Machine Learning Techniques for Cooperative Spectrum Sensing in Cognitive Radio Networks

Karaputugala Madushan Thilina, Kae Won Choi, Nazmus Saquib, and Ekram Hossain

***Abstract*—We propose novel cooperative spectrum sensing (CSS) algorithms for cognitive radio (CR) networks based on machine learning techniques which are used for pattern classification. In this regard, unsupervised (e.g., K-means clustering and Gaussian mixture model (GMM)) and supervised (e.g., support vector machine (SVM) and weighted K-nearest-neighbor (KNN)) learning-based classification techniques are implemented for CSS. For a radio channel, the vector of the energy levels estimated at CR devices is treated as a feature vector and fed into a classifier to decide whether the channel is available or not. The classifier categorizes each feature vector into either of the two classes, namely, the "channel available class" and the "channel unavailable class". Prior to the online classification, the classifier needs to go through a training phase. For classification, the K-means clustering algorithm partitions the training feature vectors into K clusters, where each cluster corresponds to a combined state of primary users (PUs) and then the classifier determines the class the test energy vector belongs to. The GMM obtains a mixture of Gaussian density functions that well describes the training feature vectors. In the case of the SVM, the support vectors (i.e., a subset of training vectors which fully specify the decision function) are obtained by maximizing the margin between the separating hyperplane and the training feature vectors. Furthermore, the weighted KNN classification technique is proposed for CSS for which the weight of each feature vector is calculated by evaluating the area under the receiver operating characteristic (ROC) curve of that feature vector. The performance of each classification technique is quantified in terms of the average training time, the sample classification delay, and the ROC curve. Our comparative results clearly reveal that the proposed algorithms outperform the existing state-of-the-art CSS techniques.**

***Index Terms*—Cognitive radio, cooperative spectrum sensing, K-means clustering, GMM, support vector machine (SVM), K-nearest-neighbor, primary user detection**

I. Introduction

THE CONCEPT of cognitive radio (CR) for designing wireless communications systems has emerged since last decade to mitigate the scarcity problem of limited radio spectrum by improving the utilization of the spectrum [1]. The CR refers to an intelligent wireless communications device, which senses its operational electromagnetic environment and can dynamically and autonomously adjust its radio operating parameters. In this context, opportunistic spectrum access (OSA) is a key concept, which allows a CR device to opportunistically access the frequency band allocated to a primary user (PU) when the PU transmission is detected to be inactive [2]–[4]. For OSA, the CR devices have to sense the radio spectrum licensed to the PUs by using its limited resources (e.g., energy and computational power), and subsequently utilize the available spectrum opportunities to maximize its performance objectives. Therefore, efficient spectrum sensing is crucial for OSA.

Cooperative spectrum sensing (CSS) can be used when the CR devices are distributed in different locations. It is possible for the CR devices to cooperate in order to achieve higher sensing reliability than individual sensing does [5] by yielding a better solution to the hidden PU problem that arises because of shadowing [6] and multi-path fading [7]. In cooperative sensing, the CR devices exchange the sensing results with the fusion center for decision making [5]. With hard fusion algorithms, the CR devices exchange only one-bit information with the fusion center, which indicates whether the received energy is above a particular threshold. For example, the OR-rule [8], the AND-rule, the counting rule [9], and the linear quadratic combining rule [10] are commonly used for CSS. In [11], a softened hard fusion scheme with two-bit overhead for each CR device is considered. In soft decision algorithms [11], [12], the exact energy levels estimated at the CR devices are transmitted to the fusion center to make a better decision. In [13], the authors propose an optimal linear fusion algorithm for spectrum sensing. Relay-based cooperative spectrum sensing schemes are studied in [14], [15].

In this paper, we propose novel CSS schemes based on machine learning techniques. The machine learning techniques are often used for pattern classification, where a feature vector is extracted from a pattern and is fed into the classifier which categorizes the pattern into a certain class. In the context of CSS, we treat an "energy vector", each component of which is an energy level estimated at each CR device, as a feature vector. Then, the classifier categorizes the energy vector into one of two classes: the "channel available class" (corresponding to the case that no PU is active) and the "channel unavailable class" (corresponding to the case that at least one PU is active). Prior to online classification, the classifier has to go through a training phase where it learns from training feature vectors. According to the type of learning method adopted, a classification algorithm can be categorized as unsupervised learning (e.g., K-means clustering and Gaussian mixture model (GMM)) or supervised learning (e.g., support vector machine (SVM) and K-nearest neighbor (KNN)) [16]–

Manuscript received November 15, 2012; revised April 4, 2013. This research was supported by the grant STPGP 380888-09 from the Natural Sciences and Engineering Research Council of Canada (NSERC) under the Strategic Project Grant (SPG) program.

K. M. Thilina, N. Saquib, and E. Hossain are with the Department of Electrical and Computer Engineering at the University of Manitoba, Canada (e-mail: Ekram.Hossain@umanitoba.ca).

K. W. Choi is with the Department of Computer Science and Engineering at the Seoul National University of Science and Technology (SeoulTech), Korea.

Digital Object Identifier 10.1109/JSAC.2013.131120.

[21]. In supervised (resp., unsupervised) learning, a training feature vector is fed into the classifier with (resp., without) its label indicating the actual class the training feature vector belongs to. In this paper, we propose both unsupervised and supervised learning techniques for CSS.

The proposed machine learning-based CSS techniques have the following advantages over the traditional CSS techniques.

