IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 31, 2021, accepted February 11, 2021, date of publication February 16, 2021, date of current version March 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059648

Physical Layer Security: Detection of Active Eavesdropping Attacks by Support Vector Machines

TIEP M. HOANG[1], (Member, IEEE), TRUNG Q. DUONG[2], (Senior Member, IEEE), HOANG DUONG TUAN[3], (Member, IEEE), SANGARAPILLAI LAMBOTHARAN[4], (Senior Member, IEEE), AND LAJOS HANZO[1], (Fellow, IEEE)

[1]School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K.
[2]School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT7 1NN, U.K.
[3]School of Electrical and Data Engineering, University of Technology Sydney, Sydney, NSW 2007, Australia
[4]Wolfson School of Mechanical, Manufacturing and Electrical Engineering, Loughborough University, Loughborough LE11 3TU, U.K.

Corresponding author: Lajos Hanzo (lh@soton.ac.uk)

The work of Lajos Hanzo was supported in part by the Engineering and Physical Sciences Research Council under Project EP/P034284/1 and Project EP/P003990/1 (COALESCE) and in part by the European Research Council's Advanced Fellow Grant QuantCom under Grant 789028.

ABSTRACT This article presents a framework for converting wireless signals into structured datasets, which can be fed into machine learning algorithms for the detection of active eavesdropping attacks at the physical layer. More specifically, a wireless communication system, which consists of an access point (AP), K legitimate users and an active eavesdropper, is considered. To detect the eavesdropper who breaks into the system during the authentication phase, we first build structured datasets based on different features and then apply sophisticated support vector machine (SVM) classifiers to those structured datasets. To be more specific, we first process the signals received by the AP and then define a pair of statistical features based on the post-processing of the signals. By arranging for the AP to simulate the entire process of transmission and the process of constructing features, we form the so-called artificial training data (ATD). By training SVM classifiers on the ATD, we classify the received signals associated with eavesdropping attacks and non-attacks, thereby detecting the presence of the eavesdropper. Two SVM classifiers are considered, including a classic twin-class SVM (TC-SVM) and a single-class SVM (SC-SVM). While the TC-SVM is preferred in the case of having *perfect channel state information (CSI) of all channels*, the SC-SVM is preferred in the realistic scenario when we have *only the CSI of legitimate users*. We also evaluate the accuracy of the trained models depending on the choice of kernel functions, the choice of features and on the eavesdropper's power. Our numerical results show that careful parameter-tuning is required for exceeding an eavesdropper detection probability of 95%.

INDEX TERMS Physical layer security, active eavesdropping, machine learning, support vector machine (SVM), single-class SVM.

I. INTRODUCTION

Information-security is of paramount importance, inspiring the research of physical layer security (PLS) in wireless systems. In contrast to security solutions operating at the upper ISO layers, PLS exploits the the random nature of wireless channels in order to guard agains eavesdroppers (E) [1]–[3].

In the context of PLS, typically two different types of eavesdroppers are discussed: i) passive eavesdroppers who only listen and ii) active eavesdroppers who break into the system by impersonating legitimate users. Naturally, an active E is more destructive than a passive one, because the amount of information leaked to the active E is higher [3], [4]. Hence, numerous contributions have dealt with active eavesdropping [3], [5]–[7]. A promising solution is to exploit the detection of eavesdroppers.

The associate editor coordinating the review of this manuscript and approving it for publication was Alessandra De Benedictis.

TABLE 1. Contrasting our Contributions to the Related Literature

	Related Contributions (Years & Papers)											This work
	2008 [8]	2014 [9]	2014 [10]	2016 [11]	2017 [12]	2017 [13]	2017 [14]	2018 [15]	2019 [16]	2019 [17]	2020 [18]	
TOA			✓								✓	
RSS			✓								✓	
Pearson correlation						✓		✓				
Euclidean distance						✓		✓				
I-Q components									✓			
CFO		✓										
Channel Probing	✓											
MEAN & RATIO												✓
Single-class data												✓
Hypothesis Testing	✓											
k-NN										✓		
k-means												
LDA			✓									
TC-SVM			✓				✓			✓		✓
SC-SVM												✓
Gaussian Mixture					✓			✓				
Decision Tree										✓		
Neural Networks									✓			
Reinforcement learning				✓							✓	

In fact, if a transmitter is unaware of E's presence, then it might broadcast its signals without hesitation. By contrast, if this transmitter is aware of the security risks, then it can generate artificial noise for drowning out the eavesdroppers, but this method consumes a precious portion of the power budget [2]. By contrast, if a transmitter can detect the eavesdroppers, then it can activate sophisticated countermeasures. In short, intrusion detection has to be addressed in all security problems, including PLS. When it comes to applying machine learning to intrusion detection, most of the previous contributions focus on the upper ISO layers, where the personal datasets of users, e.g., the faces, the gaits and other features [19]–[21], are available. Nonetheless, there are impressive contributions on authentication [8]–[11], [13]–[18], but apart from [12], there is a paucity of physical-layer-related intrusion detection contributions based on machine learning.

To elaborate, Hou *et al.* [9] suggest using carrier frequency offset (CFO) along with Kalman filtering, while Xiao *et al.* [8] suggest channel probing and hypothesis testing. However, no machine learning aided intelligence is used in the context of these works. By contrast, Xiao *et al.* [11] use Q-learning and Dyna-Q in combination with game theory, while Lu *et al.* [18] also apply Q-learning for enhancing the PLS of vehicular ad-hoc networks. As a further advance, extreme learning machines relying on *single hidden layer* based feedforward neural networks have been introduced by Wang *et al.* [13]. In [15], Qiu *et al.* use Gaussian mixture models to identify spoofing attacks, while Pan *et al.* [17] rely on industrial datasets as the input data and compare the detection performance of four algorithms, including a twin-class support vector machine (TC-SVM) algorithm, the classic k-nearest neighbors (k-NN) and decision trees. Pei *et al.* [10] consider both TC-SVM and Fisher's linear discriminant analysis (Fisher's LDA) in order to identify the channels compromised by an eavesdropper. TC-SVM is also applied to the PLS problems of the smart grid [14], while in-situ machine learning is applied in [16]. Again, a range of different machine learning algorithms has been applied for enhancing a range of PLS metrics, but the detection of attacks by E has received limited attention.

Hence, despite the efforts suggested, numerous PLS-related aspects remain unsolved. The authors of [10], [11], [13], [14], [16], [18] do not consider the impact of wireless propagation on realistic modulated signals. As an exception, Weinand *et al.* [12] proactively handle the wireless propagation of realistic signals using machine learning algorithms. Similar to [12], we also consider the transmission process by exploiting the relationship between the transmitted and received signals. However, in contrast to [12] that uses Gaussian mixture models for classification, we demonstrate the superiority of both the TC-SVM and of the single-class SVM (SC-SVM) models in this article. Explicitly, compared to Gaussian mixture models, TC-SVM and SC-SVM do not rely on probabilistic assumptions, but on the distance of a data point from another. In the typical scenario, when the distribution of the data is unknown, Gaussian mixture models may not be suitable choices.

When it comes to creating features for characterizing the data extracted from wireless signals, the authors of [10], [13], [14], [16], [18] suggest sophisticated techniques for detecting E. Lu *et al.* [18] suggest using the received signal strength (RSS) indicator and the time of arrival (TOA) of a packet as a pair of unique features of a data point.

Pei *et al*. [10] consider three features, including the TOA, the RSS and the Z-transform. In [13], the feature space is constructed by Wang *et al*. based on formulating both the Pearson correlation coefficient and the Euclidean distance between two samples. In [14], each feature has a certain distribution that is fitted to the preprocessed data by Esmalifalak *et al*. As a further advance, Weinand *et al*. [12] consider channel estimation and use the normalized magnitudes of the estimated channels as the features of the input data. Furthermore, according to Chatterjee *et al*. [16], the in-phase and quadrature (I-Q) components along with the CFO can also be used to create features. However, no open-source solution has emerged as a clear winner for constructing the features of a dataset, especially not at the intersection of PLS and machine learning. Accordingly, the judicious question arises: how wireless PLS signals can be processed before entering them into a machine learning algorithm for intrusion detection? Since the data is one of the most important pillars of machine learning [22], it is necessary to answer this open question. Moreover, according to [23], the choice of suitable data features can reduce the cost of 5G authentication systems and make them more reliable.

Hence, we conceive a framework of creating the data from wireless PLS signals for ensuring that all the features of the data exhibit the required statistical properties. Using the resultant data, we detect active eavesdropping attacks with the aid of both TC-SVM and SC-SVM. Specifically, our contributions can be summarized as follows.

- We process the received signals by transforming them into a *structured dataset*, which consists of two different features (namely, the MEAN and the RATIO). The features are specially formulated for ensuring that they are associated with certain constants. Moreover, data points in a feature column satisfy the particular requirement that the newer a data point, the closer it approaches the corresponding constant.
- We construct two specific types of so-called *artificial training data* (ATD), explicitly depending on whether the channel state information (CSI) is known or unknown, we formulate two types of ATD. If the CSI of an active E's channel is known, then the ATD has two classes, which can then be used for training our TC-SVM models. By contrast, if the CSI of a passive eavesdropper's channel is unknown, the ATD has a single class, which will be used for SC-SVM models. The ATD concept introduced in this article can also be readily applied to other types of supervised and unsupervised learning models.
- By using TC-SVM and SC-SVM, we evaluate the performance of our solution conceived for detecting the eavesdropper's attacks. We then critically appraise the pros and cons of four kernel functions.

The remainder of the article is organized as follows. Section II presents the principles of TC-SVM and SC-SVM. In Section III, we discuss how to construct the features of a dataset based on the reception of wireless signals during the authentication period. Section IV introduces two different types of artificial training data, which are used by TC-SVM and SC-SVM, respectively. Finally, our numerical results and conclusions are provided in Sections V and VI, respectively.

