


RESEARCH

Open Access



Primary user emulation and jamming attack detection in cognitive radio via sparse coding

Haji M. Furqan^{1*} , Mehmet A. Aygül[†], Mahmoud Nazzal¹ and Hüseyin Arslan^{1,2}

*Correspondence:

hamadni@st.medipol.edu.tr

[†]Haji M. Furqan and Mehmet A. Aygül contributed equally to this work.

¹School of Engineering and Natural Sciences, Istanbul Medipol University, 34810 Istanbul, Turkey
Full list of author information is available at the end of the article

Abstract

Cognitive radio is an intelligent and adaptive radio that improves the utilization of the spectrum by its opportunistic sharing. However, it is inherently vulnerable to primary user emulation and jamming attacks that degrade the spectrum utilization. In this paper, an algorithm for the detection of primary user emulation and jamming attacks in cognitive radio is proposed. The proposed algorithm is based on the sparse coding of the compressed received signal over a channel-dependent dictionary. More specifically, the convergence patterns in sparse coding according to such a dictionary are used to distinguish between a spectrum hole, a legitimate primary user, and an emulator or a jammer. The process of decision-making is carried out as a machine learning-based classification operation. Extensive numerical experiments show the effectiveness of the proposed algorithm in detecting the aforementioned attacks with high success rates. This is validated in terms of the confusion matrix quality metric. Besides, the proposed algorithm is shown to be superior to energy detection-based machine learning techniques in terms of receiver operating characteristics curves and the areas under these curves.

Keywords: Cognitive radio, Jamming detection, Machine learning, Physical layer security, Primary user emulation detection, Residual components, Sparse coding, Authentication, Physical layer authentication

1 Introduction

Due to the rapid growth in wireless technology and services, the scarcity of the wireless spectrum has become a major problem [1]. To meet the requirements of future wireless networks and to alleviate this spectrum scarcity problem, cognitive radio (CR) is one of the most promising solutions. CR allows spectrum sharing between the primary users (PUs) and secondary users (SUs). More specifically, it enables SUs to opportunistically utilize empty spectrum bands without harming the PUs by following these steps: (i) determining whether the channel is occupied or not, (ii) choosing the best part of the spectrum based on their quality of service (QoS) requirements, (iii) coordinating with other users to access the spectrum, and (iv) leaving the channel whenever a PU starts to transmit its data [2].

Although CR is a promising solution to address the spectrum shortage problem, it is inherently vulnerable to both traditional and new security threats [3]. This is due to the wireless nature and unique characteristics of CR. Traditional security threats include eavesdropping, spoofing, and jamming attacks [4], while new security threats include spectrum sensing data falsification (SSDF) and primary user emulation attack (PUEA) [3, 5].

An eavesdropper tries to “hear” the secret communication between legitimate nodes while a spoofer can modify, intercept, and replace the messages between the legitimate parties. On the other hand, a jammer can generate intentional interference signals to degrade the quality of communication for both PUs and SUs. Thus, a jammer can also prevent an SU from efficiently utilizing the white spaces of the spectrum by causing false alarms regarding spectrum occupancy [4].

In an SSDF attack, an illegitimate node provides false sensing information to degrade the performance of the collaborative spectrum sensing approach, where collaborative approaches include the interaction of multiple CRs to improve the sensing performance in the fading environment. On the other hand, PUEA is based on emulating the characteristics of the PU transmission to deceive the SUs about spectrum occupancy. PUEA prevents them from utilizing the existing spectrum holes and can even cause interference to the PUs in some cases [6].

Popular PUEA types include malicious and selfish attacks. The malicious attackers objective is to degrade the CR performance by preventing them from opportunistic exploitation of spectrum. Particularly, a malicious attacker destroys the operations of the CR network. Thus, it can stop CRs from sensing and can also disengage the already used spectrum by them. On the other hand, a selfish attacker aims at exploiting the space of the spectrum by preventing other secondary users from using it. More specifically, it focuses on enhancing its consumption of the spectrum by degrading the overall fairness of the system.

The focus of this work is to detect false alarm about the spectrum occupancy that is caused by illegitimate nodes. An illegitimate node can transmit a signal similar to that of a PU, considered as a primary user emulator (PUE), or can send an unstructured signal, considered as a jamming attack. In the literature, several solutions are proposed for illegitimate node detection. For instance, the power level of the signal through the energy detection (ED) algorithm can decide on the source of the signal [7] using a pre-defined threshold. In [8], the authors presented a Markov random field-based belief propagation framework with ED for PUEA detection. Firstly, SUs employ the energy-based algorithm and calculate the belief values about the real source of the signal. Afterwards, the belief values are shared between different users. Finally, the average belief value is compared with the pre-defined threshold. If the average is less than the threshold, it is assumed that the signal source is fake; otherwise, the source of the signal is assumed to be real. These approaches are simple; however, they are shown to create high levels of false alarm rates. Cross-layer techniques are also effective for illegitimate node detection. In [9], the authors proposed a cross-layer approach for jamming attack and PUEA detection in CR networks by using information from physical layer spectrum sensing, statistical analysis of routing information, and prior knowledge about PUs. This technique is effective for detecting PUEA and jamming attack. However, there is an excessive overhead in analyzing and comparing information from physical and network layers.

The wireless channel and inherent physical characteristics of communication devices are also effective for illegitimate node detection [10–13]. For instance, wireless channel-based detection schemes are proposed in [10–12] for PUEA detection. These techniques are based on the fact that the channel between different transmitter-receiver pairs is different due to its spatial decorrelation nature. In [13], the inherent physical layer features of devices based on hardware impairments are exploited for PUEA detection. Nevertheless, these techniques require excessive software and hardware overheads for their implementation.

Localization-based detection is also popular for PUEA detection. The basic idea is to infer the position of the signal's source by using the received signal and compare it with a database of pre-known locations of legitimate PUs. However, database management is not applicable in all scenarios [14, 15]. Similarly, the authors in [16] used the time difference of arrival-based position estimation approach for PUEA detection. However, this requires a strict synchronization between the receiver and the transmitter.

Machine learning (ML)-based solutions also received considerable attention for CR security. In [17], an anomaly detection framework for CR networks based on the characteristics of radio propagation is proposed. However, it does not consider specific attacks and is designed only for the detection of general anomalies. In [18], the authors proposed a technique based on support vector data description (SVDD) and zoom fast Fourier transform (zoom FFT). In the first step, the pilot and symbol rate are estimated using zoom FFT. Afterwards, a boundary around the PU objects is constructed using the SVDD classifier which is used to distinguish between PU and PUE. However, this method does not perform well in low signal-to-noise ratio (SNR) operating conditions. Furthermore, the method fails when the PUE is extremely intelligent (the only information unknown by the PUE is the channel). In [19], the authors proposed an ML-based algorithm for PUEA detection that exploits the signal strength and boundaries around the position of PU for the correct detection. This method is good in terms of complexity, but it suffers from performance degradation.

Recently, compressive sensing (CS)-based approaches were applied in spectrum sensing where CS offers several benefits. For example, it can alleviate the need for high sampling rate analog-to-digital converters [20–22]. This results in a reduction of the overall complexity, energy consumption, and memory requirements. Following its success in various application areas [20], CS has been applied to the problem of PUEA detection. Works along this line include PUEA detection based on CS and received signal strength [23]. This approach needs multiple sensors throughout the network. Hence, it increases the overall complexity. Another example considers exploiting belief propagation and CS for PUEA detection [24]. However, this requires a centralized node for its implementation. In [25], the authors proposed an algorithm for jamming attack detection in wide-band CR. In the first step, CS is performed to estimate a wide-band spectrum. Afterwards, an ED algorithm is applied to identify the occupied spectrum sub-bands. Lastly, waveform parameters of the sub-bands are compared with the known user database to determine the jamming attack. However, this method also requires database management.