- The proposed techniques are capable of implicitly learning the surrounding environment (e.g., the topology of the PU and the CR networks and the channel fading) in an online fashion. Therefore, the proposed techniques are much more adaptive than the traditional CSS techniques, which need prior knowledge about the environment for optimization.
- The proposed techniques can describe more optimized decision region[1] on the feature space than the traditional CSS techniques (e.g., OR/AND-rule-based and linear fusion techniques) can, which results in better detection performance.

In spite of these advantages, there have been only few studies on the application of machine learning techniques to CSS for the CR networks. The authors in [22] suggest a pattern recognition-based linear fusion rule for CSS, in which linear coefficients are obtained by using the Fisher linear discriminant analysis. To the best of our knowledge, other than [22], the machine learning techniques have not been adopted for CSS in the existing literature.

The main contributions of our paper are as follows.

- We propose to use unsupervised learning approaches such as the K-means clustering and the GMM for CSS. The K-means clustering algorithm partitions the features into K clusters. Each cluster is mapped to either the channel available class or the channel unavailable class. On the other hand, in the GMM, we obtain a Gaussian mixture distribution from training feature vectors, where each Gaussian distribution in the mixture distribution corresponds to a cluster.
- Due to their higher prediction capability, we also propose to use supervised learning approaches such as the SVM and the KNN for CSS. In the SVM, the support vectors (i.e., a subset of training vectors which fully specify the decision function) are obtained by maximizing the margin between separating hyperplanes and feature vectors. In addition, the weighted KNN classification technique is also investigated for CSS with different distance measures.
- The performance of each of the classification techniques is evaluated in terms of the training time, the classification delay, and the ROC curve. The effect of the number of the CR devices in cooperation is also quantified.

The rest of the paper is organized as follows. In Section II, we present the system model and the assumptions. The machine learning-based cooperative spectrum sensing framework is presented in Section III. Then, we describe unsupervised and supervised CSS algorithms in Sections IV and V, respectively. Section VI presents the performance evaluation results for the proposed CSS algorithms. Lastly, Section VII concludes the paper.

II. System Model and Assumptions

A. Cognitive Radio Network and Primary User Model

We consider a CR network which shares a frequency channel with PUs. Henceforth, a CR device in the CR network will be called a secondary user (SU). The CR network consists of N secondary users (SUs), each of which is indexed by $n = 1, \ldots, N$. SU n is located at the coordinate $\mathbf{c}_n^{\mathrm{SU}}$ in the two-dimensional space. For cooperative sensing, each SU estimates the energy level and reports it to another SU which takes the role of a fusion center. The fusion center determines the channel availability based on the energy levels reported by all SUs.

In this paper, we adopt a very generalized PU model where multiple PUs alternate between active and inactive states. There are M PUs, each of which is indexed by $m = 1, \ldots, M$. Let $\mathbf{c}_m^{\mathrm{PU}}$ denote the coordinate of PU m in the two-dimensional space. Let S_m indicate the state of PU m. We have $S_m = 1$ if PU m is in the active state (i.e., PU m transmits a signal); and $S_m = 0$ otherwise. Let $\mathbf{S} = (S_1, \ldots, S_M)^T$ be the vector of the states of all PUs, where the superscript T denotes the transpose operation. The probability that $\mathbf{S} = \mathbf{s}$ for given $\mathbf{s} = (s_1, \ldots, s_M)^T$ is denoted by

$$v(\mathbf{s}) = \Pr[\mathbf{S} = \mathbf{s}]. \tag{1}$$

If at least one PU is active (i.e., $S_m = 1$ for some m), the channel is considered as unavailable for the CR network to access. The channel is available only when there is no PU in the active state (i.e., $S_m = 0,\ \forall\ m$). If we let A denote the channel availability, we have

$$A = \begin{cases} -1, & \text{if } S_m = 1 \text{ for some } m \\ 1, & \text{if } S_m = 0 \text{ for all } m. \end{cases} \tag{2}$$

B. Energy Vector Model

To estimate the energy level, an SU performs energy detection for a time duration of τ. If we denote the bandwidth by w, the energy detector takes $w\tau$ baseband complex signal samples during τ. Let $Z_n(i)$ denote the ith signal sample taken by SU n. The signal samples consist of the summation of the signals from all PUs in the active state and the thermal noise, that is,

$$Z_n(i) = \sum_{m=1}^{M} S_m h_{m,n} X_m(i) + N_n(i), \tag{3}$$

where $h_{m,n}$ denotes the channel gain from PU m to SU n, $X_m(i)$ is the signal transmitted by PU m, and $N_n(i)$ is the thermal noise at SU n. The transmission power of PU m is assumed to be fixed to $\rho_m = \sum_{i=1}^{w\tau} \mathrm{E}[|X_m(i)|^2]/\tau$ and the noise spectral density is denoted by $\eta = \mathrm{E}[|N_n(i)|^2]$. The energy detector of SU n estimates the energy level normalized by the noise spectral density, which is denoted by Y_n, from the signal samples as

$$Y_n = \frac{2}{\eta} \sum_{i=1}^{w\tau} |Z_n(i)|^2. \tag{4}$$

[1] The classifier categorizes a feature vector according to which decision region the feature vector falls in.