Notations: $\text{sign}(z) = 1$ if $z > 0$ and $\text{sign}(z) = -1$ if $z \leq 0$. Vectors and matrices are represented by lowercase boldface and uppercase boldface, respectively; $[\cdot]^T$ and $[\cdot]^\dagger$ denote the transpose operator, and Hermitian operator, respectively; $\mathbf{I}_L$ is the $L \times L$ identity matrix; $\|\cdot\|$ denotes the Euclidean norm; $\mathbf{z} \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}_L)$ denotes a complex Gaussian random vector $\mathbf{z} \in \mathbb{C}^{L\times 1}$ with zero-mean and covariance $\mathbf{I}_L$; $\mathbb{E}_{\mathbf{v}_1,\ldots,\mathbf{v}_M}\{\cdot\}$ denotes the expectation over $\mathbf{v}_1, \ldots, \mathbf{v}_M$. Some important symbols are defined in Table 2, while the remaining symbols will be defined whenever used.

II. A BRIEF INTRODUCTION TO TC-SVM AND SC-SVM

A. THE DEVELOPMENT OF TYPICAL MODELS

In the 1960s, the perceptron was suggested using the functional relationship $l = \text{sign}(\langle \mathbf{w}, \mathbf{r} \rangle + b)$ between the input vector $\mathbf{r} \in \mathcal{R} \subseteq \mathbb{C}^{n\times 1}$ and the output value l. Herein, l is referred to as the label for the input vector $\mathbf{r}$ and, assigning a value (say 1 or -1) to l represents the classification of the input vector. For example, if $l = 1$ means that $\mathbf{r}$ has some attribute A, then $l = -1$ can be used to show that $\mathbf{r}$ is not of that attribute A. Due to the above-mentioned functional relationship, the input space $\mathcal{R}$ is separated into two regions by the hyperplane $\langle \mathbf{w}, \mathbf{r} \rangle - b = 0$. In one of the two regions, l can take the value of 1, while in the other region, the value of l is (-1). Given that the binary classification depends on the position of the separating hyperplane, during the learning process the perceptron seeks the most suitable coefficients $\mathbf{w}$ and b for ensuring that assigning a value to l (i.e. classifying $\mathbf{r}$) is as close to optimal as possible.

Upon replacing $\text{sign}(\cdot)$ by a continuous sigmoid function, such as $\tanh(\cdot)$, *multi-layer perceptrons* (MLPs) were formed. In many cases [24]–[26], the so-called support vector machines (SVMs) typically provide better results than the classical MLPs. Various SVM classifiers may be constructed based on different types of kernel functions, such as the *radial basis function* (RBF) kernel and the sigmoid kernel. The choice of the kernel functions is quite pivotal in terms of its complexity. One of the main advantages of SVM classifiers is the ability to perform well even on small data sets.

B. KERNEL METHODS

Prior to describing the operation of SVMs, we will clarify the role of kernel functions in SVM classifiers. Let $\phi(\cdot) : \mathcal{R} \to \mathcal{H}$ be a nonlinear mapping that casts the *input space* $\mathcal{R}$ to a higher-dimensional space $\mathcal{H}$ (the *feature space*). If $\mathbf{x}_t \in \mathcal{R}$ and $\mathbf{x}_{t'} \in \mathcal{R}$, then the inner product $\mathcal{K}(\mathbf{x}_t, \mathbf{x}_{t'}) = \langle \phi(\mathbf{x}_t), \phi(\mathbf{x}_{t'}) \rangle$ is a kernel function [27], [28]. The main rationale of using the kernel function is that once it has been given beforehand, one can directly compute $\mathcal{K}(\mathbf{x}_t, \mathbf{x}_{t'})$ from $\mathbf{x}_t$ and $\mathbf{x}_{t'}$ without necessarily having explicit expressions for $\phi(\mathbf{x}_t)$ and $\phi(\mathbf{x}_{t'})$ during learning. That helps us to accelerate the

TABLE 2. Notation

Symbols	Meanings
$\mathcal{R}, \mathcal{H}$	Input space and feature space, respectively.
$\phi(\mathbf{x}_s)$	The image of a training sample $\mathbf{x}_s$ in $\mathcal{H}$.
$\mathcal{K}(\mathbf{x}_s, \mathbf{x})$	The kernel function that computes the inner products between $\phi(\mathbf{x}_s)$ and $\phi(\mathbf{x})$ in $\mathcal{H}$.
C	The regularization parameter in SVM.
γ	The parameter that appears in kernel functions, e.g., radius basis function (RBF), polynomial, and sigmoid kernels.
$\left\{\varphi_k^{(i)}\right\}_{i=1}^{2}$	The *ideal* features of training data used to check if an eavesdropper is attacking user k. These two features are theoretically formulated.
$\left\{f_k^{(i)}[t]\right\}_{i=1}^{3}$	The practical features of the t-th training sample in training data. These two features are formulated based on post-processing signals.
$\left\{\widehat{f}_k^{(i)}\left[\widehat{t}\right]\right\}_{i=1}^{3}$	The features of the $\widehat{t}$-th training sample in ATD.
T	When a certain pilot is requested for authentication during uplink phase, it is repeatedly sent T times.
$\widehat{T}$	The number of *artificial* training samples that are stuck with the label (#1) (or the label (#0)).
T_{tot}	The total number of training samples that are fed into SVM algorithms. In numerical results, $T_{tot} = 2\widehat{T}$.
(#1), $\mathcal{H}_1$	There is an eavesdropping attack.
(#0), $\mathcal{H}_0$	There is not any eavesdropping attack.

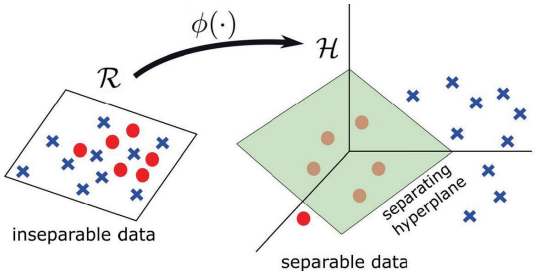

FIGURE 1. Using kernel methods, the data in the input space can become more linearly separable in a higher-dimensional space.

computational speed. Moreover, the kernel function allows us to avoid working in $\mathcal{R}$; instead, we only evaluate the input samples using the inner product in $\mathcal{H}$. From the viewpoint of data classification, the map $\phi(\cdot)$ can help us to convert *linearly inseparable* data (in $\mathcal{R}$) into linearly separable structures (in $\mathcal{H}$) [28]. As an example, Figure 1 illustrates the reason for mapping data points in $\mathcal{R}$ to the associated data points in $\mathcal{H}$ in a way that the crosses and circles become separable by the hyperplane.

C. CLASSIFICATION BASED ON TWIN-CLASS SVM

The binary classification problem based on SVM can be formulated as follows [28], [29]:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \tag{1a}$$

$$\text{s.t. } l_\omega\left(\langle \mathbf{w}, \mathbf{x}_\omega\rangle + b\right) \geq 1, \tag{1b}$$

where $\mathbf{x}_\omega$ is the position vector of the ω-th sample, $\omega \in \Omega \triangleq \{1, 2, \ldots, T_{tot}\}$, and T_{tot} is the total number of training samples. In (1b), $l_\omega = +1$ if $\mathbf{x}_\omega$ is labelled as (#1); otherwise, $l_\omega = -1$ if $\mathbf{x}_\omega$ is labelled as (#0).

As illustrated in Figure 2, the samples are separated by a hyperplane that takes the form of $h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x}\rangle + b = 0$. The separating hyperplane is in the middle of two margins, and the Euclidean distance between the two margins is equal to $2/\|\mathbf{w}\|$. The goal of (1) is to maximize the margin width $2/\|\mathbf{w}\|$, so that the samples are correctly separated. Note that maximizing $2/\|\mathbf{w}\|$ is equivalent to (1a), while the correct separation is equivalent to (1b).

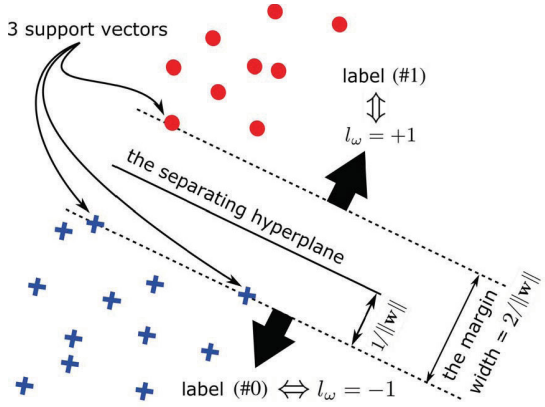

FIGURE 2. A dataset, with two types of samples, is separated by a separating hyperplane. The ω-th sample, which is positioned at $\mathbf{x}_\omega$, will be labelled as (#1) (or $l_\omega = +1$) if it lies above the separating hyperplane. Otherwise, it will be labelled as (#0) (or $l_\omega = -1$).

Let $\Omega_{(\#0)}$ and $\Omega_{(\#1)}$ denote the set of indices $\{\omega \in \Omega | l_\omega = -1\}$ and the set of indices $\{\omega \in \Omega | l_\omega = 1\}$, respectively. If (1) has optimal solutions of $\mathbf{w} = \mathbf{w}_\star$ and $b = b_\star$, then we say that $\mathcal{X} = \{(\mathbf{x}_\omega, l_\omega) | \omega \in \Omega\}$ are *linearly separable*. In this case, there exists at least one separating hyperplane that separates the samples into two sub-sets, i.e., $\mathcal{X}_{(\#0)} = \left\{(\mathbf{x}_\omega, l_\omega) | \omega \in \Omega_{(\#0)}\right\}$ and $\mathcal{X}_{(\#1)} = \mathcal{X} \setminus \mathcal{X}_{(\#0)}$. Furthermore, the optimal separating hyperplane among all possible hyperplanes will have the following equation:

$$h_\star(\mathbf{x}) = \langle \mathbf{w}_\star, \mathbf{x}\rangle + b_\star = 0. \tag{2}$$

Note that $\mathbf{x}$ in (2) is not necessarily the same as $\mathbf{x}_\omega$ ($\omega \in \Omega$). A certain labelled sample $\mathbf{x}_\omega$ ($\omega \in \Omega$) only lies on the optimal hyperplane if it satisfies $h_\star(\mathbf{x}_t) = 0$. Given the hyperplane $h_\star(\mathbf{x})$, the input space is divided into two sub-spaces, where one of the sub-spaces contains all samples $\mathbf{x}_\omega \in \mathcal{X}_{(\#0)}$, while the other contains all samples $\mathbf{x}_\omega \in \mathcal{X}_{(\#1)}$.