In this paper, we propose an algorithm for PUEA and jamming attack detection corresponding to the narrow-band spectrum using the convergence patterns of the sparse coding over channel-dependent sampled dictionary. This convergence is characterized by

the sparse coding residual signal energy decay rates. The proposed algorithm does not require a centralized node or strict synchronization between transceiver ends. Moreover, it does not require information from multiple sensors for the implementation. Furthermore, it eliminates the need for estimating the sparse coding error tolerance or the sparsity level, as typically required in CS-based approaches. The reason is that the sparse recovery in the proposed algorithm is just used for energy convergence rate revelation rather than accurate signal reconstruction. The main contributions of this paper are as follows:

- First, the decaying pattern of sparse coding is used for PUEA detection. This is achieved by exploiting the convergence patterns of the sparse coding over a PU channel-dependent dictionary. In this context, these patterns guide on identifying a spectrum hole, a PU, and a PUE through ML approaches.
- Second, jamming attack detection is also performed based on the decay pattern of sparse coding. Here, the idea is that the noise and jamming signals are not compressible because they are not structured. So, residual energy decay patterns with a channel-dependent dictionary along with the non-compressive nature of jamming signals are used for efficient jamming attack detection via ML classification.

The rest of this paper is organized as follows. Preliminary information and the system model are presented in Section 2. Section 3 provides the proposed algorithm, while the complexity analysis is presented in Section 4. Section 5 presents the simulation results and discussions. Finally, the paper is concluded in Section 6.

Notation: Uppercase bold-faced, lowercase bold-faced, and lowercase plain letters represent matrices, vectors, and scalars, respectively. The symbols $\|\cdot\|_0$ and $\|\cdot\|_2$ denote the number of nonzero elements and the 2-norm of a vector, respectively. The $\langle \cdot, \cdot \rangle$, \dagger , and \mathbb{C} symbols represent inner product, Moore-Penrose pseudoinverse, and complex number field.

2 Preliminaries and system model

This section reviews background information related to CS, sparse recovery, and ML approaches.

2.1 Compressive sensing and sparse recovery

Using a random sensing matrix, CS merges data measurement and compression into a unified operation. CS applies to compressible signals, i.e., either the explicitly sparse signals or the ones admitting sparsity in a certain domain [26].

Let us assume a signal vector $\mathbf{y} \in \mathbb{C}^N$. A compressed version of \mathbf{y} can be obtained by applying a measurement matrix $\Phi \in \mathbb{C}^{M \times N}$ as $\mathbf{y}_c = \Phi \mathbf{y}$, where $M \ll N$. Hence, a reduction in dimensionality from N -to- M is achieved. A high-dimensional version of the original signal can be reconstructed from this low dimensional measurement via sparse recovery [26].

Generally speaking, let us assume that a signal \mathbf{y} admits sparse coding over a dictionary ($\mathbf{D} \in \mathbb{C}^{N \times K}$). The signal can be represented in terms of \mathbf{D} as $\mathbf{y} \approx \mathbf{D}\mathbf{w}$, where $\mathbf{w} \in \mathbb{C}^K$ is a sparse coefficient vector. The calculation of \mathbf{w} can be cast as follows.

$$\operatorname{argmin}_{\mathbf{w}} \|\mathbf{y} - \mathbf{D}\mathbf{w}\|_2^2 \text{ s.t. } \|\mathbf{w}\|_0 < S, \tag{1}$$

where S denotes the sparsity level of the signal. Sparse recovery is an NP-hard problem. However, sparse recovery methods offer efficient approximate solutions. As shown in (1), the ℓ_0 pseudo-norm is principally used to exactly quantify the sparsity level. However, its minimization is mathematically intractable and highly complex. Therefore, there exist only approximate solutions to ℓ_0 minimization, such as the matching pursuit and orthogonal matching pursuit (OMP) approaches. Alternatively, this problem can be overcome by relaxing the ℓ_0 norm minimization condition to minimizing the ℓ_1 norm which is a loose bound on sparsity. Still, ℓ_1 minimization is convex and accepts linear programming. Thus, replacing ℓ_0 minimization with ℓ_1 minimization offers a significant reduction to the computational complexity of sparse coding. However, ℓ_1 minimization requires information about the noise level of the signal being recovered. Thus, in this work, we adopt approximate ℓ_0 minimization through the OMP algorithm.¹

The intrinsic sparsity of the signal can be revealed by a dictionary. This dictionary can be formed of fixed basis functions such as Fourier basis, Gabor functions, wavelets, and contourlets. Alternatively, it can be generated as a learned dictionary. In this setting, a dictionary is obtained by training over training data signals $Y \in \mathbb{C}^{N \times L}$ [28]. This dictionary learning process can be formulated as

$$\operatorname{argmin}_{\mathbf{w}, \mathbf{D}} \|\mathbf{w}\|_0 \text{ s.t. } \|\mathbf{Y}_i - \mathbf{D}\mathbf{w}_i\|_2^2 < \epsilon \quad \forall i, \tag{2}$$

where ϵ represents error tolerance. Since the problem is non-tractable and non-convex, most of the dictionary learning algorithms perform the learning by iteratively alternating between a sparse representation stage and a dictionary update stage. As an example, the K-SVD algorithm [28] is one of the widely used algorithms for the dictionary learning process.

The abovementioned dictionary learning is a computationally demanding process. Therefore, developing efficient alternatives to the classical dictionary learning approach is needed for CR-related applications [21]. In this context, the use of sampled dictionaries is an efficient alternative. One can obtain a sampled dictionary by picking a set of randomly selected data vectors that serve for the sparse coding without the need for applying an expensive learning process. Thus, this offers a compromise in terms of computational complexity at a tolerable loss in the representational power of the dictionary. In [22], the use of sampled dictionaries is justified by their usage to represent data points in a specific class, which have a general similarity. Similarly, sampled dictionaries are used in this work to represent signals.

2.2 Residual components in pursuit sparse coding

A widely used sparse representation algorithm is OMP. This algorithm is based on iteratively obtaining the coefficients in a sparse coefficient vector (\mathbf{w}). Particularly, each iteration identifies the location and adjusts the value of a nonzero element in \mathbf{w} . This is achieved by selecting one atom (column) from a dictionary \mathbf{D} and adjusting its respective weight.

To implement the above-explained atom selection and coefficient update processes, algorithms such as OMP define a so-called residual signal \mathbf{r} . Conceptually, \mathbf{r} represents signal portions that have not yet been represented by the selected dictionary atoms.

¹The proposed algorithm is not limited to OMP, and it can be implemented with any sparse recovery algorithm [27]. We prefer to use the OMP algorithm since it is computationally efficient and simple.

Hence, sparse coding initializes \mathbf{r} with the signal itself, as $\mathbf{r} \leftarrow \mathbf{x}$. In the first iteration, the sparse representation algorithm loops through all dictionary atoms and selects the one most similar to the current residual \mathbf{r} . Once this atom is selected, the corresponding weight is calculated. To this end, the next residual is calculated by subtracting the resultant one-atom sparse approximation from the original residual. Then, the residual is considered as a new signal for which another dictionary atom is selected and another coefficient is calculated and the process continues until a certain halting condition is met.

The interesting point to consider in the above-explained sparse coding approach is that the energy of the residual components should dramatically decrease as sparse coding progresses. Intuitively, this is because more atoms are selected, and thus, more signal portions are excluded from the residual.

2.3 Machine learning for classification

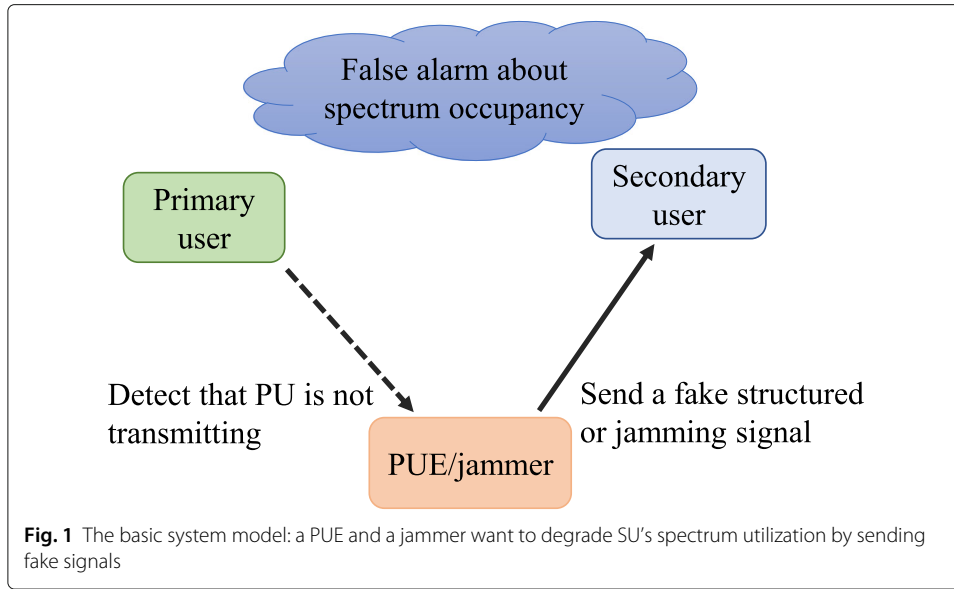
The successful works of the ML algorithms in many application areas such as computer vision, fingerprint identification, image processing, and speech recognition led these algorithms to become appealing for the area of wireless communication [29]. These ML algorithms are categorized under three categories called supervised, unsupervised, and reinforcement learning. Supervised learning-based ML algorithms are widely used for classification problems when the number of present classes is known and the information of the classes that samples belong to in the training stage is available.