All SUs report the estimated energy levels to the fusion center and the fusion center generates the "energy vector," which is defined as

$$\mathbf{Y} = (Y_1, \ldots, Y_N)^T. \tag{5}$$

Now, we investigate the distribution of the energy vector. It is known that, conditioned on $\mathbf{S} = \mathbf{s}$, the energy level Y_n follows a noncentral chi-squared distribution with the degree of freedom $q = 2w\tau$ and the non-centrality parameter,

$$\zeta_n = \frac{2\tau}{\eta} \sum_{m=1}^{M} s_m g_{m,n} \rho_m, \tag{6}$$

where $g_{m,n}$ is the power attenuation from PU m to SU n such that $g_{m,n} = |h_{m,n}|^2$. The power attenuation $g_{m,n}$ is given as

$$g_{m,n} = PL(\|\mathbf{c}_m^{\text{PU}} - \mathbf{c}_n^{\text{SU}}\|) \cdot \psi_{m,n} \cdot \nu_{m,n}, \tag{7}$$

where $\| \cdot \|$ is the Euclidean distance, $PL(d) = d^{-\alpha}$ is the path-loss component for relative distance d with the path-loss exponent α, $\psi_{m,n}$ is the shadow fading component, and $\nu_{m,n}$ is the multi-path fading component. We assume that PUs and SUs are immobile (e.g., the base station (BS), the consumer premise equipment (CPE), and the TV station in IEEE 802.22-based wireless regional area network (WRAN)). We assume that the shadow fading and the multi-path fading components are quasi-static during the time of interest.

If the number of samples (i.e., $w\tau$) is sufficiently large, the distribution of the energy level Y_n given that $\mathbf{S} = \mathbf{s}$ can be approximated by a Gaussian distribution with mean $\mu_{Y_n|\mathbf{S}=\mathbf{s}}$ and variance $\sigma^2_{Y_n|\mathbf{S}=\mathbf{s}}$, where

$$\begin{aligned}\mu_{Y_n|\mathbf{S}=\mathbf{s}} &= \mathrm{E}[Y_n|\mathbf{S}=\mathbf{s}] = q + \zeta_n \\ &= 2w\tau + \frac{2\tau}{\eta}\sum_{m=1}^{M} s_m g_{m,n} \rho_m, \qquad (8) \\ \sigma^2_{Y_n|\mathbf{S}=\mathbf{s}} &= \mathrm{E}[(Y_n - \mu_{Y_n|\mathbf{S}=\mathbf{s}})^2|\mathbf{S}=\mathbf{s}] = 2(q + 2\zeta_n) \\ &= 4w\tau + \frac{8\tau}{\eta}\sum_{m=1}^{M} s_m g_{m,n} \rho_m. \qquad (9)\end{aligned}$$

Therefore, the energy vector $\mathbf{Y}$ given that $\mathbf{S} = \mathbf{s}$ follows a multivariate Gaussian distribution with the mean vector and the covariance matrix such that

$$\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}} = (\mu_{Y_1|\mathbf{S}=\mathbf{s}}, \ldots, \mu_{Y_N|\mathbf{S}=\mathbf{s}})^T, \tag{10}$$

$$\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}} = \mathrm{diag}(\sigma^2_{Y_1|\mathbf{S}=\mathbf{s}}, \ldots, \sigma^2_{Y_N|\mathbf{S}=\mathbf{s}}), \tag{11}$$

where $\mathrm{diag}(x_1, \ldots, x_N)$ denotes the diagonal matrix whose diagonal entries are $x_1, \ldots, x_N$.

III. Machine Learning-Based Cooperative Spectrum Sensing Framework

A. Operation of Proposed CSS Framework

The purpose of the proposed CSS techniques is to correctly determine the channel availability A based on the given energy vector $\mathbf{Y}$. In the context of machine learning, this is equivalent to constructing a classifier to correctly map the energy vector $\mathbf{Y}$ to the channel availability A. Therefore, an energy vector in our problem is analogous to a feature in the machine learning terminology. To construct the classifier, the first step is to collect a sufficient number of training energy vectors. Let $\mathbf{y}^{(l)}$ denote the lth training energy vector and let $a^{(l)}$ denote the channel availability corresponding to $\mathbf{y}^{(l)}$. Then, the set of the training energy vectors, i.e., $\overline{\mathbf{y}} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(L)}\}$ (where L is the number of training samples), is fed into the classifier for training. In case of unsupervised learning, each of the training energy vectors does not need to be labeled with the corresponding channel availability. On the other hand, supervised learning requires to have the set of the channel availabilities, i.e., $\overline{a} = \{a^{(1)}, \ldots, a^{(L)}\}$, for training as well as the set of the training energy vectors.

Next, the classifier is trained by using the training energy vectors. The training procedure differs for each machine learning technique under consideration. For example, in case of K-means clustering, the training involves partitioning the training energy vectors into K clusters and the centroid of each cluster is used later for classification. In another example, the SVM tries to find the maximum-margin hyperplane that splits the training energy vectors as clearly as possible.

Once the classifier is successfully trained, it is ready to receive the test energy vector for classification. Let $\mathbf{y}^*$ denote the test energy vector received by the classifier and let a^* denote the corresponding channel availability. In addition, let $\widehat{a}$ denote the channel availability determined by the classifier. The classifier categorizes the energy vector $\mathbf{y}^*$ into either "channel available class" (i.e., $\widehat{a} = 1$) or "channel unavailable class" (i.e., $\widehat{a} = -1$). If the energy vector is classified into the channel available class (resp., the channel unavailable class), it means that there is no PU (resp., at least one PU) in the active state and the channel is available (resp., unavailable) for the CR network to access. Therefore, the channel availability is correctly determined in the case that $\widehat{a} = a^*$, while misdetection (resp., false alarm) occurs in the case that $\widehat{a} = 1$ and $a^* = -1$ (resp., $\widehat{a} = -1$ and $a^* = 1$).