It should be noted that (1) may not work in numerous practical scenarios, because the labelled samples are linearly inseparable. In this case, the classification tasks simply fail to work and no hyperplane can be found. For this reason, the use of kernel methods will make classification

tasks easier when the inseparable data can be translated into a higher-dimensional feature space in which the structure of data becomes more separable [27]–[29].[1] Thus, we will replace $\mathbf{x}_\omega$ in (1) by the function $\phi(\mathbf{x}_\omega)$ so as to avoid working directly in the input space. However, the problem is that the transformed data may remain inseparable. To overcome this difficulty, slack variables $\xi_\omega \geq 0$ are added to the classification problems (see [30] and [31, ch.7]). This breakthrough method has become the background of SVM-based classification methods [30]. Following this method, one can change the original problem (1) to the following:

$$\min_{\mathbf{w},b,\xi_t} \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{\text{regularizer}} + C \underbrace{\sum_{\omega=1}^{T_{tot}} \xi_\omega}_{\text{error}} \tag{3a}$$

$$\text{s.t. } l_\omega \left(\langle \mathbf{w}, \phi\left(\mathbf{x}_\omega\right)\rangle + b\right) \geq 1 - \xi_\omega, \tag{3b}$$

where C is the regularization/margin parameter. To elaborate, (3) is called a $L1$ soft-margin in SVM terminology. If $\xi_\omega = 0$ for $\forall \omega \in \Omega$, then (3) reduces to the hard-margin SVM (1). Otherwise, with $\xi_\omega \neq 0$, we accept that the ω-th sample $\mathbf{x}_\omega$ can be misclassified [27, Ch.2]. According to [27], (3) may be transformed into an equivalent optimization problem through the use of the Lagrangian function and Karush–Kuhn–Tucker conditions, which is formulated as

$$\max_{a_1,\ldots,a_T} \sum_{\omega=1}^{T_{tot}} a_\omega - \frac{1}{2} \sum_{\substack{\omega=1 \\ \omega'=1}}^{T_{tot}} a_\omega a_{\omega'} l_\omega l_{\omega'} \mathcal{K}\left(\mathbf{x}_\omega, \mathbf{x}_{\omega'}\right) \tag{4a}$$

$$\text{s.t. } \sum_{\omega=1}^{T} a_\omega l_\omega = 0 \tag{4b}$$

$$0 \leq \alpha_\omega \leq C, \ \omega \in \Omega \tag{4c}$$

where $\mathcal{K}\left(\mathbf{x}_\omega, \mathbf{x}_{\omega'}\right) \triangleq \langle \phi(\mathbf{x}_\omega), \phi(\mathbf{x}_{\omega'})\rangle$ is the kernel function.

Remark 1: Any data point for which $a_\omega = 0$ will not contribute to the prediction of new data points, thus only the data points for which $a_\omega \neq 0$ are called support vectors (SVs). Furthermore, any data point for which $0 < a_\omega < C$ is called an unbounded SV. Also, any data point for which $a_\omega = C$ is called a bounded SV. While the unbounded SVs lie on the margins, the bounded SVs are between the two margins (see [27, p.24]). Additionally, we denote $\mathcal{S}$ as the set of indices of all SVs, $\mathcal{U}$ as the set of unbounded SV indices, and $\mathcal{S}\backslash\mathcal{U}$ as the set of bounded SV indices. Note that $\mathcal{U} \subseteq \mathcal{S} \subseteq \Omega$.

The equation of the optimal hyperplane is given by

$$h_{(\star|\text{SVM})}(\mathbf{x}) = \sum_{s\in S} a_s l_s \mathcal{K}\left(\mathbf{x}_s, \mathbf{x}\right) + b, \tag{5}$$

where b is calculated for unbounded SVs (see [27, ch.2] or [31, (7.37)])

$$b = \frac{1}{|U|} \sum_{u\in U} \left(l_u - \sum_{s\in S} a_s l_s \mathcal{K}\left(\mathbf{x}_s, \mathbf{x}_u\right) \right). \tag{6}$$

[1]Once again, Figure 1 illustrates the role of kernel methods in translating inseparable data points in the two-dimensional space into separable data points in the three-dimensional space.

Note that in (5)–(6), $\mathbf{x}_s$ is the s-th SV, while $\mathbf{x}_u$ is the u-th unbounded SV. Due to $\mathcal{U} \subseteq \mathcal{S}$, we have $\{\mathbf{x}_u | u \in \mathcal{U}\} \subseteq \{\mathbf{x}_s | s \in \mathcal{S}\}$. For the kernel $\mathcal{K}\left(\mathbf{x}_s, \mathbf{x}\right)$, we will consider four specific functions in Section V for comparison purposes.

D. CLASSIFICATION BASED ON SC-SVM

Slightly differently from classical TC-SVM models, a SC-SVM model is described as follows [32]–[34]:

$$\underset{\mathbf{w},\rho,\xi_i}{\text{minimize}} \underbrace{\frac{1}{2}\|\mathbf{w}\|^2}_{\text{regularizer}} + \underbrace{\frac{1}{\nu T_{tot}} \sum_{w=1}^{T_{tot}} \xi_w - \rho}_{\text{error}} \tag{7a}$$

$$\text{subject to } \langle \mathbf{w}, \phi(\mathbf{x}_w)\rangle \geq \rho - \xi_w. \tag{7b}$$

Herein, ρ is an offset parameter determining the distance from the origin to the hyperplane, while $\nu \in (0, 1]$ is a parameter balancing the maximal distance from the origin and the number of data points in the region created by the hyperplane [33]. Each $\xi_w \geq 0$ is a slack variable, and $\phi(\cdot) : \mathcal{X} \to \mathcal{F}$ is a mapping that casts the input space $\mathcal{X}$ to a higher-dimensional space $\mathcal{F}$ (namely, the feature space). Once this minimization problem has been solved, the decision function will be given by

$$y(\mathbf{x}) = \text{sign}\left(\mathbf{w}^T \phi(\mathbf{x}) - \rho\right). \tag{8}$$

According to [34], most of the data points are put into a region (e.g., a ball) and these data points in the set $\{\mathbf{x}_1, \ldots, \mathbf{x}_{T_{tot}}\}$ will be labelled by $+1$. At the same time, any data points outside the region will be referred to as *outliers*. Consequently, some true outliers might be potentially misclassified when being put inside the region. In short, the objective of SC-SVM is to create a region for most of the data points in $\mathcal{X}$, while the remaining ones lie outside the region and are considered as being associated with eavesdropping attacks.

III. COLLECTING WIRELESS SIGNALS AND CREATING/DEFINING FEATURES

While the previous section presented useful tools for classifying new data, this section shows the process of creating relevant features and then forming datasets.

A. COLLECTING WIRELESS DATA

We assume that a single access point (AP) supports K legitimate users in the presence of an active E. Each node is equipped with a single antenna and all nodes are randomly positioned. Let the channel between the AP and the k-th user be g_k. Similarly, the channel between the AP and E is denoted by g_{E}. We have two phases: Uplink phase for authentication and downlink phase for confidential data transmission.

1) UPLINK PHASE

For the purposes of authentication and channel estimation, the AP requests legitimate users to send some pilot vectors before it transmits confidential messages in return. The k-th

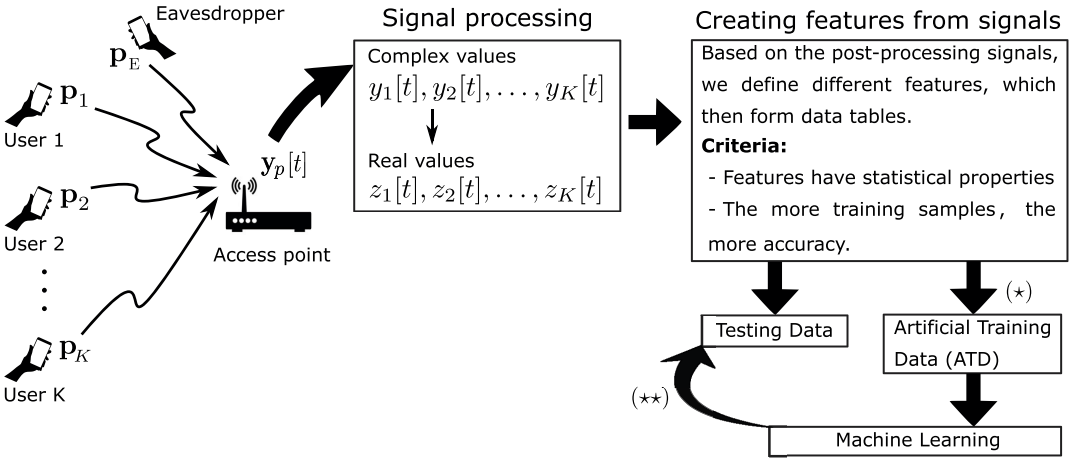

FIGURE 3. System model. At the stage ($\star$), the AP forms the ATD by imitating the uplink phase and simulating signal processing. Using machine learning algorithms, the AP has the trained models and uses them for predicting the testing data at the stage ($\star\star$).

user (or user k) is assumed to send some pilot vector $\mathbf{p}_k$ to the AP. Herein, $\mathbf{p}_k \in \mathbb{C}^{\mathcal{L}\times 1}$ is a column vector with $\mathcal{L}$ entries. By assuming $\mathcal{L} \geq K$, we can design K orthogonal pilot vectors satisfying that $\mathbf{p}_k^\dagger \mathbf{p}_{k'} = 0$ for $k \neq k'$ and $\|\mathbf{p}_k\|^2 = 1$. If E wants to overhear the signal s_k that is intended for the k-th user, E will design her pilot sequence $\mathbf{p}_{\mathrm{E}}$ to be the same as $\mathbf{p}_k$ and will send $\mathbf{p}_{\mathrm{E}}$ to the AP (see [7] and [35]). After receiving pilots from the legitimate users and E, the AP interprets these pilots as requests for information. Consequently, the AP will transmit the confidential downlink signal s_k, which is intended for user k, to both user k and E. Hence, the confidential information between the AP and the k-th user is inevitably leaked to E.