Amongst many supervised learning-based algorithms, the feed-forward neural network has received growing interest in classification problems since it can recognize classes accurately and quickly [30]. This network can be used with a single-layer and multi-layer. Although single-layer algorithms are computationally good, these algorithms can only be used for simple problems. Alternatively, the multi-layer-based algorithms that include the usage of one or more hidden layers are used. Even though these algorithms increase computational complexity, they are able to solve more complex problems. Besides the effect of the extra layers, the number of neurons that are used in hidden layers is also effective on the accuracy and complexity performances. Therefore, it is quite significant to set these hyper-parameters optimally. Moreover, the complexity and accuracy performances can be increased by feature extraction (with the domain knowledge). Along this line, CS is used to extract features in this work with the aim of increasing the performance of the ML.

2.4 System model

The system model used is intended to characterize the existence of legitimate and illegitimate source nodes. Thus, it consists of a PU node, an SU node, and an illegitimate node as presented in Fig. 1. In this setting, an SU node opportunistically exploits the spectrum in the presence of an illegitimate node that can launch either PUEA or jamming attack. A jammer transmits a random signal, while a PU node and a PUE transmit structured signals that mimic the legitimate PUs.

We can represent the transmitted signal as $\mathbf{x} = \mathbf{A}\mathbf{s}$, where \mathbf{A} is a coefficient matrix with a size of $N \times N$. Each component is denoted by $a_{i,j}$ with $i, j = 1, \dots, N$, and $\mathbf{s} = [s_1(t), \dots, s_N(t)]^T$ represents the transmitted data vector. Any coordinate of \mathbf{s} is given as $s_i(t) = \sum_{k=-\infty}^{\infty} d_k u(t - kT_s) e^{j2\pi f_{c,o} t}$, where T_s is the symbol duration, $f_{c,o}$ represents the center frequency, d represents digitally modulated data symbols, $u(t)$ represents the pulse shaping filter, and $o = 1, 2, \dots, N$.



The signal at the receiver sent by any node can be written as

$$y = \mathbf{h}x + \mathbf{n}, \tag{3}$$

where \mathbf{h} is a multipath Rayleigh fading channel between any transmitter-receiver pair and \mathbf{n} is additive white Gaussian noise. Due to the spatial decorrelation concept, the channel between different transmitter-receiver pairs is assumed to be different [31].

3 The proposed algorithm for PUEA and jamming attack detection

The objective of this work is to differentiate between the following hypotheses:

$$y = \begin{cases} \mathbf{n} & \mathcal{H}_0 : \text{there is no PU,} \\ \mathbf{h}_{PU}\mathbf{x}_s + \mathbf{n} & \mathcal{H}_1 : \text{a PU is present,} \\ \mathbf{h}_i\mathbf{x}_s + \mathbf{n} & \mathcal{H}_2 : \text{a PUE is present,} \\ \mathbf{h}_i\mathbf{x}_n + \mathbf{n} & \mathcal{H}_3 : \text{a jammer is present,} \end{cases} \tag{4}$$

where \mathbf{n} is additive white Gaussian noise and \mathbf{y} is the received signal. Besides, \mathbf{h}_{PU} denotes the channel corresponding to the legitimate PU, \mathbf{h}_i is the channel corresponding to PUE or jammer, \mathbf{x}_n represents the (unstructured) jamming signal, and \mathbf{x}_s is a structured signal. In this work, two goals are set. The first is to detect PUEA, i.e., to differentiate between the \mathcal{H}_0 , \mathcal{H}_1 , and \mathcal{H}_2 hypotheses. The second goal is to detect jamming attacks, i.e., to differentiate between \mathcal{H}_0 , \mathcal{H}_1 , and \mathcal{H}_3 .

To meet the abovementioned goals, a compressed version of the received signal is observed by the CS algorithm and its sparse coding is calculated with respect to a PU channel-depended dictionary \mathbf{D}_{PU} . As detailed in Section 2, sparse coding iteratively minimizes the energy of a residual ($\|\mathbf{r}\|_2$). For each iteration, we calculate the value of $\|\mathbf{r}\|_2$. Then, we quantify the rate of its decay using the gradient operator ($|\mathbf{G}|$). It is noted that the speed of this decay depends on the harmony between the received signal and the dictionary.

The convergence profile of this residual or gradient versus iteration can be used to distinguish between the aforementioned hypotheses. The idea behind this approach is that the unstructured signals (noise and jamming) are not compressible, while structured signals are compressible. Hence, different signals have different $\|r\|_2$ and $|G|$ profiles that help to distinguish between different hypotheses. Following the same logic, different signals have different patterns based on the similarity between the dictionary atoms and signals. In other words, residual energy patterns show how much dictionary atoms can guarantee accurate and sparse representation for signals that can also help in distinguishing between various hypotheses. Intuitively speaking, a signal that is compressible in the given dictionary has a faster decay speed compared to other signals. Thus, if the dictionary is channel-dependent, it will also affect the pattern corresponding to $\|r\|_2$ and $|G|$, which can be used also to differentiate between different hypotheses.

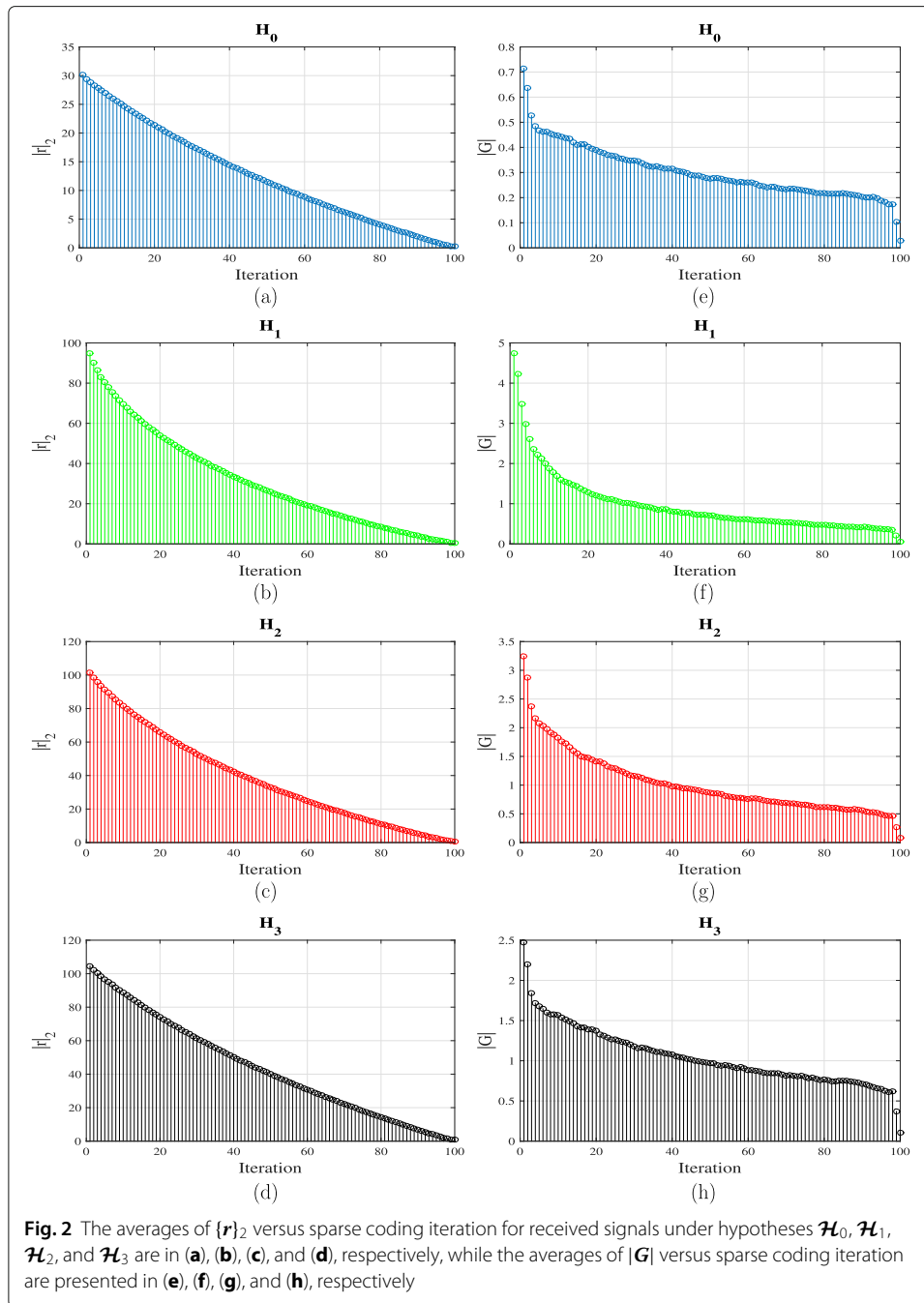
To this end, we analyze the usefulness of $\|r\|_2$ and $|G|$ in distinguishing between the aforementioned hypotheses in (4) with the following test. We use a test set of 10^3 quadruplets of synthetically generated received signals (y) that correspond to the hypotheses $\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2$, and \mathcal{H}_3 , respectively. In other words, one signal is mere noise, the other one is the signal received from the legitimate PU, the third one is a PUE signal that mimics the PU signal, and the fourth one is an unstructured jamming signal. These signals are generated as described in Section 5.

