td>PSVC</td>
<td>( O(N^3) )</td>
<td>( O(mN^2) )</td>
</tr>
<tr>
<td>3</td>
<td>CDCL</td>
<td>( O(N^3) )</td>
<td>( O(\ell N_{SV} + 2m N_e) )</td>
</tr>
<tr>
<td>4</td>
<td>VCC</td>
<td>( O(N^2) )</td>
<td>( O(\ell N_{SV} + m N_e) ) Mode I: ( O(\ell N_{SV} + m N_e) )</td>
</tr>
<tr>
<td>5</td>
<td>FSSVC</td>
<td>( O(N^2 + M^2) )</td>
<td>( O(\ell N_{SV} + 2m N_e) ) or ( O(N_{SV}^2) )</td>
</tr>
<tr>
<td>6</td>
<td>FRSVC</td>
<td>( O(dN^2) )</td>
<td>( O(\ell N_{SV} + m N_e) )</td>
</tr>
<tr>
<td>7</td>
<td>ETSVC</td>
<td>( O(N^2 + L^2) )</td>
<td>( O(\ell_{etsvc} L + f(\zeta)) )</td>
</tr>
</tbody>
</table>

Note: \( d \in [1 \leq d \leq d], m \in [1, 2] \)
\( N_{tr} = \theta N, \theta \) is the sample rate in \([0, 1]\).

Additionally, we compare ETSVC with the state-of-the-art methods, i.e., FRSVC [1], VCC [7], FSSVC [8], fast support
vector clustering (FSVC) [10], PSVC [12], and CDCL [16]. In Table 1, \( \gamma \) ranges from 1/N to 1, \( N_v \) is the number of data which is uniformly sampled with a predefined sample rate \( \theta \), and \( N_s \) is the number of small balls extracted from either \( N_v \), data samples for VCC or the whole data set for the others. Even though two optional modes of labeling phase are presented for VCC, in this study, Mode I with fast phase to label the remaining data is preferred for its relative stability. Notice that even though FR-SVC has similar solver with ET-SVC, their problems and data forms are different. The data with large \( N \) makes FR-SVC adopt the strategy of calculation on demand. It generally takes \( O(dN^2) \) where \( d \) is the data dimension. So its time complexity in the training phase should be \( O(dN^2) \) with \( d \in [1 \leq d \leq d] \).

**B. DATASETS AND EVALUATION METRICS**

Although the proposed ETSVC follows the fundamental principle of SVC, it adopts a distinct way to construct and train the model with a noise eliminated subset of boundaries. Therefore, to verify the effectiveness of ETSVC, we conduct two series of experiments. The first one is to check whether it keeps the representative data profile for traffic analysis. In general, a well-kept data profile makes a high accuracy reached easier. On the one hand, we want to ease the burden of iterative analysis in RSolver; on the other hand, different labeling strategies will not scarify the expected accuracy a lot. The second series of experiments is to find out the performance of ETSVC on challengeable traffic analysis and non-traffic domain issues with different data types, e.g., integer and real value. In practical, different data types correspond to distinguished methods of data pre-processing, which are usually efficiency related.

Towards the objectives above, the first employed traffic is the UNIBS Anonymized 2009 Internet Traces (UNIBS-AIT) [26] supplied by TNG@UniBS Lab. It consists of 9209 flows in 4 imbalance distributed categories, i.e., WEB (HTTP and HTTPS), MAIL (POP2, IMAP as well as their encrypted flows), BitTorrent, and eMule. Following the method of [27], UNIBS-AIT is featured by the first package size from the client to the server, the first package size from the server to the client, the second package size from the server to the client, and the port number. All the features are integer values. The second is the KDD’99 dataset [28]. In spite of several limitations reported by [29], KDD’99 remains a standard and important dataset which presents a classic challenge and is widely used in the design of network intrusion detection [30–33]. KDD’99 includes three independent sets: the whole KDD training data, 10% KDD training data, and KDD correct data. Each record of KDD’99 represents a network connection which can be categorized as normal or one of four attack classes, namely remote-to-local (R2L), denial-of-service (DoS), user-to-root (U2R), and Probe (Prb). Following the work of [32], we keep 494,021 records for cluster analysis. Meanwhile, based on its 41 features for each record, namely 7 nominal features and 34 continuous features, we map the nominal features such as ‘protocol’, ‘TCP Status flag’ and ‘service type’ into binary numeric features that results in 52 features. Without enough prior knowledge of the actual number of clusters, we use k-means to extract 9 centroids that implicitly support multi-centroids in a class of records. Thus, all the records are represented by their normalized distances to the 9 centroids, respectively. That means the dimension of the final records is reduced to 9 in real value while remaining the specific relationship between each record and the 9 cluster centers. Size and distribution of UNIBS-AIT and KDD’99 are illustrated in Table 2.