At some instance t, the AP receives the following signal

$$\mathbf{y}_p[t] = \begin{cases} \sqrt{\mathcal{L}\rho_u}\sum_{k=1}^{K}\mathbf{p}_k g_k[t] + \mathbf{n}[t], & \text{non-attack} \\ \sqrt{\mathcal{L}\rho_u}\sum_{k=1}^{K}\mathbf{p}_k g_k[t] & \\ + \sqrt{\mathcal{L}\rho_{\mathrm{E}}}\mathbf{p}_{\mathrm{E}} g_{\mathrm{E}}[t] + \mathbf{n}[t], & \text{attack,} \end{cases} \tag{9}$$

where $\rho_u \triangleq P_u/N_0$ and $\rho_{\mathrm{E}} \triangleq P_{\mathrm{E}}/N_0$. Herein, P_u and P_{E} are the average transmit power of each user and that of E, respectively; while N_0 is the average noise power per receive antenna; $\mathbf{n}$ is an additive white Gaussian noise (AWGN) vector with $\mathbf{n} \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}_{\mathcal{L}})$. Note that $\mathbf{y}_p[t]$, $g_k[t]$, $g_{\mathrm{E}}[t]$ and $\mathbf{n}[t]$ are the realizations of $\mathbf{y}_p$, g_k, g_{E} and $\mathbf{n}$ at time t, respectively.

2) DOWNLINK PHASE

In the downlink phase, the AP transmits its signals to the legitimate users. Naturally, E also receives the signals intended for user k because the AP assumes that a pair of legitimate users (i.e., user k and E) are requesting the same messages. Recall that the AP may not be aware of the presence of E and her attack.

It may be readily shown that the data rate of user k reduces, when E breaks into the uplink phase (see [7] for more details). In other words, the signal-to-noise ratio (SNR) of the k-th user, denoted as snr_k, reduces with the increase of E's power. Consequently, the difference between the data rate of user k and the data rate of E, namely the security rate, also becomes lower.

Hence, it is crucial that we detect the presence of E in the uplink phase. Once the AP has identified an eavesdropping attack, it will be able to design suitable strategies, such as the use of secure beamforming and artificial noise injection for drowning out E. Herein, we do not delve into such strategies in the downlink phase because this topic is richly documented in the literature (see [7] and references therein). Instead, sophisticated SVM-based methods of detecting active eavesdropping attacks in the uplink phase will be considered.

B. CREATING FEATURES/ATTRIBUTES

By projecting $\mathbf{y}_p[t]$ along the pilot vector $\mathbf{p}_k^\dagger$, we have the post-processed signal $y_k[t] = \mathbf{p}_k^\dagger \mathbf{y}_p[t]$, i.e.

$$y_k[t] = \begin{cases} \sqrt{\mathcal{L}\rho_u} g_k[t] + \mathbf{p}_k^\dagger \mathbf{n}[t], & \text{non-attack} \\ \sqrt{\mathcal{L}\rho_u} g_k[t] + \sqrt{\mathcal{L}\rho_{\mathrm{E}}} g_{\mathrm{E}}[t] & \\ + \mathbf{p}_k^\dagger \mathbf{n}[t], & \text{attack.} \end{cases} \tag{10}$$

Proceeding by defining $z_k[t] \triangleq |y_k[t]|^2$, the AP can calculate

$$\varphi_k^{(1)} \triangleq \mathbb{E}_t\left\{z_k[t]\right\}, \tag{11}$$

$$\varphi_k^{(2)} \triangleq \frac{\mathbb{E}_t\left\{z_k[t]\right\} - \mathbb{E}_t\left\{\left|\mathbf{p}_k^\dagger \mathbf{n}[t]\right|^2\right\}}{\mathbb{E}_t\left\{\left|\mathbf{p}_k^\dagger \mathbf{n}[t]\right|^2\right\}} \tag{12}$$

based on sufficient statistical knowledge of $\{g_k\}_{k=1}^K$ and g_{E}. Note that $\mathbb{E}_t\{\cdot\} \equiv \mathbb{E}_{\{g_k\}_{k=1}^K, g_{\mathrm{E}}, \mathbf{n}}\{\cdot\}$ due to the dependence of $\{g_k\}_{k=1}^K$, g_{E} and $\mathbf{n}$ on t.

In practice, if user k sends his/her pilot vector $\mathbf{p}_k$ to the AP T times,[2] then the AP will have T different values $z_k[1]$, ..., $z_k[T]$. Using these values, the AP will create a structured dataset that consists of the following two features:

- Feature 1 (MEAN):

$$f_k^{(1)}[T] \triangleq \frac{1}{T}\sum_{t=1}^{T} z_k[t] = \begin{cases} f_{k|\mathcal{H}_0}^{(1)}[T], & \text{non-attack} \\ f_{k|\mathcal{H}_1}^{(1)}[T], & \text{attack.} \end{cases} \quad (13)$$

  Herein, $\mathcal{H}_0$ implies that there is no eavesdropping attack, while $\mathcal{H}_0$ implies eavesdropping.
- Feature 2 (RATIO):

$$f_k^{(2)}[T] \triangleq \frac{\sum_{t=1}^{T} z_k[t] - \sum_{t=1}^{T} |\mathbf{p}_k^{\dagger}\mathbf{n}[t]|^2}{\sum_{t=1}^{T} |\mathbf{p}_k^{\dagger}\mathbf{n}[t]|^2} = \begin{cases} f_{k|\mathcal{H}_0}^{(2)}[T], & \text{non-attack} \\ f_{k|\mathcal{H}_1}^{(2)}[T], & \text{attack.} \end{cases} \quad (14)$$

Statistically, if the AP receives a sufficient number of samples (i.e., if T is large enough), we have

$$f_k^{(1)}[T] \overset{\text{large } T}{\approx} \varphi_k^{(1)}, \quad (15)$$

$$f_k^{(2)}[T] \overset{\text{large } T}{\approx} \varphi_k^{(2)}. \quad (16)$$

In short, we can create a testing dataset from wireless signals. Table 3 illustrates the two features of the testing data used in this article. The testing dataset begins with the T_0-th data point, which is related to the T_0-th time slot, $T_0 \in \{1, 2, \ldots, T\}$. If $T_0 = 1$, we use all the T data points collected during T authentication time-slots in the uplink. By contrast, if $T_0 > 1$, we only exploit $T - T_0$ data points from the T_0-th time slot to the T-th time slot. The position of the t-th data point in the 2-dimensional space is $(f_k^{(1)}[t], f_k^{(2)}[t])$. Note that the subscript k still implies that we are checking if the k-th user is under attack. It should also be noted that our testing data has not yet been labelled. In Figure 4, the impact of T on the distribution of data points is shown. It can be seen that the first data points are close together and hard to separate. However, the larger T, the more separable the data becomes.

IV. ARTIFICIAL TRAINING DATA

In this section, artificial training data (ATD) will be created. The ATD used for TC-SVM corresponds to the case of having both the legitimate users' CSI and the active E's CSI. By contrast, the ATD used for SC-SVM corresponds to the passive-E scenario of only having the legitimate users' CSI.

A. ATD FOR TWIN-CLASS SVM

The AP can create the ATD by imitating the uplink phase by performing the following steps:

- Step 1: Start with $k = 1$.
- Step 2: Start with $\widehat{t} = 1$.
- Step 3: Generate $\widehat{g}_k \overset{dist}{=} g_k$, $\widehat{g}_{\mathrm{E}} \overset{dist}{=} g_{\mathrm{E}}$ and $\widehat{\mathbf{n}} \sim \mathcal{CN}(\mathbf{0}, \mathbf{I}_{\mathcal{L}})$. Note that the notation $X \overset{dist}{=} Y$ implies that X and Y have the same distribution.

[2]Herein, $\mathbf{p}_k$ is repeatedly sent for authentication.

TABLE 3. Testing Data: T Points are Associated With the Label (#0) and T Points are Associated With the Label (#1)

Fearture 1 (MEAN)	Fearture 2 (RATIO)
$f_{k\|\mathcal{H}_0}^{(1)}[T_0]$	$f_{k\|\mathcal{H}_0}^{(2)}[T_0]$
$\vdots$	$\vdots$
$f_{k\|\mathcal{H}_0}^{(1)}[T]$	$f_{k\|\mathcal{H}_0}^{(2)}[T]$
$f_{k\|\mathcal{H}_1}^{(1)}[T_0]$	$f_{k\|\mathcal{H}_1}^{(2)}[T_0]$
$\vdots$	$\vdots$
$f_{k\|\mathcal{H}_0}^{(1)}[T]$	$f_{k\|\mathcal{H}_0}^{(2)}[T]$

- Step 4: Calculate

$$\widehat{z}_k\left[\widehat{t}\right] = \begin{cases} \left|\sqrt{\mathcal{L}\rho_u}\widehat{g}_k\left[\widehat{t}\right] + \mathbf{p}_k^{\dagger}\widehat{\mathbf{n}}\left[\widehat{t}\right]\right|^2, & \text{non-attack} \\ \left|\sqrt{\mathcal{L}\rho_u}\widehat{g}_k\left[\widehat{t}\right] + \sqrt{\mathcal{L}\rho_{\mathrm{E}}}\widehat{g}_{\mathrm{E}}\left[\widehat{t}\right]\right. & \\ \left. + \mathbf{p}_k^{\dagger}\widehat{\mathbf{n}}\left[\widehat{t}\right]\right|^2, & \text{attack} \end{cases} \quad (17)$$

$$\widehat{f}_k^{(1)}\left[\widehat{t}\right] = \frac{1}{\widehat{t}}\sum_{\varsigma=1}^{\widehat{t}} \widehat{z}_k[\varsigma] = \begin{cases} \widehat{f}_{k|\mathcal{H}_0}^{(1)}\left[\widehat{t}\right], & \text{non-attack} \\ \widehat{f}_{k|\mathcal{H}_1}^{(1)}\left[\widehat{t}\right], & \text{attack,} \end{cases} \quad (18)$$

$$\widehat{f}_k^{(2)}\left[\widehat{t}\right] = \frac{\sum_{\varsigma=1}^{\widehat{t}} \widehat{z}_k[\varsigma] - \sum_{\varsigma=1}^{\widehat{t}} |\mathbf{p}_k^{\dagger}\widehat{\mathbf{n}}[\varsigma]|^2}{\sum_{\varsigma=1}^{\widehat{t}} |\mathbf{p}_k^{\dagger}\widehat{\mathbf{n}}[\varsigma]|^2} = \begin{cases} \widehat{f}_{k|\mathcal{H}_0}^{(2)}\left[\widehat{t}\right], & \text{non-attack} \\ \widehat{f}_{k|\mathcal{H}_1}^{(2)}\left[\widehat{t}\right], & \text{attack} \end{cases} \quad (19)$$