For each quadruplet, we calculate a PU-dependent dictionary (D_{PU}) based on the known PU channel (h_{PU}). In this work, a channel-dependent dictionary is obtained by convolving a set of randomly selected data (X) with the channel corresponding to the legitimate PU. Formally stated, $D_{PU} = h_{PU} * X$, where $*$ denotes convolution. Afterwards, we perform an iterative sparse coding operation on a compressed version of each signal in the quadruplet with D_{PU} while calculating $\|r\|_2$. Next, we calculate the gradient of each residual vector as $|G|$.

The average values of $\|r\|_2$ and $|G|$ in the above-explained test are presented in Fig. 2. In view of this figure, it is seen that one can differentiate between the four hypotheses based on $|G|$ and $\|r\|_2$ using ML approaches. For example, the gradient of \mathcal{H}_1 has faster decay as compared to $\mathcal{H}_0, \mathcal{H}_2$, and \mathcal{H}_3 as presented in Figs. 2f–h, respectively. The reason for exhibiting a faster decay is that the received signal in \mathcal{H}_1 (corresponding to PU) is the only one compressible in the given dictionary.

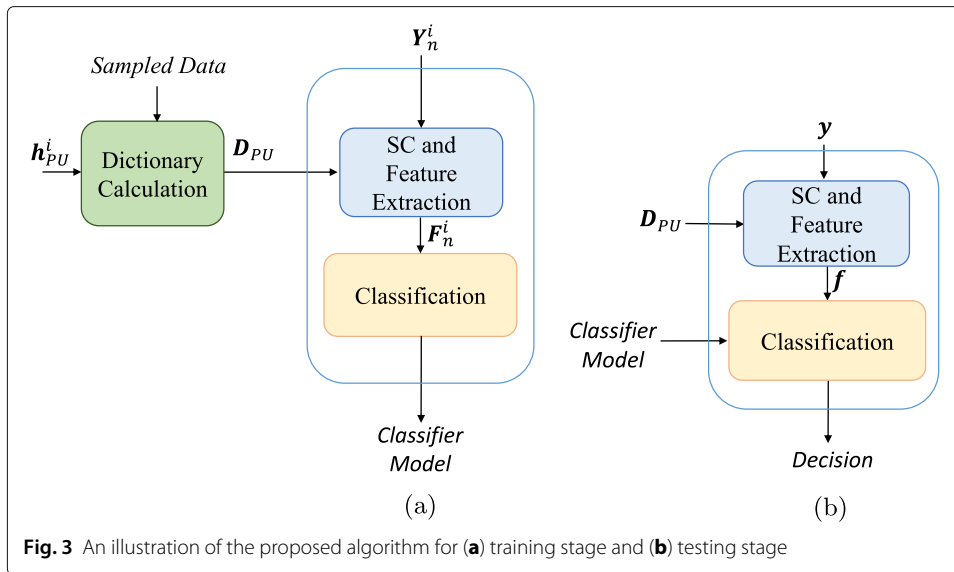
Based on the above discussion, we present the proposed algorithm. It is divided into two main stages. First, is a classifier training stage, where one uses a comprehensive set of training signals. We can either concatenate $\|r\|_2$ and its absolute gradient $|G|$ into a unified feature vector or use them separately as classification features. These features are used to make training data sets f_0^i, f_1^i, f_2^i , and f_3^i according to the hypotheses explained in (4).

For the case of PUEA detection, the training set contains f_0^i, f_1^i , and f_2^i corresponding to the hypotheses $\mathcal{H}_0, \mathcal{H}_1$, and \mathcal{H}_2 , respectively. On the other hand, for the case of jammer detection, the training set contains f_0^i, f_1^i , and f_3^i corresponding to the hypotheses $\mathcal{H}_0, \mathcal{H}_1$, and \mathcal{H}_3 , respectively. Afterwards, these training vectors, along with their class labels are fed to the ML training stage, where a classifier model is trained accordingly. The workflow of the training set preparation stage is pictorially described in Fig. 3a. In this figure, Y_n^i represents the set of compressed received signals y_0^i, y_1^i , and y_2^i for the case of



PUEA detection or \mathbf{y}_0^i , \mathbf{y}_1^i , and \mathbf{y}_3^i for the case of jamming attack detection. Similarly, \mathbf{F}_n^i represents the set of training vectors.

After classifier training, the testing stage represents the run-time operation of the proposed algorithm. This process is explained in Fig. 3b. For each incoming test signal, \mathbf{y} , sparse coding is performed over \mathbf{D}_{PU} and feature vector \mathbf{f} is obtained. Afterwards, \mathbf{f} is fed into the learned classifier. Finally, this classifier will decide on the hypothesis corresponding to the current signal of interest. An analysis of this idea is provided in the Appendix.



4 Complexity analysis

In this section, we roughly quantify the computational complexity of the proposed algorithm. This complexity is primarily required by sparse coding and ML.

The OMP computational complexity at the k th iteration is $\mathcal{O}(MK + KS + KS^2 + S^3)$ while the overall complexity is $\mathcal{O}(MKS + KS^2 + KS^3 + S^4)$, where S represents the sparsity level [32]. Thus, the overall computational complexity of sparse coding with a sparsity level of M is $\mathcal{O}(KM^2 + KM^2 + KM^3 + M^4)$. This can be simplified as $\mathcal{O}(2KM^2 + KM^3 + M^4)$. Note that sparse coding is used during both the training and the testing phase in the proposed algorithm.

The computational complexity of ML is divided into two main stages which are training and testing. The computational complexity of two-layer neural network per sample is $\mathcal{O}(e(lk + ml))$ for training stage, where e denotes the number of epochs, while k , l , and m represent the number of neurons at the input, hidden, and output layers, respectively. The total complexity of training stage is $\mathcal{O}(ep(lk + ml))$ for p number of samples. Moreover, the computational complexity of training per sample is roughly double as compared to the complexity of testing per sample [33]. It is worth to note that $k = 2M$, since $\|r\|_2$ and $|G|$ are concatenated into a unified feature vector in the simulations.

5 Results and discussion

This section presents numerical experiments to assess the performance of the proposed algorithm comparing it with the ED approach.