**TABLE 2: Data descriptions of UNIBS-AIT and KDD’99**

<table>
<thead>
<tr>
<th>Class</th>
<th>Data size</th>
<th>Class</th>
<th>Data size</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEB</td>
<td>6,713</td>
<td>Normal</td>
<td>97,278</td>
</tr>
<tr>
<td>MAIL</td>
<td>653</td>
<td>DoS</td>
<td>391,458</td>
</tr>
<tr>
<td>BitTorrent</td>
<td>215</td>
<td>Prb</td>
<td>4,107</td>
</tr>
<tr>
<td>eMule</td>
<td>1,628</td>
<td>R2L</td>
<td>1,126</td>
</tr>
<tr>
<td></td>
<td>9,209</td>
<td>U2R</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>Total</td>
<td>494,021</td>
</tr>
</tbody>
</table>

To evaluate the accuracy, we adopt the adjusted rand index (ARI) [34] formulated by Eq.(26). It is a widely used similarity measure between two data partitions where both true labels and predicted cluster labels are given. In Eq.(26), \( N_{ij} \) is the number of data points with true label \( i \) but they are assigned by \( j \), \( N_i \) and \( N_j \) are the number of data points with label \( i \) and \( j \) respectively. Furthermore, derived from [30], we define a detection rate (DR) by Eq.(27) to show the detection performance on each type of flows.

\[
ARI = \frac{\sum_{i,j} (N_{ij})^2 - \left[ \sum_i (N_i^2) \sum_j (N_j^2) \right] / (N^2)}{\frac{1}{2} \left[ \sum_i (N_i^2) + \sum_j (N_j^2) \right] - \left[ \sum_i (N_i^2) \sum_j (N_j^2) \right] / (N^2)}
\]

\[
DR_i = 1 - \frac{\sum_{j \neq i} N_{ij}}{N_i}
\]

In this study, we implement ETSVC in MATLAB 2017a. As the previous settings, the flexibility and usability of the proposed ETSVC are the crucial concerns in this study. We do not try to find out the computational time savings brought by powerful CPU or parallelization, because this aspect of improvement on efficiency is a matter of course. Therefore, the testbed features with a computer running Windows 7-X64 on Intel Quad Core 2.0 GHz and 32GB RAM.

**C. EXPERIMENTAL RESULTS**

Towards achieving the full and fair analysis of ETSVC with its components, the conducted experiments consist of five parts. The first three are evaluations on UNIBS-AIT for SBS (Algorithm 1), RSolver (Algorithm 2) and ETSVC, respectively. Compared with the state-of-the-art methods, the fourth part uses KDD’99 to check the performance and applicability of ETSVC. Additionally, we introduce different types of data sets to evaluate ETSVC’s performance on non-traffic domains data in part five.
1) Performance of the SBS in ETSVC
As described in Section IV-A, the proposed SBS is derived from BEPS (employed by FSSVC [8]) to get shrunken edges for the following hypersphere construction in ETSVC. By fixing $k$ to 80, Fig. 2(a) shows the ratio of the extracted edges by SBS on UNIBS-AIT with dynamical $\gamma_1$ and $\gamma_u$. From Fig. 2(a), the greater of the gap between $\gamma_1$ and $\gamma_u$ is, the more edges are kept. The relationship is approximately linear. Undoubtedly, a nest edge set with noise eliminated benefits the edges are kept. The relationship is approximately linear. Un-

ginies
TABLE 3: Notations of ETSVC with different labeling strategies

<table>
<thead>
<tr>
<th>Notation</th>
<th>Connectivity analysis</th>
<th>Sampling method</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETSVC-1-1</td>
<td>Analysis between SVs</td>
<td>Linear in CG [3]</td>
</tr>
<tr>
<td>ETSVC-1-2</td>
<td>Analysis between SVs</td>
<td>Nonlinear in CDCL [16]</td>
</tr>
<tr>
<td>ETSVC-1-3</td>
<td>Analysis between SVs</td>
<td>Nonlinear in FSSVC [8]</td>
</tr>
<tr>
<td>ETSVC-2-1</td>
<td>Analysis between CHs</td>
<td>Linear in CG [3]</td>
</tr>
<tr>
<td>ETSVC-2-2</td>
<td>Analysis between CHs</td>
<td>Nonlinear in CDCL [16]</td>
</tr>
<tr>
<td>ETSVC-2-3</td>
<td>Analysis between CHs</td>
<td>Nonlinear in FSSVC [8]</td>
</tr>
<tr>
<td>ETSVC-2-4</td>
<td>Analysis between CHs</td>
<td>Nonlinear in R-CG [13]</td>
</tr>
<tr>
<td>ETSVC-2-5</td>
<td>Analysis between CHs</td>
<td>Sample once in FR SVC [1]</td>
</tr>
</tbody>
</table>

For efficiency, we expect RSolver in Algorithm 2 requiring as few iterations as possible due to its sale. However, in tradition, few iterations frequently suffer from the non-optimal results of the objective optimization problems which are related to the accuracy. To find a balance between accuracy and efficiency, we conduct experiments on UNIBS-AIT to verify the relationship between the number of iterations and the obtained ARI. Fig. 3 depicts the results. Amongst eight ETSVC’s variants using RSolver, six methods perform stably as the iteration number increases. Only ETSVC-1-2 and ETSVC-2-5 have some vibrations, e.g., the first iteration and tenth iteration, respectively. As shown in Fig. 3, no significant impact on accuracy brought by iteration number when it is greater than 10. Therefore, we set the maximal iteration number to 10 for RSolver in the following analysis.