- Step 5: When $\widehat{t} \geq T_0$, we stick the label (#0) to the $\widehat{t}$-th data point $\left(\widehat{f}_{k|\mathcal{H}_0}^{(1)}\left[\widehat{t}\right], \widehat{f}_{k|\mathcal{H}_0}^{(2)}\left[\widehat{t}\right]\right)$ in order to imply that user k is not under attack. By contrast, we stick the label (#1) to the $\widehat{t}$-th data point $\left(\widehat{f}_{k|\mathcal{H}_1}^{(1)}\left[\widehat{t}\right], \widehat{f}_{k|\mathcal{H}_1}^{(2)}\left[\widehat{t}\right]\right)$ in order to imply that user k is under attack.
- Step 6: Set $\widehat{t} = \widehat{t} + 1$ and repeat Steps 3-5. Go to Step 7 if $\widehat{t} > \widehat{T}$. Herein, $\widehat{T}$ is a large number that can be freely determined by the designers. To make the ATD statistically reliable, we choose $\widehat{T} \gg T$.
- Step 7: Set $k = k + 1$ and repeat Steps 2-6. Stop the process if $k > K$.

Remark 2: The ATD for classical TC-SVM is shown in Table 4. It will be used to train classical TC-SVM models in this article (and any other supervised learning models). Those trained models are then applied to the testing dataset in Table 2, whereby we can ascertain whether the signal obtained is affected by an active E.

Remark 3: In contrast to T, we can let $\widehat{T}$ be a very large number because $\widehat{T}$ belongs to an artificial process. For example, in the uplink phase, the AP can request both user k and E to send the same pilot vector $\mathbf{p}_k$ twenty times (i.e., $T = 20$), but cannot request them to send $\mathbf{p}_k$ too many times (e.g., $T = 2000$). However, for an artificial process carried out at the AP, it is possible to simulate the reception of a large

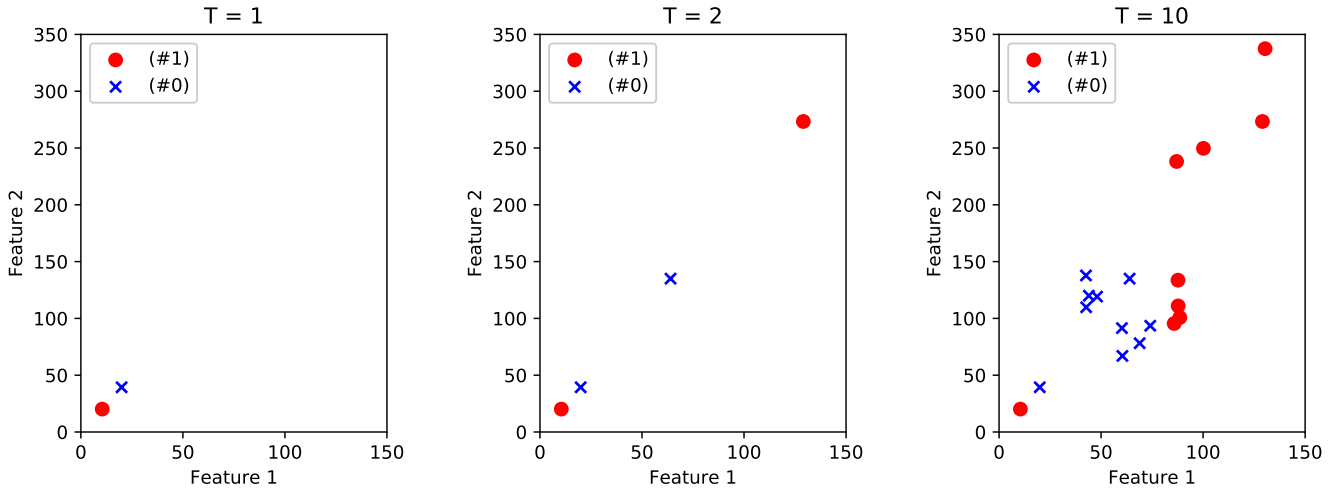

FIGURE 4. The distribution of data points in $\mathbb{R}^2$. The 1-st red dot and the 1-st blue cross are, respectively, positioned at $\left(f^{(1)}_{k|\mathcal{H}_1}[1], f^{(2)}_{k|\mathcal{H}_1}[1]\right)$ and at $\left(f^{(1)}_{k|\mathcal{H}_0}[1], f^{(2)}_{k|\mathcal{H}_0}[1]\right)$. Similarly, the 2-nd red dot and the 2-nd blue cross are, respectively, positioned at $\left(f^{(1)}_{k|\mathcal{H}_1}[2], f^{(2)}_{k|\mathcal{H}_1}[2]\right)$ and at $\left(f^{(1)}_{k|\mathcal{H}_0}[2], f^{(2)}_{k|\mathcal{H}_0}[2]\right)$. When the value of T increases, red dots tend to cluster together and become separable from blue crosses.

TABLE 4. Artificial Training Data (ATD) Used for Classical TC-SVM

MEAN	RATIO	Labels
$\widehat{f}^{(1)}_{k\|\mathcal{H}_1}[T_0]$	$\widehat{f}^{(2)}_{k\|\mathcal{H}_1}[T_0]$	(#1)
$\vdots$	$\vdots$	$\vdots$
$\widehat{f}^{(1)}_{k\|\mathcal{H}_1}[\widehat{T}]$	$\widehat{f}^{(2)}_{k\|\mathcal{H}_1}[\widehat{T}]$	(#1)
$\widehat{f}^{(1)}_{k\|\mathcal{H}_0}[T_0]$	$\widehat{f}^{(2)}_{k\|\mathcal{H}_0}[T_0]$	(#0)
$\vdots$	$\vdots$	$\vdots$
$\widehat{f}^{(1)}_{k\|\mathcal{H}_0}[\widehat{T}]$	$\widehat{f}^{(2)}_{k\|\mathcal{H}_0}[\widehat{T}]$	(#0)

number of copies of $\mathbf{p}_k$ as desired. Given sufficiently large $\widehat{T}$, we can reach $\widehat{f}^{(1)}_k[\widehat{T}]/\varphi^{(1)}_k \approx 1$ and $\widehat{f}^{(2)}_k[\widehat{T}]/\varphi^{(2)}_k \approx 1$.

B. ATD FOR SINGLE-CLASS SVM

While the ATD used for TC-SVM contains two types of data points (corresponding to two labels), the ATD used for SC-SVM contains only a single type of data point (corresponding to a single label). Naturally, the ATD used for SC-SVM can be extracted from the ATD used for SVM by removing one of the labels and keeping the other one.

In practical use cases, SC-SVM is related to the scenario of not having the CSI of E. Thus, it is reasonable to assume that the ATD used for SC-SVM only includes data points associated with the label (#0) (i.e., non-attack). Naturally, if we only have the perfect CSI of legitimate users, we can only simulate the virtual process of transmitting signals from the legitimate users to Alice and create (#0)-related data points.

TABLE 5. Artificial Training Data (ATD) Used for SC-SVM

MEAN	RATIO	Labels
$\widehat{f}^{(1)}_{k\|\mathcal{H}_0}[T_0]$	$\widehat{f}^{(2)}_{k\|\mathcal{H}_0}[T_0]$	(#0)
$\vdots$	$\vdots$	$\vdots$
$\widehat{f}^{(1)}_{k\|\mathcal{H}_0}[\widehat{T}]$	$\widehat{f}^{(2)}_{k\|\mathcal{H}_0}[\widehat{T}]$	(#0)

Table 5, which is constructed of the last $\widehat{T}$ rows of Table 4, shows the ATD used for SC-SVM. As expected, Table 5 does not include any data points associated with eavesdropping attacks.

C. ATD NORMALIZATION/WHITENING

Normally, SVM works with data within the range of [0, 1]. Thus, it is necessary for the AP to normalize all features in the training datasets. Normalizing (or whitening) makes all values in a feature column fall within [0, 1]. More specifically, if a certain feature column consists of values u_1, u_2, ..., and $u_{T_{tot}}$, then the AP will have to run the following algorithm:

- Step 1: Find $u_{min} = \min\{u_1, u_2, \ldots, u_{T_{tot}}\}$ and $u_{max} = \max\{u_1, u_2, \ldots, u_{T_{tot}}\}$.
- Step 2: Compute $u^{temp}_k = \dfrac{u_k - u_{min}}{u_{max} - u_{min}}$ where $k \in \{1, 2, \ldots, T_{tot}\}$. Assign the temporary value u^{temp}_k to u_k by letting $u_k = u^{temp}_k$.

Upon applying the above algorithm to all feature columns in the ATD, we can normalize the ATD before actually using the SVM classifiers. Naturally, if the training data (i.e., the ATD in this article) is normalized, then the testing data also must be normalized accordingly.

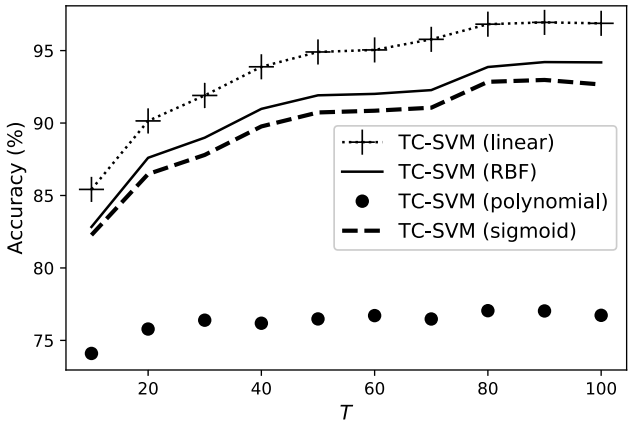

FIGURE 5. An illustration of four trained TC-SVM models that correspond to four different kernel functions (i.e., linear, RBF, polynomial (degree 2), and sigmoid kernels).