5.1 Parameter setting

The simulations are conducted with different modulation settings based on the system model specifications presented in Section 2. The modulation types used include quadrature amplitude modulation (QAM), pulse amplitude modulation (PAM), frequency-shift keying (FSK), and phase-shift keying (PSK). Moreover, the proposed algorithm uses a 100×400 dictionary. For each received signal, a channel realization [34] is generated for the PU and uncorrelated channel realizations are generated for illegitimate node based on

Table 1 Synthetic received signal simulation parameters

Parameter	Value
Channel model	Rayleigh
No. of taps	7
Channel delay unit	Sample period
Signal length	100
Oversampling rate	10
Pulse shaping	Square-root-raised-cos.
Raised cos. symbol span	50
Raised cos. roll-off factor	0.2
Correlation factor	0.9

channel decorrelation concept [31]. The assumed model of \mathbf{h}_{PU} is $\mathbf{h}_{PU} = \rho \mathbf{h} + (1 - \rho)$, where ρ is the correlation factor and \mathbf{h} is Rayleigh fading channel [35]. The details of the simulation parameters are presented in Table 1.

We use a standard two-layer feed-forward network [30] for that consists of a hidden layer and an output layer with sigmoid functions. The number of hidden neurons is set to 64 while the number of output neurons is set to the number of elements in the target vector which is 3 (corresponding to the number of classes in PUEA or jamming attack detection). For the case of PUEA detection, the vectors $\mathbf{f}_0^i, \mathbf{f}_1^i$, and \mathbf{f}_2^i are used for training. For jamming attack detection, $\mathbf{f}_0^i, \mathbf{f}_1^i$, and \mathbf{f}_3^i are used as input vectors. Energy decay rate

Table 2 Confusion matrices for PUEA detection

M=30								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
18.2	31.5	50.3	98.4	1.6	0.0	100.0	0.0	0.0
2.5	44.4	53.1	2.6	61.1	36.3	7.0	63.5	29.5
32.6	26.8	40.6	0.6	50.4	49.0	3.4	43.1	53.5
M=50								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
36.0	29.4	34.6	99.4	0.6	0.0	100.0	0.0	0.0
1.4	37.4	61.2	2.9	71.2	25.9	7.2	71.6	21.2
35.4	16.3	48.3	0.3	52.8	46.9	5.9	45.3	48.8
M=70								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
57.6	19.1	23.3	99.9	0.1	0.0	100.0	0.0	0.0
1.5	70.9	27.6	4.3	57.0	38.7	8.1	69.2	22.7
33.8	15.9	50.3	0.4	43.8	55.8	10.7	34.9	54.4
M=100								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
88.9	6.4	4.7	100.0	0.0	0.0	100.0	0.0	0.0
1.4	81.2	17.4	1.0	53.0	46.0	4.0	79.4	16.6
32.2	6.7	61.1	0.3	17.0	82.7	3.2	35.0	61.8

and gradient vectors $\|r\|_2$ and $|G|$ are used as feature vectors. Here, the dimension of both $\|r\|_2$ and $|G|$ is $1 \times M$. Therefore, the feature vector dimension is $1 \times 2M$.

It is noted that we take 4000 samples from each class in the training stage for all cases and 1000 samples from each class in the testing stage for each of the SNR values. Also, the neural network is trained over the SNR values ranging between -5 dB and 15 dB with a step size of 5 dB.

5.2 Performance analysis

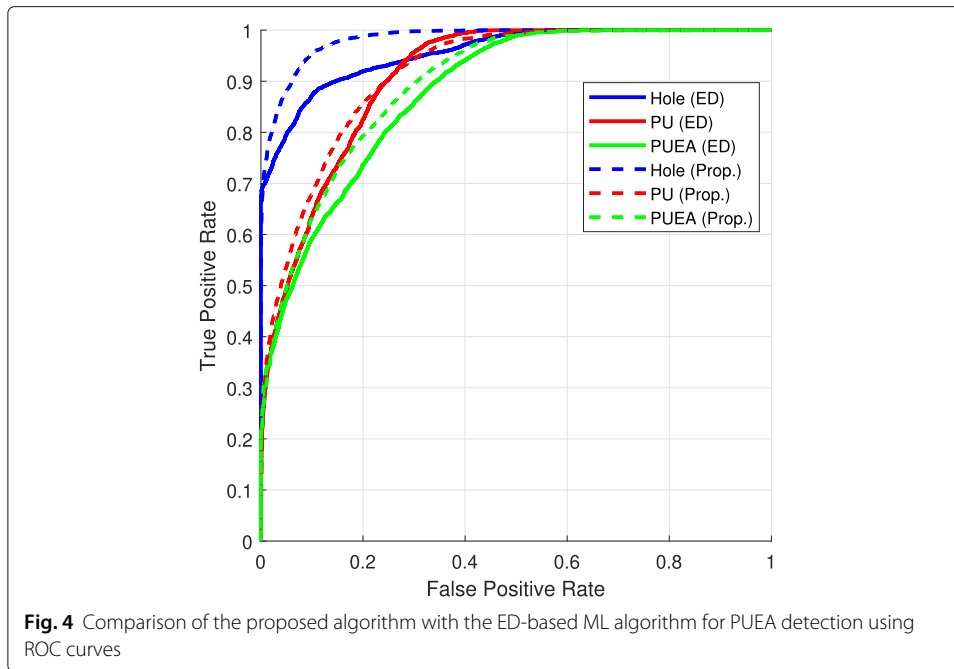
This section presents the performance analysis of the proposed algorithm in terms of confusion matrices, receiver operating characteristics (ROC) curves, and area under ROC (AUROC) curves. For the jamming detection scenario, it is assumed that the illegitimate node broadcasts non-structured signals. On the other hand, it is assumed that PUE signal's parameters are identical to that of PU signal.

To examine the performance of the classification, confusion matrices are often used. They present the number of both correctly and incorrectly classified observations. Thus, diagonal elements present the number of those observations correctly classified while off-diagonal elements indicate the number of incorrectly classified observations.

Table 2 presents the confusion matrices for the case of PUEA detection for different M and SNR values, where M is the number of samples in the compressed received signal. It is observed from Table 2 that the overall performance of the proposed algorithm is

Table 3 Confusion matrices for jamming attack detection

M=30								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
12.5	25	62.5	98.4	1.6	0	100	0	0
0.9	44.6	54.5	2.3	54.8	42.9	3.5	58	38.5
27.3	25.6	47.1	0	27.9	72.1	1.3	26.2	72.5
M=50								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
24.5	20.9	54.6	99.5	0.5	0	100	0	0
0.9	51.9	47.2	2.4	65.3	32.3	4.6	69.7	25.7
42.7	14.5	42.8	0	23.5	76.5	2.6	20	77.4
M=70								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
29.8	15	55.2	99.5	0.5	0	100	0	0
0.2	63.3	36.5	2.1	65.6	32.3	4.3	74.1	21.6
37.6	14.3	48.1	0	18.7	81.3	2.6	16.4	81
M=100								
0 dB			5 dB			10 dB		
\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2	\mathcal{H}_0	\mathcal{H}_1	\mathcal{H}_2
58.2	3	38.8	100	0	0	100	0	0
0.3	90.6	9.1	0.7	88.8	10.5	1.1	93.8	5.1
37	3.7	59.3	0	10.7	89.3	0.2	4.9	94.9



satisfactory for PUEA detection, especially at high SNR. Besides, the performance also improves with the increase in the values of M . Table 3 presents the confusion matrices for the case of jamming detection for different M and SNR values. It is seen from the table that the classification accuracy based on the proposed algorithm improves with the increase in M and SNR similar to PUEA case.