In Fig. 1, we illustrate the modular architecture of the proposed CSS framework, which consists of the training module and the classification module. In this architecture, the training and classification modules can operate independently. Whenever the CR network needs to find out the channel availability, the CR network generates the test energy vector and puts it into the classification module. The classification module determines the channel availability based on the test energy vector by using the classifier. Usually, finding the channel availability in the CR network requires very short delay. The classification delay of the proposed CSS techniques can meet this requirement due to low complexity.

The training module is responsible to train the classifier from the training energy samples and to provide the classification module with a trained classifier. The training module can be activated when the CR network is first deployed and when the radio environment changes (e.g., when the PU network changes its configuration). In addition, the CR network can periodically activate the training module to catch up with the changing environment. The training procedure of machine learning techniques generally takes a long time. However, this is not a significant problem since the training module is activated only by the above-mentioned events. Moreover, the training procedure can be performed in the background while the classification module operates normally.

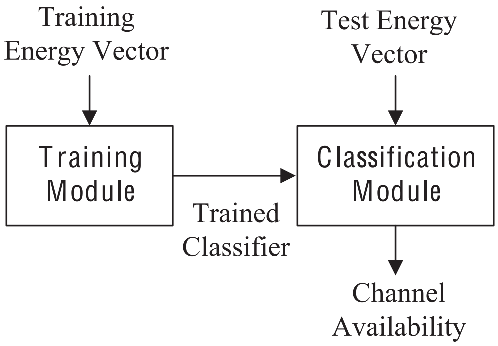


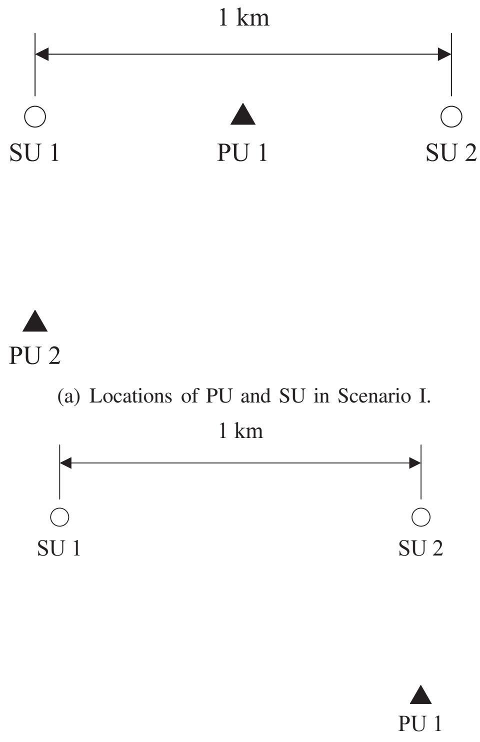
Fig. 1. Modular architecture of the proposed CSS framework.

B. Advantages of Proposed CSS Framework

The advantage of the proposed machine learning-based CSS framework over the traditional CSS techniques is twofold.

- The proposed CSS techniques retain a learning ability since the optimized classifier is learnt from the training energy vectors, which makes the proposed CSS techniques adaptive to changing radio environment without human intervention. The training procedure is fully autonomous in that it does not require any prior information about the environment and does not involve any parameter setting. Moreover, the proposed CSS techniques can adapt themselves to the changing environment by retraining the classifier periodically.
- The proposed CSS techniques can describe more optimized decision surface than the traditional CSS techniques can, which result in better performance in terms of the detection and the false alarm probabilities. The generalized multiple PU model in this paper leads to very complex probability space of the energy vector, which cannot be handled by the traditional CSS techniques. However, the proposed CSS techniques can find the decision surface which efficiently classifies the energy vectors even in the multiple PU model.

In Fig. 3, we present example scatter plots of the energy vectors of two SUs in two different scenarios to highlight the advantages of the proposed CSS techniques. In Scenario I, there are two PUs whose locations are given in Fig. 2(a). The PUs in Scenario I are activated according to the probability of $v((0,0)^T) = 0.36$, $v((0,1)^T) = 0.24$, $v((1,0)^T) = 0.24$, and $v((1,1)^T) = 0.16$. In Scenario II, there is only one PU whose location is given in Fig. 2(b). In this scenario, the PU is activated with the probability of $v((1)) = 0.5$. In Figs. 3(a)–3(d)[2], the energy vectors as well as the decision surfaces of the proposed technique are plotted for each scenario. The decision surface divides the energy vectors into two decision regions – one for the channel available class and the other for the channel unavailable class. In these figures, one of the proposed techniques, the Gaussian mixture model (GMM) technique, is used to draw the decision surface. The GMM will be explained in detail in Section IV-C. The threshold for classification in the GMM (i.e., δ) is set to zero for Fig. 3. Figs. 3(a) and 3(b) are plotted for Scenarios I while Figs. 3(c) and 3(d) are plotted for Scenario II. The transmission power of each PU is 200 mW in Figs. 3(a) and 3(c), and is 80 mW in Figs. 3(b) and 3(d).



(a) Locations of PU and SU in Scenario I.