V. NUMERICAL RESULTS

In this section, we present several numerical examples for characterizing the proposed framework under specific settings. As for the channel's fading, we set $g_k = \sqrt{\beta_k} h_k$ and $g_{\rm E} = \sqrt{\beta_{\rm E}} h_{\rm E}$. Herein, β_k and $\beta_{\rm E}$ represent the path loss, while $h_k \sim \mathcal{CN}(0, 1)$ and $h_{\rm E} \sim \mathcal{CN}(0, 1)$ represent the Rayleigh fading.[3]

The complexity depends heavily on the choice of the loss function and the optimization method [36]. Using the big-O notation, it is experimentally confirmed that if the implementation is based on LIBSVM, the run-time of TC-SVM as well as SC-SVM is on the order of about $\mathcal{O}(\widehat{T}_{tot}^3)$, where $\widehat{T}_{tot}$ is the total number of training samples in the ATD [37]–[39]. Our experiments are carried out in Python and the execution of each TC-SVM/SC-SVM classifier takes around 0.09 seconds for training an ATD-based model and for labelling the data points in the testing data.

A. EXAMINING $\widehat{T}$, T AND T_0

Table 6 shows the accuracy of RBF-based TC-SVM (and SC-SVM) classifiers for different values of $\widehat{T}$ and T. Observe that the accuracy of 99% is achievable, provided that the parameters are appropriately chosen. Moreover, the results show that in the case of TC-SVM, the accuracy increases both with T as well as with $\widehat{T}$. By contrast, in the case of SC-SVM, there is little change in the accuracy. Explicitly, the change of $\widehat{T}$, as well as T does not affect much how SC-SVM learns the decision boundary. Additionally, although TC-SVM offers higher accuracy than SC-SVM, we have to note that a comparison between these two may be unfair due to the differences inherent in these two approaches (e.g., (3) uses the regularization parameter C, while (7) uses the parameter ν), and that SC-SVM requires only the CSI of the legitimate users.

[3]Although Rayleigh fading is used to model h_k and $h_{\rm E}$, other types of fading can also be used because the framework, presented in Sections III and IV, is not limited to any specific type of fading.

TABLE 6. The Kernel Function, Which is Mentioned in (5), is the RBF Kernel, i.e., $\mathbf{x}_s\mathbf{x} = \exp\{-\gamma||\mathbf{x}_s - \mathbf{x}||\}$. The SVM Parameters are $\gamma = 0.5$ and $C = 1$. The SC-SVM Parameters are $\nu = 0.02$, $\gamma = 0.5$. System Parameters: $K = 4$, $\mathcal{L} = 10$, $\rho_u = \rho_{\rm E} = 5$, $\beta_k = 1$, $\beta_{\rm E} = 1$, and $T_0 = 1$

$\widehat{T}$	T	The accuracy of algorithms (%)	
		TC-SVM	SC-SVM
200	10	76.86	99.37
	20	81.81	98.6
	50	89.23	97.78
2000	10	76.76	98.76
	20	81.87	98.21
	50	91.28	97.26
4000	10	77.38	99.39
	20	82.76	98.93
	50	91.44	98.31

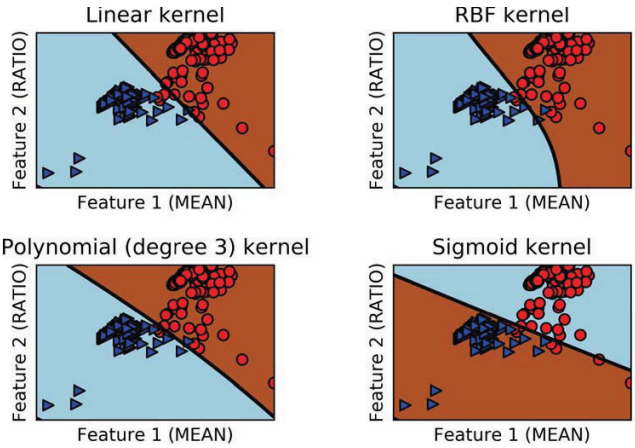

FIGURE 6. An illustration of 4 trained TC-SVM models that correspond to four different kernel functions (i.e., linear, RBF, polynomial (degree 3), and sigmoid kernels). The cyan area and ▷ markers relate to (#0), while the brown area and ◦ markers relate to (#1).

B. EXAMINING FOUR DIFFERENT KERNEL FUNCTIONS

Fig. 5 shows the accuracy of four different classifiers versus T for TC-SVM. These four classifiers are based on

- linear kernel: $\mathcal{K}(\mathbf{x}_s, \mathbf{x}) = \langle \mathbf{x}_s, \mathbf{x} \rangle$,
- RBF kernel: $\mathcal{K}(\mathbf{x}_s, \mathbf{x}) = \exp\{-\gamma \|\mathbf{x}_s - \mathbf{x}\|\}$,
- polynomial kernel: $\mathcal{K}(\mathbf{x}_s, \mathbf{x}) = (\gamma \langle \mathbf{x}_s, \mathbf{x} \rangle + r)^d$,
- sigmoid kernel: $\mathcal{K}(\mathbf{x}_s, \mathbf{x}) = \tanh(\gamma \langle \mathbf{x}_s, \mathbf{x} \rangle + r)$.

Herein, it should also be noted that we set r to 0, so that the RBF, polynomial and sigmoid kernels are fairly treated. The other parameters are as follows: $K = 4$, $\mathcal{L} = 10$, $\rho_u = 5$, $\rho_{\rm E} = 4$, $\beta_k = 1$, $\beta_{\rm E} = 1$, $\widehat{T} = 2000$, $T_0 = 5$, and $\gamma = 0.001$. Also in Fig. 5, the accuracy increases with T. For example, when $T \geq 80$, the accuracy exceeds 95% for the linear kernel. Among the four kernels considered, the linear kernel has the highest accuracy (e.g., over 95%), while the sigmoid kernel has the lowest accuracy (e.g., less than 80%). The accuracy of four different SC-SVM classifiers versus T indicate no accuracy improvements with T, hence it has been omitted. However, it should be noted that among the SC-SVM models, the sigmoid kernel has the highest accuracy (over

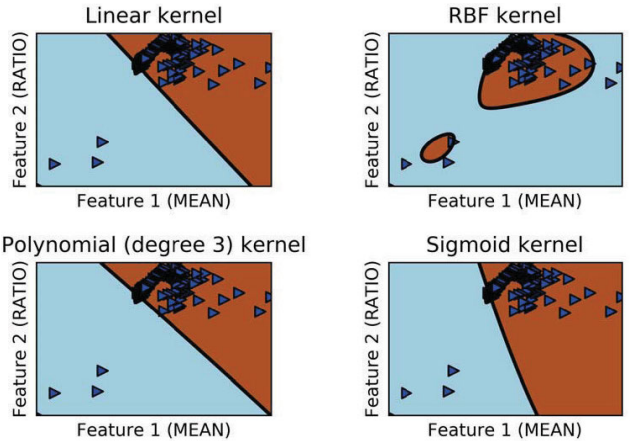

FIGURE 7. An illustration of 4 trained SC-SVM models that correspond to linear, RBF, polynomial (degree 3), and sigmoid kernels. ▷ markers are the ATD points associated with the label (#0). The brown area is the area that contains most of the ATD points ▷.

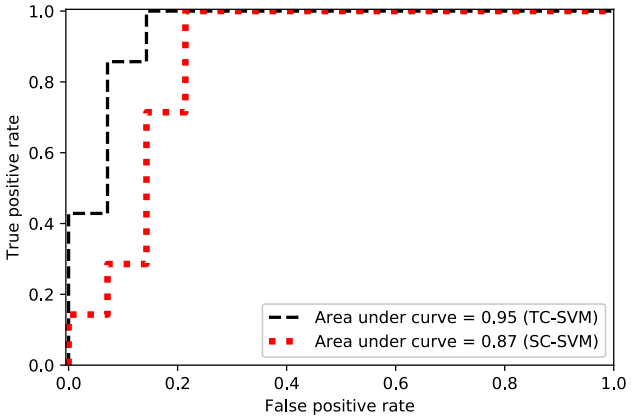

FIGURE 8. ROC curves and the areas under the corresponding ROC curves in two different regimes.

95%), while the RBF kernel has the lowest accuracy (i.e., around 90%). Additionally, there is no obvious difference between the polynomial kernel and the linear kernel (i.e., around 93%).

For characterizing the above-mentioned kernel functions in the case of TC-SVM, we provide an intuitive visual comparison in Fig 6. The total number of training samples is $T_{tot} = 2\widehat{T} = 400$, i.e., 200 samples with the label (#1) and 200 samples with the label (#0). Similarly, Fig. 7 illustrates four different SC-SVM models with $\widehat{T} = 200$ samples. Observe from Fig. 7 that there are only ATD points corresponding to the label (#0), because the SC-SVM models are trained on the data having a single class.

C. RECEIVER OPERATING CHARACTERISTIC CURVES AND SENSITIVITY-SPECIFICITY TRADE-OFF

According to [40]–[42], the true positive rate (or probability of detection) is defined as

$$TPR = \frac{\text{True positive (TP)}}{\text{Positive (Pos)}},$$

where TP represents the number of (#1)-labelled samples *correctly classified as (#1)*, and Pos is the total number of *actually* (#1)-labelled samples. By contrast, the false positive rate (or probability of false alarm) is defined as [42]

$$FPR = \frac{\text{False positive (FP)}}{\text{Negative (Neg)}},$$

where FP is the number of (#0)-labelled samples *incorrectly classified as (#1)*, and Neg is the total number of *actually* (#0)-labelled samples.