It is also observed from Tables 2 and 3 that the performance of the proposed jammer detection outperforms PUEA detection. This is because the jammer detection benefits

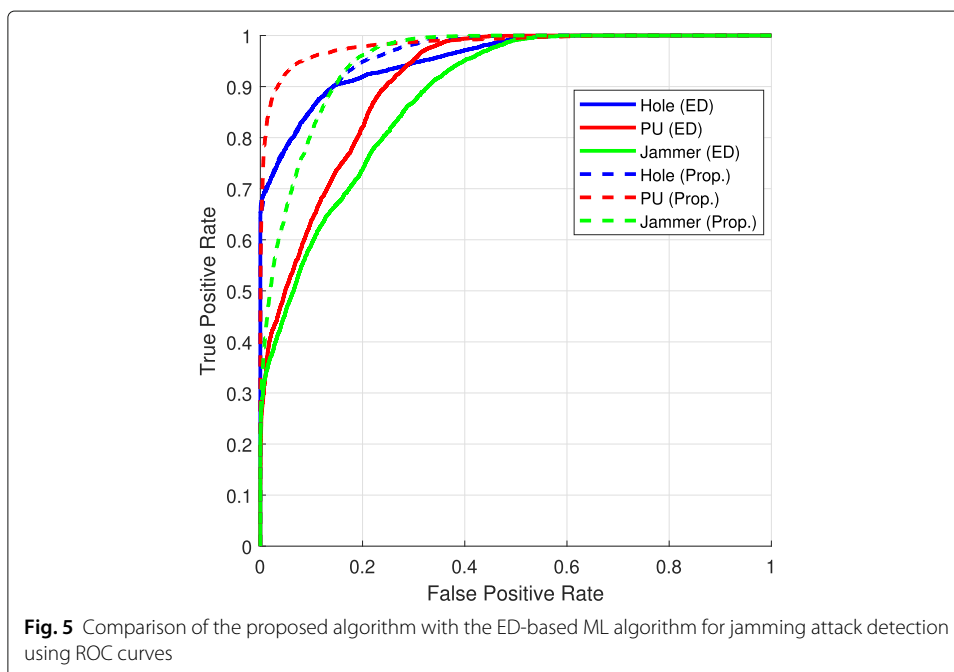


Table 4 AUROC values for PUEA

	PU	Hole	PUE
ED-based	0.9089	0.9560	0.8719
Proposed	0.9152	0.9828	0.8943

from both the non-compressive nature of the jamming signal and the channel-dependent dictionary while the PUEA detection benefits only from the channel-dependent dictionary.

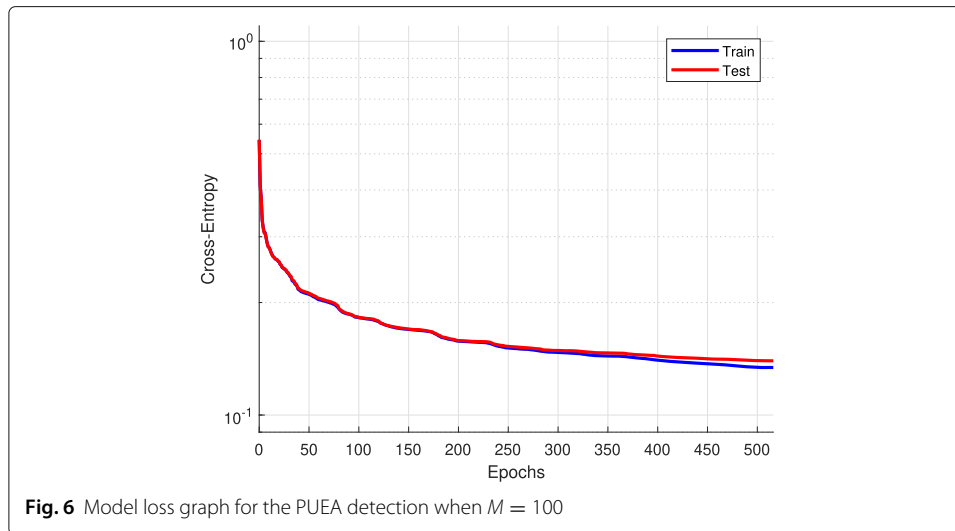
In classification, if a signal belongs class i and is correctly classified in to belong to the same class, then it is said to be as true positive (TP). If it is wrongly classified to belong to a different class j , then it is said to be a false negative (FN). If, however, the signal does not belong to class i and is wrongly classified as such, then it is counted as false positive (FP). Finally, if it does not really belong to i and is classified to belong to j , then it is a true negative (TN). To this end, the true positive rate (TPR) or recall can be defined as $TPR = TP/(TP + FN)$, whereas the false positive rate (FPR) can be defined as $FPR = FP/(FP + TN)$. ROC curves and AUROC curve values show the capability of a classifier to distinguish between different classes. ROC is a probabilistic curve which is plotted with a TPR on the vertical axis and FPR on the horizontal axis. Ideally, the TPR equals 1 and the FPR equals 0. Generally speaking, the closer the ROC curve is to the top-left corner, the better the performance. Similarly, the higher values of the AUROC curve shows better performance. In this work, there are three classes ($\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2$ or $\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_3$) and ROC curve for each class is plotted separately.

Figures 4 and 5 present a performance comparison of the proposed algorithm with the ED-based ML algorithm for PUEA and jamming attack detection, respectively. In the case of ED-based ML, the energy of the received signals is used for the detection of different hypotheses while using ML structure similar to the one used for the proposed algorithm. It is observed from Figs. 4 and 5 that the ROC curves of the proposed algorithm are closer to the top-left corner compared to ROC curves of the ED-based algorithm for PUEA and jamming attack detection. Moreover, Tables 4 and 5 also show that the values of AUROC of the proposed algorithms are higher compared to the AUROC values of the ED-based algorithms to detect different hypothesis presented in (4). For example, the proposed algorithm outperforms the ED-based algorithm by 2.24% in the case of PUEA and 6.88% in case of jamming attack detection in terms of AUROC values. This is because the energy patterns in the residual and gradient vector enhance the detection capability of the proposed algorithm compared to the ED-based algorithm.

From an ML point-of-view, a trained model (classifier in this work) should not memorize the inputs used in its training. To investigate this quality in the trained ML classifier model in the proposed algorithm, Fig. 6 shows the training and testing losses versus epochs for the PUEA detection when $M = 100$. In view of this figure, it is evident that the accuracy of the training sets converges to the test set. These results signify the absence of overfitting, thereby validating the generalizability of the proposed model. In other words,

Table 5 AUROC values for jamming attack

	PU	Hole	Jammer
ED-based	0.9097	0.9542	0.8820
Proposed	0.9830	0.9637	0.9508



the trained model does not memorize the training data. Here, it should be noted that we include the loss graph only for $M = 100$ case of PUEA to avoid repetition. For the other values of M and for the jammer case, we observe the similar behavior in loss graphs.

6 Conclusions

In this paper, the convergence patterns of sparse recovery are exploited for the purpose of PUEA and jamming attack detection. Sparse recovery was conducted over a legitimate PU channel-dependent dictionary. Consequently, the signal from the legitimate node has smooth convergence as compared to the signal from the illegitimate node. Essentially, this owes to the fact that this signal is the only one compressible in the domain exclusively defined by this sparsifying dictionary. Besides, the non-compressive nature of a jamming signal with sparse coding over a PU channel-dependent dictionary was also exploited to detect jamming attacks. This detection algorithm made use of ML-based approaches. Numerical experiments showed the effectiveness of the proposed algorithm and its superior performance compared to ED-based ML algorithms. These results were validated in terms of confusion matrices, ROC curves, and values of AUROC curves, as quality metrics. In terms of AUROC curve values, the proposed algorithm outperformed the ED-based algorithm by 2.24% in the case of PUEA and 6.88% in case of jamming attack detection.

A Appendix residual energy gradient decay analysis

The proposed algorithm is based on the convergence patterns in the sparse coding of the compressed received signal. More specifically, the proposed algorithm uses a channel-dependent dictionary to identify different characteristics of gradients and residuals to detect PUEA and jamming attack.

In this work, we employ the computationally efficient OMP for sparse coding. Let us focus on its first iteration for the sake of simplicity. At the start of the first OMP iteration, the signal itself is used to initialize the zero-th residual r_0 . Afterwards, OMP chooses an atom (d) from the atoms of the given dictionary D_{PU} that have the strongest similarity to the r_0 . This similarity is characterized by the projection corresponding to each atom as

$E = dd^\dagger$. The updated residual after the selection of atom can be given as

$$r_1 = r_0 - Er_0. \tag{A1}$$

For simplicity, the least-squares refinement of OMP is ignored. With each iteration, the residual magnitude is decreasing and the pattern of the concatenated residual values ($\|r_1\|_2^2$) is used for classification.