![FIGURE 3: On UNIBS-AIT, the relationship between accuracy and the iteration number in the proposed RSolver.](image)

Further, we make comparisons between the proposed RSolver and the widely used QPSSVM solver [2], [10], [25], namely quadratic programming task for struct SVM learning. On UNIBS-AIT, ARIs achieved by the eight variants of ETSVC with RSolver and QPSSVM are given by Fig. 4. For each comparison, the better performance is highlighted by boldface. Eventually, seven out of eight ETSVCs with RSolver outperform that of with QPSSVM, especially for ETSVC-2-5, while RSolver and QPSSVM get the same results in ETSVC-2-2. Intuitively, in terms of accuracy, we conclude that RSolver performs better than QPSSVM in ETSVC. Notice that the efficiency discussions will be made in the next section.

2) Performance of the RSolver in ETSVC
RSolver is critical for our ETSVC, which is compatible with the existing labeling strategies. Before evaluating its performance, we first summarize the notations of ETSVC with different labeling strategies in Table 3. For the sake of simplicity, we group the employed components for connectivity analysis in literature into SVs and CHs. They are respectively denoted by “1” and “2”. Meanwhile, 1-5 represent five different sampling methods (between SVs or CHs) in sequence, i.e., linear in CG [3], nonlinear in CDCL [16], nonlinear in FSSVC [8], nonlinear in R-CG [13], and sample once in FR_svc [1]. For instance, ETSVC-2-3 means we use ETSVC with connectivity analysis between CHs and nonlinear sampling strategy presented in FSSVC [8].

3) Performance and adaptability analysis of ETSVC
To make objective evaluations, we conduct experiments on UNIBS-AIT for eight variants of ETSVC illustrated by Table 3 and the state-of-the-art SVC methods in Table 1. Results of accuracy and efficiency are separately presented in Fig. 5 and Table 5. Three points are important to be noted. First, due to the particular sampling strategy before model training, VCC can not fixedly achieve accuracy with unchanged parameters. Thence, we choose the best accuracy after far more than ten results by fixing the optimal parameters. Secondly, the time cost is an average value of ten times of the execution on UNIBS-AIT. Thirdly, even though UNIBS-AIT is a relatively large-scale data, the adopted platform can afford the pre-computed kernel matrix which is suggested by FR SVC on a server with sufficient memory. So we separately evaluate the time-consumptions for FR SVC with or without sufficient memory by using the pre-computed kernel matrix and cal-
FIGURE 2: Performance of SBS on UNIBS-AIT. (a) The ratio of the obtained $L$ edges to $N$ by the proposed SBS with $k = 80$ and $(\gamma_l, \gamma_u)$ ranging from 0.6 to 1. (b) Accuracies achieved by ETSVC with FSSVC’s labeling strategy with respect to $k$ while $\gamma_l$ and $\gamma_u$ are fixed to 0.85 and 0.95, respectively.

FIGURE 4: On UNIBS-AIT, ARIs achieved by ETSVCs with RSolver and QPSSVM in [10], respectively.

FIGURE 5: On UNIBS-AIT, the accuracies (ARI) achieved by the state-of-the-art methods and ETSVC with different labeling strategies.

calculating on demand. They are denoted by FRSVC(s) and FRSVC(c), respectively.

In Fig.5, all the compared methods are presented in the order of ARI from high to low, and the first three ranks of ARI measures are highlighted by boldface. Apparently, CDCL reaches the best accuracy 0.8917, which is better than 0.8863 of ETSVC-2-3, and ETSVC-2-3 outperforms the others significantly. As a supplement, Table 4 shows the detection performance of ETSVC-2-3 on four classes of flows in UNIBS-AIT. Besides ETSVC-2-3, six out of the remaining seven ETSVCs and FSSVC achieve the same accuracy (0.8815) which outperforms FRSVC, ETSVC-2-5, FSSVC, VCC, and PSVC. Compared with FSSVC which uses BEPS to select boundaries, most of the variants of ETSVCs (7 out of 8) perform comparably or even better in this case although they adopt different solvers. Furthermore, ETSVC has an excellent adaptability with various labeling strategies.