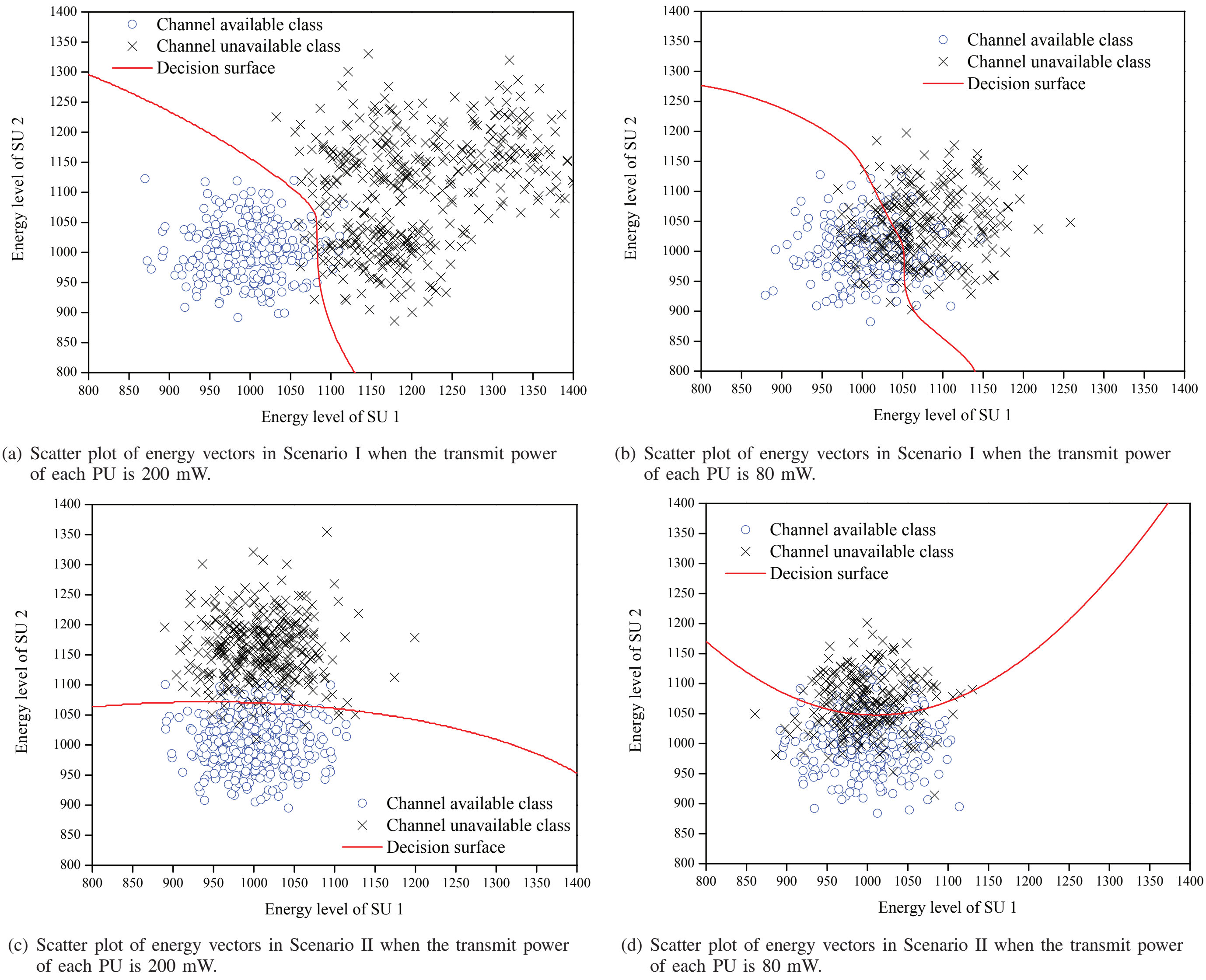
(b) Locations of PU and SU in Scenario II.

Fig. 2. Two scenarios of user locations.

From Fig. 3, we can notice the following advantages of the proposed machine learning-based CSS framework.

- We can see that the GMM technique is able to adaptively adjust the decision surface for different scenarios. Suppose that the CR network has the configuration in Scenario I and the transmission power of each PU is 200 mW. In this case, the CR network has the decision surface as shown in Fig. 3(a). Then, suppose that the PU network changes its configuration to Scenario II. The CR network can adapt to this change by gathering the energy vectors for a while and recalculate the decision surface as shown in Fig. 3(c). This process is autonomous and does not require any human intervention.
- We can see that the decision surface divides the energy vectors in each class as clearly as possible, which leads to improved detection performance. In Figs. 3(a) and 3(b), the decision surface, derived by the GMM technique, optimally separates the energy vectors. This decision surface takes a complex form, which cannot be described by any other existing CSS technique.

[2] The simulation parameter values for Fig. 3 are as follows: the bandwidth w is 5 MHz, the sensing duration τ is 100 μs, the noise spectral density η is -174 dBm, and the path-loss exponent α is 4. We assume that the shadow fading and the multi-path fading components are fixed as $\psi_{m,n} = 1$ and $\nu_{m,n} = 1$.



(a) Scatter plot of energy vectors in Scenario I when the transmit power of each PU is 200 mW.

(b) Scatter plot of energy vectors in Scenario I when the transmit power of each PU is 80 mW.

(c) Scatter plot of energy vectors in Scenario II when the transmit power of each PU is 200 mW.

(d) Scatter plot of energy vectors in Scenario II when the transmit power of each PU is 80 mW.

Fig. 3. Example scatter plots of energy vectors in two scenarios.

IV. Unsupervised Learning for Cooperative Spectrum Sensing

A. Motivation for Unsupervised Learning

In this section, we propose unsupervised learning approaches for the proposed CSS framework. In case of unsupervised learning, only the training energy vectors (i.e., $\overline{\mathbf{y}} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(L)}\}$) are fed into the classifier for training. Unsupervised learning does not need the information regarding the channel availability corresponding to each training energy vector, i.e., $\overline{a} = \{a^{(1)}, \ldots, a^{(L)}\}$. Therefore, unsupervised learning can be easily implemented in a practical sense compared to the supervised learning which requires $\overline{a}$ for training.