To this end, the first element in G can be represented as

$$G(1) = \|r_1\|_2^2 - \|r_0\|_2^2 = \langle r_1, r_1 \rangle - \langle r_0, r_0 \rangle. \tag{A2}$$

Using this gradient magnitude property, we can differentiate the cases $\mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2,$ and \mathcal{H}_3 . The general received signal can be given as $y = hx + n$, and we can write the $G(1)$ as follows:

$$G(1) = \|y - Ey\|_2^2 - \|y\|_2^2. \tag{A3}$$

For the first hypothesis, $\mathcal{H}_0, x = 0$. Thus, y is merely noise and can be written as

$$\begin{aligned} G(1)_{\mathcal{H}_0} &= \|n - En\|_2^2 - \|n\|_2^2, \\ &= \langle n - En, n - En \rangle - \langle n, n \rangle, \\ &= \langle n, n \rangle - 2 \langle n, En \rangle + \langle En, En \rangle - \langle n, n \rangle. \end{aligned} \tag{A4}$$

With respect to the properties of projection, we know that $\langle En, n \rangle = \langle En, En \rangle = \|En\|_2^2$. Hence, (A4) can be written as

$$G(1)_{\mathcal{H}_0} = -\|En\|_2^2. \tag{A5}$$

Following the same logic, $G(1)$ for $\mathcal{H}_1, \mathcal{H}_2,$ and \mathcal{H}_3 can be expressed as follows

$$\begin{aligned} G(1) &= \langle hx + n - E(hx + n), hx + n - E(hx + n) \rangle \\ &\quad - \langle hx + n, hx + n \rangle, \\ &= a - 2b + c - d, \end{aligned} \tag{A6}$$

where $a, b, c,$ and d are defined next. Specifically, a can be written as

$$\begin{aligned} a &= \langle hx + n, hx + n \rangle, \\ &= \langle hx + hx \rangle + \langle hx + n \rangle + \langle hx + n \rangle + \langle n + n \rangle, \\ &= \langle hx + hx \rangle + 2 \langle hx + n \rangle + \langle n + n \rangle. \end{aligned} \tag{A7}$$

Assuming that the noise is independent of $hx, \langle hx + n \rangle = 0$, we can write (A7) as

$$a = \langle hx + hx \rangle + \langle n + n \rangle. \tag{A8}$$

By its turn, b can be expressed as

$$\begin{aligned} b &= \langle hx + n, E(hx + n) \rangle, \\ &= \langle hx + n, Ehx + En \rangle, \\ &= \langle hx, Ehx \rangle + \langle hx, En \rangle + \langle n, Ehx \rangle + \langle n, En \rangle, \\ &= \langle Ehx, Ehx \rangle + \langle hx, En \rangle + \langle n, Ehx \rangle + \langle En, En \rangle, \\ &= \langle Ehx, Ehx \rangle + \langle En, En \rangle, \end{aligned} \tag{A9}$$

where $\langle hx, En \rangle = \langle n, Ehx \rangle = 0$.

Moreover, \mathbf{c} can be expressed as

$$\begin{aligned} \mathbf{c} &= \langle E(\mathbf{h}\mathbf{x} + \mathbf{n}), E(\mathbf{h}\mathbf{x} + \mathbf{n}) \rangle, \\ &= \langle E\mathbf{h}\mathbf{x} + E\mathbf{n}, E\mathbf{h}\mathbf{x} + E\mathbf{n} \rangle, \\ &= \langle E\mathbf{h}\mathbf{x}, E\mathbf{h}\mathbf{x} \rangle + \langle E\mathbf{n}, E\mathbf{n} \rangle. \end{aligned} \tag{A10}$$

Lastly, \mathbf{d} can be given as

$$\mathbf{d} = \langle \mathbf{h}\mathbf{x} + \mathbf{h}\mathbf{x} \rangle + \langle \mathbf{n} + \mathbf{n} \rangle = \mathbf{a}. \tag{A11}$$

Based on (A8), (A9), (A10), and (A11), and making the appropriate substitution, $\mathbf{G}(1)$ can be written as

$$\begin{aligned} \mathbf{G}(1) &= \mathbf{a} - 2\mathbf{b} + \mathbf{c} - \mathbf{d}, \\ &= -2 \langle E\mathbf{h}\mathbf{x}, E\mathbf{h}\mathbf{x} \rangle - 2 \langle E\mathbf{n}, E\mathbf{n} \rangle \\ &\quad + \langle E\mathbf{h}\mathbf{x}, E\mathbf{h}\mathbf{x} \rangle + \langle E\mathbf{n}, E\mathbf{n} \rangle, \\ &= - \langle E\mathbf{h}\mathbf{x}, E\mathbf{h}\mathbf{x} \rangle - \langle E\mathbf{n}, E\mathbf{n} \rangle. \end{aligned} \tag{A12}$$

Finally, the generic expression of the gradient magnitude for hypotheses \mathcal{H}_1 , \mathcal{H}_2 , and \mathcal{H}_3 can be expressed

$$\mathbf{G}(1)_{\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3} = -\|E\mathbf{h}\mathbf{x}\|_2 - \|E\mathbf{n}\|_2, \tag{A13}$$

where \mathbf{h} corresponds to \mathbf{h}_{PU} in case of PU or \mathbf{h}_i in case of PUE/jammer as explained in (4). Moreover, \mathbf{x} will be structured in case of PU and PUE, while unstructured in the case of a jamming attack.

Abbreviations

AUROC: Area under receiver operating characteristics; CR: Cognitive radio; CS: Compressive sensing; ED: Energy detection; FFT: Fast fourier transform; FN: False-negative; FP: False positive; FPR: False positive rate; FSK: Frequency-shift keying; OMP: Orthogonal matching pursuit; PSK: Phase-shift keying; PAM: Pulse amplitude modulation; PU: Primary user; PUE: Primary user emulator; PUEA: Primary user emulation attack; QAM: Quadrature amplitude modulation; QoS: Quality of service; ML: Machine learning; ROC: Receiver operating characteristics; SNR: Signal-to-noise ratio; SSDF: Spectrum sensing data falsification; SU: Secondary users; SVDD: Support vector data description; TN: True negative; TPR: True positive rate; TP: True positive

Authors’ contributions

HM, MA, and MN developed the main idea under the supervision of H. A². They implemented the basic simulations and drafted the manuscript jointly. All authors participated in the development of the main idea, its implementation, and the interpretation of the results. The authors have read and approved the manuscript.

Availability of data and materials

Please contact the corresponding author at haji.madni@std.medipol.edu.tr.

Competing interests

The authors declare that they have no competing interests.

Author details

¹School of Engineering and Natural Sciences, Istanbul Medipol University, 34810 Istanbul, Turkey. ²Department of Electrical Engineering, University of South Florida, 3620 Tampa, USA.

Received: 6 November 2019 Accepted: 20 May 2020

Published online: 02 July 2020

References

1. Y. Arjoune, N. Kaabouch, A comprehensive survey on spectrum sensing in cognitive radio networks: recent advances, new challenges, and future research directions. *Sensors*. **19**(1), 126 (2019). <https://doi.org/10.3390/s19010126>
2. T. Yucek, H. Arslan, A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE Commun. Surv. Tuts.* **11**(1), 116–130 (2009). <https://doi.org/10.1109/surv.2009.090109>

²H. Arslan was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under Grant 119E433.