TABLE 4: Detection performance of ETSVC-2-3 on four classes of flows in UNIBS-AIT

<table>
<thead>
<tr>
<th>Class</th>
<th>WEB (%)</th>
<th>MAIL (%)</th>
<th>BitTorrent (%)</th>
<th>eMule (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDCL</td>
<td>3.917</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>ETSVC-2-1</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-2</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-3</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-4</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-5</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-6</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-7</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-8</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-9</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
<tr>
<td>ETSVC-2-10</td>
<td>0.916</td>
<td>0.851</td>
<td>0.035</td>
<td>0.013</td>
</tr>
</tbody>
</table>

Y. Ping et al.: Efficient Training Support Vector Clustering with Appropriate Boundary Information
In Table 5, time consumptions for each phase of the compared 15 methods are separately illustrated. We group them into two types on the basis of whether to use a pre-processing of edge selection or boundary selection. EdgeS and HyperSC represent edge selection and hypersphere construction in Section IV, respectively. For FSVC, PSVC, CDCL, VCC, and FR SVC, we combine the corresponding 2-3 columns and show one value for the training phase for each. Methods adopting QPSSVM solver in their training phases are noted. Apparently, some conclusions are revealed as follows:

- For the training phase, ETSVCs gain excellent performances in both stability and efficiency. Here, FR SVC(s) and VCC reach comparable efficiencies which outperform the others significantly. Actually, only 11% data samples are selected in the training of VCC with $\theta = 0.11$, and we have omitted the time-consumption for kernel matrix construction in FR SVC(s) which runs with a precomputed kernel matrix. Without sufficient memory, FR SVC(c) usually requires more time. Besides VCC and FR SVC(s), ETSVCs show significant advantages in comparison with FSSVC, FR SVC(c), CDCL, FSVC, and PSVC. For instance, the most time-consumption method is ETSVC-1-1 which only costs less than one-eleventh of FR SVC(c). All the ETSVCs perform comparably and require less time than FSSVC in EdgeS and HyperSC. Particularly for HyperSC, ETSVCs consume less than 5 seconds while FSSVC costs 14.6621 seconds. The fact suggests that RSolver runs more effectively than QPSSVM. Amongst ETSVCs, different time-costs for either EdgeS or HyperSC are due to different parameters setting.

- For the labeling phase, ETSVCs benefit the improvement of efficiency. Although there is no specifically designed labeling strategy for ETSVC, the adaptability of ETSVC makes the known strategies work well. Firstly, we consider methods doing connectivity analysis between SVs, i.e., PSVC, ETSVC-1-1, ETSVC-1-2, and ETSVC-1-3. The last three obviously cost less than PSVC. Secondly, amongst the remaining methods doing connectivity analysis between CHs (including ETSVCs, FSSVC, FR SVC, and CDCL), circle-like cells (FSVC) and Voronoi cells (VCC), FSSVC performs the best and ETSVCs take the second place. On the basis of Table 3, we find that ETSVC-2-2 and ETSVC-2-5 respectively perform better than CDCL and FR SVC, even though the corresponding labeling strategies are the same. In addition, ETSVC-2-2 (i =1,2, …, 5) outperforms ETSVC-1-2 (j=1,2,3). Therefore, for efficiency, using CHs or other representative prototypes for connectivity is strongly recommended.

- By combining the training phase and the labeling phase, column 5 presents the time cost in total. Fortunately, we find that ETSVC-2-5 wins the first rank even though its accuracy (see Fig.5) is not the best. Relatively, ETSVC-2-2 (i =1,2,3,4) not only finishes traffic analysis in a comparable time with ETSVC-2-5, but also outperforms absolutely most of the others in terms of accuracy. Especially, their time consumptions are about one-twenty seventh of CDCL that makes traffic analysis be practical. If a reduction strategy for iterations is utilized following the instructions of Fig.3, we can expect further improvements on efficiency.

### TABLE 5: Time comparisons among ETSVC and the state-of-the-art methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Training Phase (s.)</th>
<th>Labeling Phase (s.)</th>
<th>Total (s.)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EdgeS</td>
<td>HyperSC</td>
<td></td>
</tr>
<tr>
<td>FSVC</td>
<td>1066.16/7 (QPSSVM)</td>
<td>8.008†</td>
<td>1047.1755</td>
</tr>
<tr>
<td>PSVC</td>
<td>2210.51/68 (QPSSVM)</td>
<td>49.049†</td>
<td>2259.5667</td>
</tr>
<tr>
<td>CDCL</td>
<td>268.08/07 (QPSSVM)</td>
<td>11.2734</td>
<td>279.3541</td>
</tr>
<tr>
<td>VCC</td>
<td>0.3902† (QPSSVM)</td>
<td>25.1503</td>
<td>25.5433</td>
</tr>
<tr>
<td>FR SVC (c)</td>
<td>67.7448</td>
<td>10.1175</td>
<td>77.8623</td>
</tr>
<tr>
<td>FR SVC (s)</td>
<td>0.3847†</td>
<td>10.1175</td>
<td>10.5021†</td>
</tr>
<tr>
<td>FSSVC</td>
<td>0.9377</td>
<td>14.6621 (QPSSVM)</td>
<td>1.9988†</td>
</tr>
<tr>
<td>ETSVC-1-1</td>
<td>0.9242</td>
<td>4.8558</td>
<td>17.0882</td>
</tr>
<tr>
<td>ETSVC-1-2</td>
<td>0.8990</td>
<td>4.8205</td>
<td>13.5877</td>
</tr>
<tr>
<td>ETSVC-1-3</td>
<td>0.9103</td>
<td>4.8270</td>
<td>13.6311</td>
</tr>
<tr>
<td>ETSVC-2-1</td>
<td>0.8955</td>
<td>4.8137</td>
<td>4.2940</td>
</tr>
<tr>
<td>ETSVC-2-2</td>
<td>0.8943</td>
<td>4.8568</td>
<td>4.2704</td>
</tr>
<tr>
<td>ETSVC-2-3</td>
<td>0.8881</td>
<td>4.8087†</td>
<td>4.3365</td>
</tr>
<tr>
<td>ETSVC-2-4</td>
<td>0.8887</td>
<td>4.8362</td>
<td>4.3047</td>
</tr>
<tr>
<td>ETSVC-2-5</td>
<td>0.8822†</td>
<td>4.8399</td>
<td>4.2767</td>
</tr>
</tbody>
</table>