Since there is no explicit teacher which helps training, unsupervised learning has to rely on the inherent clustering structure of the training energy vectors. Recall that the energy vector $\mathbf{Y}$ given $\mathbf{S} = \mathbf{s}$ follows a multivariate Gaussian distribution with the mean vector $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$ and the covariance matrix $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$. For each possible combination of the states of PUs, a cluster of the training energy vectors is formed according to the respective multivariate Gaussian distribution. In Fig. 3(a), we observe that four visible clusters are formed each of which respectively corresponds to the cases that $(S_1, S_2)^T$ is $(0,0)^T$, $(0,1)^T$, $(1,0)^T$, and $(1,1)^T$. If there are M PUs, the number of clusters can be calculated as $K = 2^M$. Each cluster is indexed by $k = 1, \ldots, K$.

More specifically, the training energy vectors are samples taken out of the Gaussian mixture distribution the pdf of which is as follows:

$$f(\mathbf{x}) = \sum_{\mathbf{s}} v(\mathbf{s}) \cdot \phi(\mathbf{x}|\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}, \boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}), \tag{12}$$

where $v(\mathbf{s})$ is the probability that $\mathbf{S} = \mathbf{s}$ (i.e., $v(\mathbf{s}) = \Pr[\mathbf{S} = \mathbf{s}]$) and $\phi(\mathbf{x}|\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}, \boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}})$ is the pdf of the multivariate Gaussian distribution such that

$$\phi(\mathbf{x}|\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}, \boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}) = \frac{1}{(2\pi)^{N/2}|\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}|^{1/2}}$$

$$\exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}})^T \boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}^{-1} (\mathbf{x} - \boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}})\right\}. \tag{13}$$

The samples from the Gaussian mixture distribution form discernible clusters as shown in Figs. 3(a) and 3(c) if the transmission power of each PU is high. However, clusters are not visually separable in Figs. 3(b) and 3(d) in the case that the transmission power of each PU is low. It is worth noting that, even in the case of low transmission power, the proposed CSS scheme is able to obtain the decision surface separating the channel available and channel unavailable classes as shown in Figs. 3(b) and 3(d).

Among all clusters, only one cluster corresponding to the case that no PU is in the active state (i.e., $\mathbf{S} = \mathbf{0}$ for the zero vector $\mathbf{0}$) can be mapped to the channel available class, while all the other clusters are mapped to the channel unavailable class. Without loss of generality, we designate the cluster corresponding to the case that $\mathbf{S} = \mathbf{0}$ as cluster 1. The CR network is aware of the parameters for the multivariate Gaussian distribution if and only if $\mathbf{S} = \mathbf{0}$ since the CR network does not know the power attenuation $g_{m,n}$. Therefore, cluster 1 can easily be identified by the mean vector $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ and the covariance matrix $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ while the other clusters should be blindly identified by unsupervised learning.

From now on, we will investigate the application of two representative unsupervised clustering algorithms, i.e., the K-means clustering and the GMM, to CSS. After training by using these clustering algorithms, each time the classifier receives the test energy vector for classification, the classifier finds out which cluster the test energy vector belongs to and classifies it as the channel available class if and only if the test energy vector belongs to cluster 1.

B. K-Means Clustering Algorithm

The unsupervised K-means clustering algorithm partitions a set of the training energy vectors (i.e., $\overline{\mathbf{y}} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(L)}\}$) into K disjoint clusters. Let $\mathcal{C}_k$ denote the set of the training energy vectors that belong to cluster k. Cluster k has a centroid $\boldsymbol{\alpha}_k$. Differently from the ordinary K-means clustering algorithm, we assume that the centroid of cluster 1 is fixed to the mean of $\mathbf{Y}$ conditioned on $\mathbf{S} = \mathbf{0}$, that is, $\boldsymbol{\alpha}_1 = \boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$. For all other clusters, the centroid is defined as the mean of all training energy vectors in $\mathcal{C}_k$ such that $\boldsymbol{\alpha}_k = |\mathcal{C}_k|^{-1} \sum_{\mathbf{y}^{(l)} \in \mathcal{C}_k} \mathbf{y}^{(l)}$, $\forall k = 2, \ldots, K$, where $|\mathcal{X}|$ denotes the number of elements in the set $\mathcal{X}$. The K-means clustering algorithm aims to find out K clusters, $\mathcal{C}_1, \ldots, \mathcal{C}_K$, which minimize the within-cluster sum of squares as follows:

$$\underset{\mathcal{C}_1, \ldots, \mathcal{C}_K}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{\mathbf{y}^{(l)} \in \mathcal{C}_k} \left\| \mathbf{y}^{(l)} - \boldsymbol{\alpha}_k \right\|^2. \tag{14}$$

To find the clusters satisfying (14), we use an iterative suboptimal algorithm presented in **Algorithm 1**.

In **Algorithm 1**, the centroid of cluster 1 is set to $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ in Line 1. The centroids for clusters except for cluster 1 are initialized in Line 2. The iteration begins from Line 3. In Line 4, each training energy vector is assigned to the cluster the centroid of which is closest to the training energy vector. In Line 5, the centroids of clusters except for cluster 1 are updated by taking the mean of all training energy vectors in each cluster. The iterations are repeated until there is no change in the clusters. Finally, we have a suboptimal solution for (14) when the iteration is over. Let $\boldsymbol{\alpha}_k^*$ denote the centroid for cluster k obtained by the K-means clustering.