To plot the *TPR* against the *FPR*, we offer Fig. 8 that contains two receiver operating characteristic (ROC) curves. The kernel used for Fig. 8 is the RBF kernel. The other parameters are as follows: $K = 4$, $\mathcal{L} = 10$, $\rho_u = 5$, $\rho_{\mathrm{E}} = 2$, $\beta_k = 1$, $\beta_{\mathrm{E}} = 1$, $C = 1$, $\nu = 0.02$, $\gamma = 0.001$, $\widehat{T} = 1000$, $T = 20$, and $T_0 = 1$. The area under the ROC curve (AUC) is 0.95 in the case of TC-SVM and 0.87 in the case of SC-SVM. These two numbers reveal that TC-SVM may be better than SC-SVM, when taking into account both *TPR* and *FPR*. Since the *TPR* increases with the *FPR*, the careful choice of parameters are required to maintain a low *FPR* and an acceptable *TPR* (e.g. $FPR \leq 0.2$ and $TPR \geq 0.8$).

Apart from the *TPR* and *FPR*, there are two other similar rates referred to as false negative rate (*FNR*) and true negative rate (*TNR*), which are formulated as:

$$FNR = \frac{\text{False negative (FN)}}{\text{Positive (Pos)}} = 1 - TPR,$$

and

$$TNR = \frac{\text{True negative (TN)}}{\text{Negative (Neg)}} = 1 - FPR,$$

where FN is the number of (#1)-labelled samples *incorrectly identified as (#0)*, and TN is the number of (#0)-labelled samples *correctly identified as (#0)*. Note that *TPR* is also known as (a.k.a.) the *sensitivity* that quantifies how well a test can identify true positives, while *TNR* is a.k.a. the *specificity* that measures how well a test can identify true negatives. The trade-off between the sensitivity and the specificity is illustrated in Fig. 9. We can see from Fig. 9 that the sensitivity and specificity of the TC-SVM model are higher than those of the SC-SVM model. This confirms again that the TC-SVM model outperforms the SC-SVM model. Additionally, it is readily seen that the higher the sensitivity, the lower the specificity becomes. Explicitly, this means that neither the TC-SVM model nor the SC-SVM can improve the sensitivity and the specificity at the same time. In other words, if a model can improve its probability of detecting malicious samples associated with actual eavesdropping attacks, then its proportion of admitting benevolent samples associated with the situation of non-attack will go down.

D. EXAMINING THE IMPACT OF ρ_E AND γ

Fig. 10 shows the accuracy versus the ratio ρ_{E}/ρ_u in two sub-cases corresponding to $\gamma = \{0.001, 1\}$. Herein, we fix

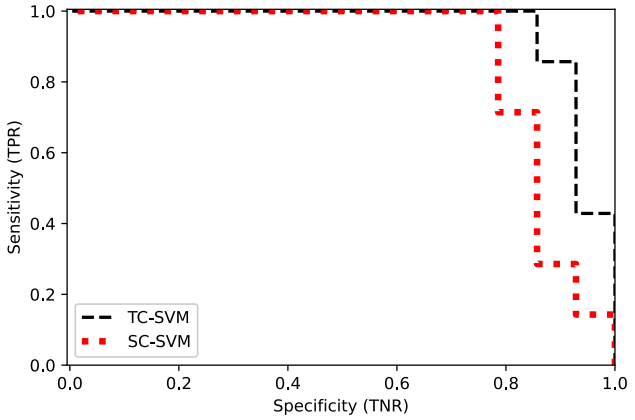

FIGURE 9. The trade-off between the sensitivity and the specificity in two different regimes.

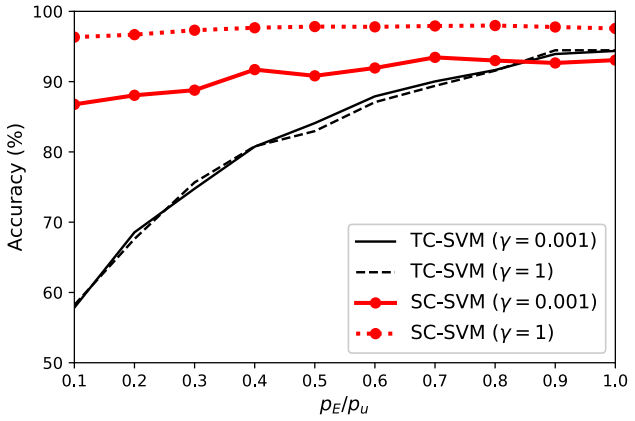

FIGURE 10. An illustration of the trained TC-SVM and SC-SVM models that correspond to the RBF kernel.

$\rho_u = 5$ and change ρ_{E}. The kernel used in Fig. 10 is the RBF kernel. The parameter C of TC-SVM is equal to $C = 1$, while the parameter ν of SC-SVM is equal to $\nu = 0.02$. The common system parameters are as follows: $K = 4$, $\mathcal{L} = 10$, $\beta_k = 1$, $\beta_{\mathrm{E}} = 1$, $(\widehat{T}, T, T_0) = (2000, 50, 15)$.

In general, the accuracy of the TC-SVM model increases from about 58% at $\rho_{\mathrm{E}} = 0.1\rho_u$ to about 92% at $\rho_{\mathrm{E}} = \rho_u$. This is because the TC-SVM model learns from the data belonging to two classes in which class (#1) is related to the eavesdropper's power. When ρ_{E} increases, the data points in class (#1) become more readily separable from the data points in class (#0), thereby yielding higher accuracy. Additionally, the change of γ has little impact on the accuracy of the TC-SVM model.

By contrast, the accuracy of the SC-SVM does not change significantly with ρ_{E}. This is because the SC-SVM model learns from the data having a single class, i.e., (#0). Thus, it is less sensitive to ρ_{E} that inherently affects only the class (#1). Furthermore, the change of γ significantly affects the accuracy of the SC-SVM model. For example, when $\gamma = 0.001$, the accuracy may exceed 97%, even when ρ_{E} is small. This means that γ is an influential parameter in allowing the SC-SVM model to learn the unique characteristics of the normal data points, thereby detecting the abnormal outliers that are related to the eavesdropper.

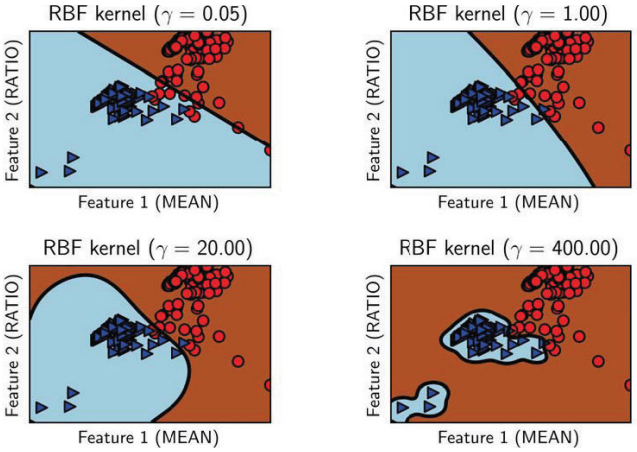

FIGURE 11. The four trained TC-SVM models correspond to four different values of γ in the case of the RBF kernel (with $C = 1$). The cyan area and $\triangleright$ markers relate to (#0), while the brown area and $\circ$ markers relate to (#1).

E. OVER-FITTING PROBLEMS: THE IMPACT OF γ

In Figure 11, we make a comparison among four different sub-cases corresponding to $\gamma = \{0.05, 1, 20, 400\}$. The total number of training samples is equal to $T_{tot} = 2\widehat{T} = 400$. Figure 11 illustrates the transformation of models, when γ increases. With $\gamma = 400$, we face an over-fitting problem due to the fact that the fourth model learns so well and becomes too detailed.

VI. CONCLUSION

We have considered a wireless communication system that requires the uplink phase for authentication. Eve has been assumed to impersonate a legitimate user. To identify the presence of Eve, we have introduced the ATD and employed TC-SVM/SC-SVM. The results have shown the following insights:

- i) Underlined the importance of formulating/defining features and converting the received signals into those features.
- ii) Characterized the impact of selecting kernel functions and TC-SVM/SC-SVM parameters.
- iii) Quantified the impact of both the training and testing dataset length on the accuracy. We have shown that when $\widehat{T}$ and T increase, the accuracy of the linear-kernel-based, RBF-kernel-based, and polynomial-kernel-based TC-SVM classifiers can be improved.
- iv) The impact of selecting γ in relation to the accuracy has been presented, especially in the case of SC-SVM. We have also shown the impact of selecting γ on the associated over-fitting problems.

The data obtained through our framework can also be fed into other machine learning algorithms, such as Gaussian mixture, random forest and isolation forest. However, the comparison of these algorithms in the context of PLS is still an open

problem to be tackled by future research. For future work, we plan to delve more into the data to explore hidden features that can further improve the performance of detection models. Explicitly, the family of data augmentation techniques holds the promise of making the data more robust for authentication purposes (e.g., see [43]).

ACKNOWLEDGMENT

This article was presented in part at the 2019 IEEE Conference on Communications and Network Security, Washington DC, USA.