Note: † marks the 1st rank in each phase of two types of methods. ‡ uses the pre-computed kernel (no construction time).

4) Performance of analyzing intrusion flows

In Section V-C3, we give the evidence of representative prototypes benefiting efficient cluster analysis. In fact, an effective clustering method critically contributes accurate prototype-finding in some perspective. Facing large-scale traffic analysis rather than UNIBS-AIT, one may expect cluster analysis extracting representative samples for fast building intrusion detection model, especially for a low-end device. Unfortunately, we seriously find that the size of KD’99 is about 53.64 times of UNIBS-AIT. If we continue the strategy of using the pre-computed kernel matrix, FR SVC(s) requires more than 450 GB memory which is far greater than what our platform can afford. Similarly, methods such as PSVC and CDCL also cannot be afforded due to similar prerequisites with FR SVC(s) for efficiency in practical use. As analyzed in Table 1, the cost for transferring the time complexity $O(N^3)$ of the training phase into the strategy of calculating on demand is too huge to work. The gap between FR SVC(c) and FR SVC(s) provides strong evidence. Furthermore, without an appropriate solution for training, the labeling phase has to face the challenges of either improper cluster profiles or too huge size of SVs that seriously reduces its applicability. Actually, we also find that FSVC, VCC, and FR SVC(c) cannot finish the analysis of intrusion flows in KD’99 in a reasonable time, i.e., 3 hours (10,800 s.) in this case.

1Generally, a numeric type of float requires 4 Bytes.
Even though the compared six methods can hardly deal with KDD’99 in a reasonable time, FRSVC has been considered as one of the best SVCs in the literature [1]. We take FRSVC as the baseline to verify the capability of ETSVC in terms of accuracy without time cost consideration. Results are provided in Table 6, where the achieved ARIs greater than FRSVC, are marked with † and the first rank is highlighted by boldface. ARI of ETSVC-1-1 is marked with “—” because its pricey sampling strategy is employing the full SV pairs in the labeling phase. Besides ETSVC-1-1, apparently, all the other seven variants of ETSVC perform better than the baseline, no mater for accuracy or efficiency. Amongst these ETSVCs, ETSVC-2-5 reaches the highest accuracy in 6336.1644 seconds in which runtimes for EdgeS, HyperSC, and the labeling phase are respectively 6259.7362, 35.6069, and 40.8213. In terms of detection rate, we also give ETSVC-2-5’s performance in which runtimes for EdgeS, HyperSC, and the labeling phase. Besides ETSVC-1-1, apparently, all the other seven variants of ETSVC perform better than the baseline, no matter for accuracy or efficiency. Considering the values of \( p \) and \( p' \) corresponding to each comparison is obtained. Obviously, ETSVC-2-3 reaches the best performance in the view of average rank. Considering the values of \( p' \), we can also confirm that ETSVCs outperform the others. Meanwhile, there are several groups of methods with comparable performance, i.e., {ETSVC-2-2, ETSVC-2-1, ETSVC-2-4}, {ETSVC-1-2, ETSVC-1-3}, and {ETSVC-1-1, ETSVC-2-5}, and {FRSVC, FSSVC}.

### TABLE 6: Clustering analysis for intrusion flows KDD’99 by ETSVCs and FRSVC

<table>
<thead>
<tr>
<th>Method</th>
<th>ARI</th>
<th>Method</th>
<th>ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRSVC</td>
<td>0.7401</td>
<td>—</td>
<td>0.8672</td>
</tr>
<tr>
<td>ETSVC-1-1</td>
<td>—</td>
<td>ETSVC-2-2</td>
<td>0.8437</td>
</tr>
<tr>
<td>ETSVC-1-2</td>
<td>0.8437†</td>
<td>ETSVC-2-3</td>
<td>0.8419†</td>
</tr>
<tr>
<td>ETSVC-1-3</td>
<td>0.8419†</td>
<td>ETSVC-2-4</td>
<td>0.7782†</td>
</tr>
<tr>
<td>ETSVC-2-1</td>
<td>0.8422†</td>
<td>ETSVC-2-5</td>
<td>0.8677†</td>
</tr>
</tbody>
</table>

Note: “—” means not available or more than 3h.