3. A. G. Fragkiadakis, E. Z. Tragos, I. G. Askoxyllakis, A survey on security threats and detection techniques in cognitive radio networks. *IEEE Commun. Surv. Tuts.* **15**(1), 428–445 (2013). <https://doi.org/10.1109/surv.2011.122211.00162>
4. B. Wang, Y. Wu, K. J. R. Liu, T. C. Clancy, An anti-jamming stochastic game for cognitive radio networks. *IEEE J. Sel. Areas Commun.* **29**(4), 877–889 (2011). <https://doi.org/10.1109/jsac.2011.110418>
5. M. Bouabdellah, N. Kaabouch, F. E. Bouanani, H. Ben-Azza, Network layer attacks and countermeasures in cognitive radio networks: a survey. *J. Inf. Security Appl.* **38**, 40–49 (2018). <https://doi.org/10.1016/j.jisa.2017.11.010>
6. D. Chaitanya, K. M. Chari, in *Proc. Comp. Commun., Netw. Internet Secur.*, Performance analysis of PUEA and SSDF attacks in cognitive radio networks (Springer, Singapore, 2017), pp. 219–225. https://doi.org/10.1007/978-981-10-3226-4_21
7. F. Jin, V. Varadarajan, U. Tupakula, in *Proc. IEEE Int. Telecommun. Net. Appl. Conf. (ITNAC)*, Improved detection of primary user emulation attacks in cognitive radio networks, (2015), pp. 274–279. <https://doi.org/10.1109/atnac.2015.7366825>
8. Z. Yuan, D. Niyato, H. Li, J. B. Song, Z. Han, Defeating primary user emulation attacks using belief propagation in cognitive radio networks. *IEEE J. Sel. Areas Commun.* **30**(10), 1850–1860 (2012). <https://doi.org/10.1109/jsac.2012.121102>
9. L. Qian, X. Li, S. Wei, in *Proc. Int. Conf. Computing, Netw. Commun. (ICNC)*, Cross-layer detection of stealthy jammers in multihop cognitive radio networks, (2013), pp. 1026–1030. <https://doi.org/10.1109/iccnc.2013.6504232>
10. S. U. Rehman, K. W. Sowerby, C. Coghill, Radio-frequency fingerprinting for mitigating primary user emulation attack in low-end cognitive radios. *Inst. Engr. Technol. Commun.* **8**(8), 1274–1284 (2014). <https://doi.org/10.1049/iet-com.2013.0568>
11. W. Chin, C. Tseng, C. Tsai, W. Kao, C. Kao, in *Proc. IEEE 75th Veh. Technol. Conf. (VTC Spring)*, Channel-based detection of primary user emulation attacks in cognitive radios, (2012), pp. 1–5. <https://doi.org/10.1109/vetecs.2012.6239877>
12. Y. Li, X. Ma, M. Wang, H. Chen, L. Xie, in *Proc. Smart Innov. Commun. Comput. Sci.* ed. by B. K. Panigrahi, M. C. Trivedi, K. K. Mishra, S. Tiwari, and P. K. Singh, Detecting primary user emulation attack based on multipath delay in cognitive radio network (Springer, Singapore, 2019), pp. 361–373. https://doi.org/10.1007/978-981-10-8968-8_31
13. N. T. Nguyen, R. Zheng, Z. Han, On identifying primary user emulation attacks in cognitive radio systems using non-parametric Bayesian classification. *IEEE Trans. Signal Process.* **60**(3), 1432–1445 (2012). <https://doi.org/10.1109/tsp.2011.2178407>
14. R. Chen, J. Park, J. H. Reed, Defense against primary user emulation attacks in cognitive radio networks. *IEEE J. Sel. Areas Commun.* **26**(1), 25–37 (2008). <https://doi.org/10.1109/jsac.2008.080104>
15. N. Patwari, S. K. Kasera, in *Proc. 13th Annual ACM Int. Conf. Mobile Comput. Netw.*, Robust location distinction using temporal link signatures (ACM Press, New York, NY, USA, 2007), pp. 111–122. <https://doi.org/10.1145/1287853.1287867>
16. W. R. Ghanem, M. Shokair, M. I. Desouky, in *Proc. 33rd National Radio Sci. Conf. (NRSC)*, An improved primary user emulation attack detection in cognitive radio networks based on firefly optimization algorithm, (2016), pp. 178–187. <https://doi.org/10.1109/nrsc.2016.7450851>
17. S. Liu, Y. Chen, W. Trappe, L. J. Greenstein, in *Proc. IEEE Int. Conf. Comp. Commun. (INFOCOM)*, ALDO: an anomaly detection framework for dynamic spectrum access networks, (2009), pp. 675–683. <https://doi.org/10.1109/infcom.2009.5061975>
18. Z. Luo, C. Lou, S. Chen, S. Zheng, S. Li, in *Proc. IEEE 11th Int. Symp. Commun. & Inf. Technol. (ISCIT)*, Specific primary user sensing for wireless security in IEEE 802.22 network, (2011), pp. 18–22. <https://doi.org/10.1109/iscit.2011.6089728>
19. S. Arul Selvi, M. Sundararajan, SVM based two level authentication for primary user emulation attack detection. *Indian J. Sci. Technol.* **9**(29), 1–8 (2016). <https://doi.org/10.17485/ijst/2016/v9i29/89270>
20. X. Zhang, Y. Ma, Y. Gao, S. Cui, Real-time adaptively regularized compressive sensing in cognitive radio networks. *IEEE Trans. Veh. Technol.* **67**(2), 1146–1157 (2018). <https://doi.org/10.1109/tvt.2017.2749254>
21. J. W. Choi, B. Shim, Y. Ding, B. Rao, D. I. Kim, Compressed sensing for wireless communications: useful tips and tricks. *IEEE Commun. Surv. Tuts.* **19**(3), 1527–1550 (2017). <https://doi.org/10.1109/comst.2017.2664421>
22. M. Nazzal, A. R. Ekti, A. Gorcin, H. Arslan, Exploiting sparsity recovery for compressive spectrum sensing: a machine learning approach. *IEEE Access.* **7**, 126098–126110 (2019). <https://doi.org/10.1109/access.2019.2909976>
23. F. Ye, X. Zhang, Y. Li, H. Huang, Primary user localization algorithm based on compressive sensing in cognitive radio networks. *Algorithms.* **9**(2) (2016). <https://doi.org/10.3390/a9020025>
24. S. Maric, A. Biswas, S. Reisenfeld, in *Proc. Int. Conf. Signals Syst. (ICSigSys)*, A complete algorithm to diagnose and alleviate the effects of physical layer attacks, (2017), pp. 29–34. <https://doi.org/10.1109/icsigsys.2017.7967058>
25. M. O. Mughal, T. Nawaz, L. Marcenaro, C. S. Regazzoni, in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Cyclostationary-based jammer detection algorithm for wide-band radios using compressed sensing, (2015), pp. 280–284. <https://doi.org/10.1109/globalsip.2015.7418201>
26. M. A. Davenport, P. T. Boufounos, M. B. Wakin, R. G. Baraniuk, Signal processing with compressive measurements. *IEEE J. Sel. Areas Commun.* **4**(2), 445–460 (2010). <https://doi.org/10.1109/jstsp.2009.2039178>
27. E. C. Marques, N. Maciel, N. Naviner, in *IEEE Access*, A review of sparse recovery algorithms, vol. 7, (2019), pp. 1300–1322. <https://doi.org/10.1109/access.2018.2886471>
28. M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.* **54**(11), 4311–4322 (2006). <https://doi.org/10.1109/tip.2006.881969>
29. C. Jiang, H. Zhang, Y. Ren, Z. Han, K. Chen, L. Hanzo, Machine learning paradigms for next-generation wireless networks. *IEEE Wireless Commun.* **24**(2), 98–105 (2017). <https://doi.org/10.1109/mwc.2016.1500356wc>
30. M. H. Beale, M. T. Hagan, H. B. Demuth, Neural network toolbox. User’s Guide, MathWorks. **2**, 77–81 (2010)
31. J. M. Hamamreh, H. M. Furqan, H. Arslan, Classifications and applications of physical layer security techniques for confidentiality: a comprehensive survey. *IEEE Commun. Surv. Tuts.* **21**(2), 1773–1828 (2019). <https://doi.org/10.1109/comst.2018.2878035>
32. B. L. Sturm, M. G. Christensen, in *Proc. 20th European Signal Process. Conf. (EUSIPCO)*, Comparison of orthogonal matching pursuit implementations, (2012), pp. 220–224

33. K. He, J. Sun, in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, Convolutional neural networks at constrained time cost, (2015), pp. 5353–5360. <https://doi.org/10.1109/cvpr.2015.7299173>
34. L. Xiao, L. J. Greenstein, N. B. Mandayam, W. Trappe, Using the physical layer for wireless authentication in time-variant channels. *IEEE Trans. Wireless Commun.* **7**(7), 2571–2579 (2008). <https://doi.org/10.1109/twc.2008.070194>
35. H. M. Furqan, J. M. Hamamreh, H. Arslan, Adaptive OFDM-IM for enhancing physical layer security and spectral efficiency of future wireless networks. *Wireless Commun. Mob. Comput.* **2018**, 1–6 (2018). <https://doi.org/10.1155/2018/3178303>

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)