Algorithm 1 K-Means Clustering Algorithm for CSS

```
1: $\boldsymbol{\alpha}_1 \leftarrow \boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$
2: $\boldsymbol{\alpha}_k$ is initialized, $\forall k = 2, \ldots, K$.
3: while $\mathcal{C}_k$ for some $k$ is changed in the previous iteration do
4:     $\mathcal{C}_k \leftarrow \{\mathbf{y}^{(l)} | \|\mathbf{y}^{(l)} - \boldsymbol{\alpha}_k\| \leq \|\mathbf{y}^{(l)} - \boldsymbol{\alpha}_i\|, \forall i = 1, \ldots, K\}, \forall k = 1, \ldots, K$.
5:     $\boldsymbol{\alpha}_k \leftarrow |\mathcal{C}_k|^{-1} \sum_{\mathbf{y}^{(l)} \in \mathcal{C}_k} \mathbf{y}^{(l)}, \forall k = 2, \ldots, K$.
6: end while
```

After the training is over, the classifier receives the test energy vector $\mathbf{y}^*$ for classification. The classifier determines if the test energy vector belongs to cluster 1 or the other classes, based on the distance from the test energy vector to the centroids. The classifier classifies $\mathbf{y}^*$ as the channel unavailable class (i.e., $\widehat{a} = -1$) if the following condition is met:

$$\frac{\|\mathbf{y}^* - \boldsymbol{\alpha}_1^*\|}{\min_{k=1,\ldots,K} \|\mathbf{y}^* - \boldsymbol{\alpha}_k^*\|} \geq \beta. \tag{15}$$

Otherwise, $\mathbf{y}^*$ is classified as the channel available class (i.e., $\widehat{a} = 1$). The parameter β is the threshold to control the tradeoff between the misdetection and the false alarm probabilities. If β becomes high, $\mathbf{y}^*$ is more likely to be classified as the channel available class, which in turn increases the misdetection probability while decreasing the false alarm probability.

C. Gaussian Mixture Model

A GMM is a weighted sum of multivariate Gaussian probability densities given by

$$f(\mathbf{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} v_k \cdot \phi(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{16}$$

where $\phi(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the Gaussian density such that

$$\phi(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{N/2}|\boldsymbol{\Sigma}_k|^{1/2}} \exp\left\{ -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\}, \tag{17}$$

and $\boldsymbol{\theta}$ is the collection of all parameters for the GMM including v_k, $\boldsymbol{\mu}_k$, and $\boldsymbol{\Sigma}_k$ for all $k = 1, \ldots, K$. The GMM exactly matches our energy vector model where $\mathbf{Y}$ conditioned on $\mathbf{S} = \mathbf{s}$ follows the multivariate Gaussian distribution with the mean vector $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$ and the covariance matrix $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$. Let the kth Gaussian density $\phi(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ in the GMM approximate the density of the energy vectors belonging to cluster k. If cluster k corresponds to the case that $\mathbf{S} = \mathbf{s}$, the parameters for the kth Gaussian density, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and v_k, correspond to $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$, $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$, and $v(\mathbf{s})$ in the energy vector model, respectively. Since cluster 1 corresponds to the case that $\mathbf{S} = \mathbf{0}$, we have $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ and $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$, which are known to the CR network in advance. Moreover, we can restrict $\boldsymbol{\Sigma}_k$ to a diagonal matrix for all k since $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{s}}$ is a diagonal matrix for all $\mathbf{s}$. The rest of the parameters in $\boldsymbol{\theta}$ are unknown and need to be estimated.

The parameters can be estimated by using the maximum-likelihood (ML) estimation given the set of the training energy vectors (i.e., $\overline{\mathbf{y}} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(L)}\}$). The log-likelihood of the set of the training energy vectors can be written as

$$\omega(\overline{\mathbf{y}}|\boldsymbol{\theta}) = \sum_{l=1}^{L} \ln \left(\sum_{k=1}^{K} v_k \cdot \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right). \tag{18}$$

The ML estimator is the parameter that maximizes this log-likelihood function. Unfortunately, direct computation of the ML estimator is not possible due to latent information [17]. However, the parameters that maximize the log-likelihood can be obtained by using the expectation maximization (EM) algorithm [23].

The EM algorithm iteratively updates the parameter $\boldsymbol{\theta}$ by maximizing the following function:

$$\begin{aligned} Q(\boldsymbol{\theta}'|\boldsymbol{\theta}) &= \mathrm{E}\left[\sum_{l=1}^{L} \ln \left(v'_{z^{(l)}} \cdot \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}'_{z^{(l)}}, \boldsymbol{\Sigma}'_{z^{(l)}}) \right) \middle| \overline{\mathbf{y}}, \boldsymbol{\theta} \right] \\ &= \sum_{l=1}^{L} \left\{ \sum_{k=1}^{K} u_k^{(l)} \ln v'_k + \sum_{k=1}^{K} u_k^{(l)} \ln \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}'_k, \boldsymbol{\Sigma}'_k) \right\}, \end{aligned} \tag{19}$$

where $z^{(l)}$ is a random variable which is the index of the cluster to which the lth training energy vector belongs to and $u_k^{(l)}$ is defined as

$$u_k^{(l)} = \Pr[z^{(l)} = k|\overline{\mathbf{y}}, \boldsymbol{\theta}] = \frac{v_k \cdot \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{i=1}^{K} v_i \cdot \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}. \tag{20}$$

Let us define $\boldsymbol{\theta}(j)$ as the estimated parameter at the jth iteration of the EM algorithm. At the jth iteration, the EM algorithm finds $\boldsymbol{\theta}(j+1)$ that satisfies

$$\boldsymbol{\theta}(j+1) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}|\boldsymbol{\theta}(j)). \tag{21}$$

It is known that $\boldsymbol{\theta}(j)$ converges to a local optimal solution over iterations [23].