### TABLE 7: Detection performance of ETSVC-2-5 on five classes of flows in KDD’99

<table>
<thead>
<tr>
<th>Normal (%)</th>
<th>DoS (%)</th>
<th>Phb (%)</th>
<th>R2L (%)</th>
<th>U2R (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>95.50</td>
<td>99.32</td>
<td>34.50</td>
<td>7.64</td>
<td>5.77</td>
</tr>
</tbody>
</table>

5) Performance on non-traffic domains data

In this section, we conduct additional experiments on various non-traffic domains data to check whether the proposed ETSVC performs well. These data sets include Wisconsin, WebKB, Reuters, Ohsumed, Abalone and Shuttle. The statistics and domain information of six data sets are listed in Table 8. Notice that WebKB, Reuters and Ohsumed are pre-processed following D\(_{G_{L-C}}\)CE by [35]. Since efficiency analysis has been made in the prior experiments, we evaluate ETSVC’s performance on non-traffic domains data in terms of accuracy. Results are listed in Table 9 where the first rank for each test is highlighted by boldface and marked with † while ‡ is used for the second rank. On Shuttle, we directly use the accuracy achieved by FSV in [10] for fairness.

In Table 9, we can find different performances of ETSVC with different labeling strategies. Nevertheless, ETSVCs always reach the first rank except on Wisconsin. Among these methods, there are significant superiorities for ETSVCs while dealing with Reuters, Ohsumed, Abalone, and Shuttle. Taking UNIBS-AIT and KDD’99 into account, ETSVCs frequently perform better than the state-of-the-art methods on data sets with more samples. If we fix the dimensionality, data set with more samples usually has clearer shape than those data sets with fewer samples. Based on ARI measures, we also give results of pair comparisons in Table 10 following the work of [40]. Among the proposed ETSVCs, we choose ETSVC-2-3 as the control method. A nonparametric statistical test of Friedman test [41] is employed to get the average ranks and unadjusted \( p \) values. By introducing an adjustment method Holm procedure, the adjusted \( p \)-value denoted by \( p_{Holm} \) corresponding to each comparison is obtained. Obviously, ETSVC-2-3 reaches the best performance in the view of average rank. Considering the values of \( p_{Holm} \), we can also confirm that ETSVCs outperform the others. Meanwhile, there are several groups of methods with comparable performance, i.e., {ETSVC-2-2, ETSVC-2-1, ETSVC-2-4}, {ETSVC-1-2, ETSVC-1-3}, {ETSVC-1-1, ETSVC-2-5}, and {FRSVC, FSSVC}.

### TABLE 9: Accuracies (ARI) achieved by the state-of-the-art methods on six non-traffic domains data

<table>
<thead>
<tr>
<th>Methods</th>
<th>Wisconsin</th>
<th>WebKB</th>
<th>Reuters</th>
<th>Ohsumed</th>
<th>Abalone</th>
<th>Shuttle</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSV</td>
<td>0.6687</td>
<td>0.5144</td>
<td>0.4775</td>
<td>—</td>
<td>0.8055</td>
<td>0.5810</td>
</tr>
<tr>
<td>PSV</td>
<td>0.2574</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>CDCL</td>
<td>0.8685</td>
<td>0.4645</td>
<td>0.8064†</td>
<td>—</td>
<td>0.6063</td>
<td>—</td>
</tr>
<tr>
<td>VCC</td>
<td>0.8029</td>
<td>0.4434</td>
<td>0.4908</td>
<td>0.4280</td>
<td>0.0710</td>
<td>0.5898</td>
</tr>
<tr>
<td>FRSVC</td>
<td>0.8798</td>
<td>0.6395†</td>
<td>0.7295</td>
<td>0.4840†</td>
<td>0.6578</td>
<td>0.8050</td>
</tr>
<tr>
<td>FSSVC</td>
<td>0.9248†</td>
<td>0.5670</td>
<td>0.5831</td>
<td>0.4514</td>
<td>0.5587</td>
<td>0.6857</td>
</tr>
<tr>
<td>ETSVC-1-1</td>
<td>0.9078‡</td>
<td>0.4928</td>
<td>0.8408†</td>
<td>0.4925‡</td>
<td>0.7090‡</td>
<td>0.6709‡</td>
</tr>
<tr>
<td>ETSVC-1-2</td>
<td>0.9078‡</td>
<td>0.5068</td>
<td>0.8408†</td>
<td>0.4829</td>
<td>0.0790‡</td>
<td>0.8901‡</td>
</tr>
<tr>
<td>ETSVC-2-1</td>
<td>0.9078†</td>
<td>0.5068</td>
<td>0.8408†</td>
<td>0.4925‡</td>
<td>0.0797†</td>
<td>0.8928†</td>
</tr>
<tr>
<td>ETSVC-2-2</td>
<td>0.9078†</td>
<td>0.5068</td>
<td>0.8408†</td>
<td>0.4925‡</td>
<td>0.0797†</td>
<td>0.8928†</td>
</tr>
<tr>
<td>ETSVC-2-3</td>
<td>0.9078†</td>
<td>0.5068</td>
<td>0.8408†</td>
<td>0.4925‡</td>
<td>0.0797†</td>
<td>0.8928†</td>
</tr>
<tr>
<td>ETSVC-2-4</td>
<td>0.9078†</td>
<td>0.5068</td>
<td>0.8408†</td>
<td>0.4925‡</td>
<td>0.0797†</td>
<td>0.9017‡</td>
</tr>
<tr>
<td>ETSVC-2-5</td>
<td>0.8963§</td>
<td>0.6398†</td>
<td>0.7932</td>
<td>0.4837</td>
<td>0.0759</td>
<td>0.8843</td>
</tr>
</tbody>
</table>