REFERENCES

[1] T. M. Hoang, T. Q. Duong, and S. Lambotharan, "Secure wireless communication using support vector machines," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Jun. 2019, pp. 1–5.
[2] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," *Proc. IEEE*, vol. 104, no. 9, pp. 1727–1765, Sep. 2016.
[3] D. Kapetanovic, G. Zheng, and F. Rusek, "Physical layer security for massive MIMO: An overview on passive eavesdropping and active attacks," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 21–27, Jun. 2015.
[4] W. Wang, K. C. Teh, K. H. Li, and S. Luo, "On the impact of adaptive eavesdroppers in multi-antenna cellular networks," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 2, pp. 269–279, Feb. 2018.
[5] A. Al-Nahari, "Physical layer security using massive multiple-input and multiple-output: Passive and active eavesdroppers," *IET Commun.*, vol. 10, no. 1, pp. 50–56, Jan. 2016.
[6] Y. Wu, R. Schober, D. W. K. Ng, C. Xiao, and G. Caire, "Secure massive MIMO transmission with an active eavesdropper," *IEEE Trans. Inf. Theory*, vol. 62, no. 7, pp. 3880–3900, Jul. 2016.
[7] T. M. Hoang, H. Q. Ngo, T. Q. Duong, H. D. Tuan, and A. Marshall, "Cell-free massive MIMO networks: Optimal power control against active eavesdropping," *IEEE Trans. Commun.*, vol. 66, no. 10, pp. 4724–4737, Oct. 2018.
[8] L. Xiao, L. Greenstein, N. Mandayam, and W. Trappe, "Using the physical layer for wireless authentication in time-variant channels," *IEEE Trans. Wireless Commun.*, vol. 7, no. 7, pp. 2571–2579, Jul. 2008.
[9] W. Hou, X. Wang, J.-Y. Chouinard, and A. Refaey, "Physical layer authentication for mobile systems with time-varying carrier frequency offsets," *IEEE Trans. Commun.*, vol. 62, no. 5, pp. 1658–1667, May 2014.
[10] C. Pei, N. Zhang, X. S. Shen, and J. W. Mark, "Channel-based physical layer authentication," in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, pp. 4114–4119.
[11] L. Xiao, Y. Li, G. Han, G. Liu, and W. Zhuang, "PHY-layer spoofing detection with reinforcement learning in wireless networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 10037–10047, Dec. 2016.
[12] A. Weinand, M. Karrenbauer, R. Sattiraju, and H. Schotten, "Application of machine learning for channel based message authentication in mission critical machine type communication," in *Proc. 23rd Eur. Wireless*, Dresden, Germany, May 2017, pp. 342–346.
[13] N. Wang, T. Jiang, S. Lv, and L. Xiao, "Physical-layer authentication based on extreme learning machine," *IEEE Commun. Lett.*, vol. 21, no. 7, pp. 1557–1560, Jul. 2017.
[14] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, and Z. Han, "Detecting stealthy false data injection using machine learning in smart grid," *IEEE Syst. J.*, vol. 11, no. 3, pp. 1644–1652, Sep. 2017.
[15] X. Qiu, T. Jiang, S. Wu, and M. Hayes, "Physical layer authentication enhancement using a Gaussian mixture model," *IEEE Access*, vol. 6, pp. 53583–53592, 2018.
[16] B. Chatterjee, D. Das, S. Maity, and S. Sen, "RF-PUF: Enhancing IoT security through authentication of wireless nodes using *in-situ* machine learning," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 388–398, Feb. 2019.
[17] F. Pan, Z. Pang, H. Wen, M. Luvisotto, M. Xiao, R.-F. Liao, and J. Chen, "Threshold-free physical layer authentication based on machine learning for industrial wireless CPS," *IEEE Trans. Ind. Informat.*, vol. 15, no. 12, pp. 6481–6491, Dec. 2019.
[18] X. Lu, L. Xiao, T. Xu, Y. Zhao, Y. Tang, and W. Zhuang, "Reinforcement learning based PHY authentication for VANETs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3068–3079, Mar. 2020.
[19] H. Li, P. He, S. Wang, A. Rocha, X. Jiang, and A. C. Kot, "Learning generalized deep feature representation for face anti-spoofing," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 10, pp. 2639–2652, Oct. 2018.
[20] L. Wu, J. Yang, M. Zhou, Y. Chen, and Q. Wang, "LVID: A multimodal biometrics authentication system on smartphones," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1572–1585, 2020.
[21] H. Zhang, J. Liu, K. Li, H. Tan, and G. Wang, "Gait learning based authentication for intelligent things," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4450–4459, Apr. 2020.
[22] I. Guyon, S. R. Gunn, M. Nikravesh, and L. Zadeh, *Feature Extraction: Foundations and Applications*. New York, NY, USA: Springer-Verlag, 2006.
[23] H. Fang, X. Wang, and S. Tomasin, "Machine learning for intelligent authentication in 5G and beyond wireless networks," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 55–61, Oct. 2019.
[24] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag, 1995.
[25] N. Sapankevych and R. Sankar, "Time series prediction using support vector machines: A survey," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 24–38, May 2009.
[26] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, 1998.
[27] S. Abe, *Support Vector Machines for Pattern Classification*. New York, NY, USA: Springer-Verlag, 2005.
[28] F. Perez-Cruz and O. Bousquet, "Kernel methods and their potential use in signal processing," *IEEE Signal Process. Mag.*, vol. 23, no. 3, pp. 57–65, May 2004.
[29] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2002.
[30] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 3, no. 20. Cham, Switzerland: Springer, Sep. 1995, pp. 273–229.
[31] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer-Verlag, 2006.
[32] L. M. Manevitz and M. Yousef, "One-class SVMs for document classification," *J. Mach. Learn. Res.*, vol. 2, pp. 139–154, Dec. 2001.
[33] Y. Wang, J. Wong, and A. Miner, "Anomaly intrusion detection using one class SVM," in *Proc. IEEE 5th Annu. SMC Inf. Assurance Workshop*, West Point, NY, USA, Jun. 2004, pp. 358–364.
[34] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.
[35] Y. Wu, R. Schober, D. W. K. Ng, C. Xiao, and G. Caire, "Secure massive MIMO transmission in the presence of an active eavesdropper," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., Jun. 2015, pp. 1434–1440.
[36] O. Chapelle, "Training a support vector machine in the primal," *Neural Comput.*, vol. 19, no. 5, pp. 1155–1178, May 2007.
[37] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, Apr. 2011. [Online]. Available: https://www.csie.ntu.edu.tw/~cjlin/papers/libsvm
[38] A. Abdiansah and R. Wardoyo, "Time complexity analysis of support vector machines (SVM) in LibSVM," *Int. J. Comput. Appl.*, vol. 128, no. 3, pp. 0975–8887, 2015.
[39] G. L. Grinblat, L. C. Uzal, and P. M. Granitto, "Abrupt change detection with one-class time-adaptive support vector machines," *Expert Syst. Appl.*, vol. 40, no. 18, pp. 7242–7249, Dec. 2013.
[40] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2006, pp. 233–240.
[41] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016786550500303X
[42] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, "SVMs modeling for highly imbalanced classification," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 39, no. 1, pp. 281–288, Feb. 2009.
[43] R.-F. Liao, H. Wen, S. Chen, F. Xie, F. Pan, J. Tang, and H. Song, "Multiuser physical layer authentication in Internet of Things with data augmentation," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2077–2088, Mar. 2020.

TIEP M. HOANG (Member, IEEE) received the B.Eng. degree in electronics and electrical engineering from the HCMC University of Technology, Vietnam, in 2012, the M.Eng. degree in electronics and radio engineering from Kyung Hee University, South Korea, in 2014, and the Ph.D. degree from Queen's University Belfast, U.K., in 2019. He is currently a Research Fellow with the Southampton Wireless Group, University of Southampton, U.K. His current research interests include wireless security, convex optimization, and machine learning.

TRUNG Q. DUONG (Senior Member, IEEE) received the Ph.D. degree in telecommunications systems from the Blekinge Institute of Technology (BTH), Sweden, in 2012. He was a Lecturer (an Assistant Professor) from 2013 to 2017 and a Reader (an Associate Professor) from 2018 to July 2020 with Queen's University Belfast, U.K., where he has been a Full Professor since August 2020, is currently a Research Chair of the Royal Academy of Engineering, and also a Professor. His current research interests include the Internet of Things (IoT), wireless communications, molecular communications, and signal processing.

Dr. Duong received the Best Paper Award at the IEEE Vehicular Technology Conference (VTC-Spring), in 2013, IEEE International Conference on Communications (ICC) 2014, the IEEE Global Communications Conference (GLOBECOM) 2016, and the IEEE Digital Signal Processing Conference (DSP) 2017. He was a recipient of the prestigious Royal Academy of Engineering Research Fellowship, from 2016 to 2021, and the prestigious Newton Prize 2017. He currently serves as an Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON COMMUNICATIONS, and a Lead Senior Editor for the IEEE COMMUNICATIONS LETTERS.

HOANG DUONG TUAN (Member, IEEE) received the Diploma (Hons.) and Ph.D. degrees in applied mathematics from Odessa State University, Ukraine, in 1987 and 1991, respectively. He spent 9 academic years in Japan as an Assistant Professor with the Department of Electronic-Mechanical Engineering, Nagoya University, from 1994 to 1999, and then as an Associate Professor with the Department of Electrical and Computer Engineering, Toyota Technological Institute, Nagoya, from 1999 to 2003. He was a Professor with the School of Electrical Engineering and Telecommunications, University of New South Wales, from 2003 to 2011. He is currently a Professor with the School of Electrical and Data Engineering, University of Technology Sydney. His research interests include optimization, control, signal processing, wireless communication, and biomedical engineering for more than 25 years.

SANGARAPILLAI LAMBOTHARAN (Senior Member, IEEE) received the Ph.D. degree in signal processing from Imperial College London, U.K., in 1997. Until 1999, he was a Postdoctoral Research Associate with Imperial College London. From 1999 to 2002, he was with the Motorola Applied Research Group, U.K., and investigated various projects including physical link layer modeling and performance characterization of GPRS, EGPRS, and UTRAN. He was a Visiting Scientist with the Engineering and Theory Centre, Cornell University, USA, in 1996. He was with King's College London and Cardiff University as a Lecturer and a Senior Lecturer, respectively, from 2002 to 2007. He is currently a Professor of digital communications and the Head of the Signal Processing and Networks Research Group, Wolfson School Mechanical, Electrical and Manufacturing Engineering, Loughborough University, U.K. His current research interests include 5G networks, MIMO, blockchain, machine learning, and network security. He has authored more than 200 journal and conference articles in these areas. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING.

LAJOS HANZO (Fellow, IEEE) (http://www-mobile.ecs.soton.ac.uk,https://en.wikipedia.org/wiki/Lajos_Hanzo) received the D.Sc. degree, holds an honorary doctorate, from the Technical University of Budapest, in 2009, and also from The University of Edinburgh, in 2015. He has published more than 1900 contributions at IEEE Xplore and 18 Wiley-IEEE Press books. He has helped the fast-track career of 119 Ph.D. students. More than 40 of them are Professors at various stages of their careers in academia and many of them are leading scientists in the wireless industry. He is a Fellow of Academy of Engineering, of IET, and of EURASIP. He is a Foreign Member of the Hungarian Academy of Sciences and a former Editor-in-Chief of the IEEE Press. He has served as a Governor for both IEEE ComSoc and of VTS.