Note: † and ‡ mark the 1st and 2nd ranks, respectively. “—” means not available or more than 3h.

### VI. CONCLUSION

Towards making large-scale data analysis affordable by resource-limit platform, we have proposed ETSVC with an accurate boundary selection strategy and a flexible RSolver for our hypersphere construction problem. The first core of ETSVC lies in the SBS method. Derived from BEPS, a smart range restriction strategy makes SBS select clearer edges.
TABLE 8: Description of various non-traffic domains data.

<table>
<thead>
<tr>
<th>Data sets</th>
<th>dims</th>
<th>size</th>
<th># of classes</th>
<th>brief info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wisconsin</td>
<td>9</td>
<td>683</td>
<td>2</td>
<td>Wisconsin breast cancer database [36]</td>
</tr>
<tr>
<td>WebKB</td>
<td>4</td>
<td>4199</td>
<td>4</td>
<td>Four categories WWW-pages collected from computer science departments of various universities [37]</td>
</tr>
<tr>
<td>Reuters</td>
<td>10</td>
<td>9990</td>
<td>10</td>
<td>The top 10 largest categories of documents appeared on Reuters newswire in 1987 [38]</td>
</tr>
<tr>
<td>Ohsumed</td>
<td>23</td>
<td>13929</td>
<td>23</td>
<td>A subset of clinically oriented MEDLINE from year 1987 to year 1991 [39]</td>
</tr>
<tr>
<td>Abalone</td>
<td>7</td>
<td>4177</td>
<td>29</td>
<td>Physical measurements of abalone for age prediction [36]</td>
</tr>
<tr>
<td>Shuttle</td>
<td>9</td>
<td>43500</td>
<td>7</td>
<td>A dataset deals with the positioning of radiators in the Space Shuttle [36]</td>
</tr>
</tbody>
</table>

TABLE 10: Comparison under non-parametric statistical test

<table>
<thead>
<tr>
<th>methods</th>
<th>average ranks</th>
<th>unadjusted p</th>
<th>P[h0m]</th>
</tr>
</thead>
<tbody>
<tr>
<td>control method</td>
<td>ETSVC-2-3</td>
<td>4.0625</td>
<td>2.5029</td>
</tr>
<tr>
<td>ETSVC-2-2</td>
<td>4.1250</td>
<td>0.9762</td>
<td>2.5029</td>
</tr>
<tr>
<td>ETSVC-2-1</td>
<td>4.3750</td>
<td>0.8812</td>
<td>2.5029</td>
</tr>
<tr>
<td>ETSVC-2-4</td>
<td>4.5000</td>
<td>0.8343</td>
<td>2.5029</td>
</tr>
<tr>
<td>ETSVC-1-2</td>
<td>5.7500</td>
<td>0.4198</td>
<td>1.6948</td>
</tr>
<tr>
<td>ETSVC-1-3</td>
<td>6.0625</td>
<td>0.3389</td>
<td>1.6948</td>
</tr>
<tr>
<td>ETSVC-1-1</td>
<td>6.5625</td>
<td>0.2319</td>
<td>1.3919</td>
</tr>
<tr>
<td>ETSVC-2-5</td>
<td>6.7500</td>
<td>0.1988</td>
<td>1.3918</td>
</tr>
<tr>
<td>FSSVC</td>
<td>8.0625</td>
<td>0.0558</td>
<td>0.4689</td>
</tr>
<tr>
<td>FRSVS</td>
<td>8.1250</td>
<td>0.0521</td>
<td>0.4689</td>
</tr>
<tr>
<td>CDCL</td>
<td>10.1250</td>
<td>0.0037</td>
<td>0.0375</td>
</tr>
<tr>
<td>FSVS</td>
<td>11.4375</td>
<td>4.221E-4</td>
<td>0.0046</td>
</tr>
<tr>
<td>VCC</td>
<td>11.5625</td>
<td>3.3619E-4</td>
<td>0.0040</td>
</tr>
<tr>
<td>PSVC</td>
<td>13.5000</td>
<td>6.4223E-6</td>
<td>8.3489E-5</td>
</tr>
</tbody>
</table>

without noises. It receives an additional ability of making the cluster boundary shrinkable to a certain degree. By taking SBS as the first step of ETSVC, a nest and separable subset of traffic frequently brings not only efficiency improvement for the following solver but also adaptability for labeling strategy. The second core is the design of RSolver for the reformulated dual problem (11) in hypersphere construction. Based on the shrunk and meaningful boundaries, RSolver can use a relatively small matrix $H$ to finish lightweight operations iteratively. For traffic analysis, we, fortunately, find that the final accuracy is not firmly related to the number of iterations. Therefore, RSolver contributes the excepted accuracy with a significantly reduced iteration number. Theoretical analysis and experimental results on both UNIBS-AIT and KDD’99 confirm the flexibility and applicability of ETSVC with different labeling strategies. Besides, additional experiments on non-traffic domains data also show the evidence of ETSVC’s generalizability and excellent performance.

Although ETSVC features by efficiency, flexibility, applicability in traffic analysis, further improvements on the efficiency of SBS and flexibility of matrix construction in RSolver become ever-lasting issues. How to make full use of distributed computing for boundary selection, finding substitutes for the complex operations and easing suffers from traffic with seriously imbalanced distribution are worthy of further investigation.

ACKNOWLEDGMENT
(Yuan Ping and Bin Hao are co-first authors.)

REFERENCES


YUAN PING received the B.S. degree in electronics and information engineering from South-west Normal University in 2003, the M.S. degree in mathematics from He‘nan University in 2008, and the Ph.D. degree in information security from Beijing University of Posts and Telecommunications in 2012. He is an associate professor with Xuchang University. He was a visiting scholar with the School of Computer and Informatics, University of Louisiana at Lafayette and with the Department of Computer Science, University of Alberta. His research interests include machine learning, public key cryptography, data privacy and security, cloud and edge computing.

BIN HAO received the B.S. degree in electronic and information engineering from China Agricultural University, Beijing, China, in 2004, the M.S. degree in signal and information processing from North China University of Technology, Beijing, China, in 2007, and the Ph.D. degree in computer science from Beijing University of Posts and Telecommunications, Beijing, China, in 2012. He is currently a Postdoctoral Fellow at School of Computing and Informatics, University of Louisiana at Lafayette, Louisiana, USA. His research interests focus on machine learning, acoustical channel based access control, wireless device security, key agreement protocol, and trusted computing.

HUINA LI received the B.S. degree in electronics and information engineering from Huazhong Normal University in 2003, and the M.S. degree in mathematics from He‘nan University in 2008. She is an associate professor with the School of Information Engineering, Xuchang University, China. Her research interests include machine learning, data mining, signal processing.

YUPING LAI has been an associate professor at North China University of Technology, China, since 2019. He received his Ph.D. degree in Information Security from Beijing University of Posts and Telecommunications, Beijing, China, in 2014. His research interests include information security, computer vision, pattern recognition, machine learning, and data mining.

CHUN GUO received B.Sc. and M.Sc in July 2008 and July 2011 from Guizhou University, respectively, and received Ph.D. in information security from Beijing University of Posts and Telecommunications in July 2014. He is currently an associate professor in the College of Computer Science and Technology, Guizhou University, P.R. China. His research interests include data mining, intrusion detection, and intrusion defense.
HUI MA received the B.S. degree in Computer Application Technology from Information engineering University in 1996, the M.S. degree in Computer Software and Theory from Information engineering University in 2001. She is an associate professor with Xuchang University and a visiting scholar with the School of Telecommunication Engineering, Xidian University. Her research interests include network security, machine learning, and cloud computing.

BAOCANG WANG received the B.S. and M.S. degrees in mathematics, and the Ph.D. degree in cryptography from Xidian University in 2001, 2004, and 2006, respectively. He is currently a professor with the School of Telecommunications Engineering, Xidian University, and partially with the School of Information Engineering, Xuchang University. His main research interests include public key cryptography, wireless network security, and data mining.

XIALI HEI received the B.S. degree in electrical engineering from Xi’an Jiaotong University, Xi’an, China, in 2002, the M.S. degree in software engineering from Tsinghua University, Beijing, China, in 2005, and the Ph.D. degree in computer science from Temple University in 2014. She is an assistant professor in the School of Computing and Informatics at the University of Louisiana at Lafayette. Prior to joining the University of Louisiana at Lafayette, she was an assistant professor at Delaware State University from 2015-2017 and Frostburg State University 2014-2015. Her research interests are secure real-time wireless medical devices, vulnerability assessment and malware detection on Android, and efficient encryption schemes design. She was awarded NSF CRII grant and Delaware DEDO grant. She got several awards such as: ACM 2014 MobiHoc Best Poster Runner-up Award, Dissertation Completion Fellowship, The Bronze Award Best Graduate Project in Future of Computing Competition, IEEE INFOCOM and IEEE GLOBECOM student travel grant, etc. She is the TPC member of USENIX Security, IEEE GLOBECOM, IEEE ICC, WASA, etc.

***