At each iteration, the EM algorithm first calculates $u_k^{(l)}$ from (20) in the expectation step, and then derives the solution of (21) in the maximization step. The solution of (21) can be evaluated by a basic optimization technique. In **Algorithm 2**, we present the EM algorithm for solving the GMM. In Line 1 of **Algorithm 2**, the mean $\boldsymbol{\mu}_1(1)$ and the covariance $\boldsymbol{\Sigma}_1(1)$ for cluster 1 are set to $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ and $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$, respectively. In Line 2, all other parameters are initialized. In Lines 4–8, the expectation and the maximization steps are repeated until $\boldsymbol{\theta}(j)$ converges. Note that $\boldsymbol{\mu}_1(j)$ and $\boldsymbol{\Sigma}_1(j)$ are not updated in the maximization step since $\boldsymbol{\mu}_1(j)$ and $\boldsymbol{\Sigma}_1(j)$ are fixed to $\boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ and $\boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$, respectively. Let $\boldsymbol{\theta}^*$ denote the parameter obtained after the EM algorithm is over.

After obtaining the optimal parameter $\boldsymbol{\theta}^*$, the classifier receives the test energy vector $\mathbf{y}^*$ for classification. The classifier determines whether the test energy vector $\mathbf{y}^*$ belongs to cluster 1 or other clusters. The log-likelihood that $\mathbf{y}^*$ belongs to cluster 1 is $\ln(v_1^* \cdot \phi(\mathbf{y}^*|\boldsymbol{\mu}_1^*, \boldsymbol{\Sigma}_1^*))$. Similarly, the log-likelihood that $\mathbf{y}^*$ belongs to the clusters other than cluster 1 is $\ln(\sum_{k=2}^{K} v_k^* \cdot \phi(\mathbf{y}^*|\boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*))$. Therefore, $\mathbf{y}^*$ is classified as the channel unavailable class (i.e., $\widehat{a} = -1$) if and only if

$$\ln\left(\sum_{k=2}^{K} v_k^* \cdot \phi(\mathbf{y}^*|\boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*) \right) - \ln(v_1^* \cdot \phi(\mathbf{y}^*|\boldsymbol{\mu}_1^*, \boldsymbol{\Sigma}_1^*)) \geq \delta, \tag{22}$$

for a given threshold δ. We can decrease the false alarm probability at the expense of misdetection probability by increasing δ since $\mathbf{y}^*$ is more likely to be classified as the channel available class if the value of δ is high.

Algorithm 2 EM Algorithm for GMM

```
1: $\boldsymbol{\mu}_1(1) \leftarrow \boldsymbol{\mu}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$ and $\boldsymbol{\Sigma}_1(1) \leftarrow \boldsymbol{\Sigma}_{\mathbf{Y}|\mathbf{S}=\mathbf{0}}$
2: Initialize $v_k(1)$ for $k = 1, \ldots, K$ and $\boldsymbol{\mu}_k(1)$ and $\boldsymbol{\Sigma}_k(1)$
   for $k = 2, \ldots, K$.
3: $j \leftarrow 1$
4: repeat
5:     Expectation Step
           $u_k^{(l)} \leftarrow \frac{v_k(j) \cdot \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}_k(j), \boldsymbol{\Sigma}_k(j))}{\sum_{i=1}^{K} v_i(j) \cdot \phi(\mathbf{y}^{(l)}|\boldsymbol{\mu}_i(j), \boldsymbol{\Sigma}_i(j))}$,
           for $l = 1, \ldots, L$ and $k = 1, \ldots, K$.
6:     Maximization Step
           $v_k(j+1) \leftarrow \frac{\sum_{l=1}^{L} u_k^{(l)}}{L}$,    for $k = 1, \ldots, K$.
           $\boldsymbol{\mu}_k(j+1) \leftarrow \frac{\sum_{l=1}^{L} u_k^{(l)} \mathbf{y}^{(l)}}{\sum_{l=1}^{L} u_k^{(l)}}$,    for $k = 2, \ldots, K$.
           $\boldsymbol{\Sigma}_k(j+1) \leftarrow \frac{\sum_{l=1}^{L} u_k^{(l)} \{\mathrm{diag}(\mathbf{y}^{(l)} - \boldsymbol{\mu}_k(j+1))\}^2}{\sum_{l=1}^{L} u_k^{(l)}}$,
           for $k = 2, \ldots, K$.
7:     $j \leftarrow j + 1$
8: until $\boldsymbol{\theta}(j)$ converges.
```

V. Supervised Learning for Cooperative Spectrum Sensing

A. *Motivation for Supervised Learning*

In this section, we propose application of the supervised learning techniques, i.e., the support vector machine (SVM) and the weighted K-nearest-neighbor (KNN), to CSS in the CR networks. The main difference of the supervised learning from the unsupervised learning is that each training energy vector $\mathbf{y}^{(l)}$ is labeled with the corresponding channel availability $a^{(l)}$. Therefore, to implement supervised learning for CSS in practice, the PU should occasionally inform the CR network of the channel availabilities for some training energy vectors for the purpose of training. Since supervised learning can work with the explicit help from the PU, it is more difficult to implement than the unsupervised learning. However, supervised learning tends to show better performance due to the extra information on the channel availability. We assume that the training energy vectors (i.e., $\overline{\mathbf{y}} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(L)}\}$) and the channel availability corresponding to each training energy vector (i.e., $\overline{a} = \{a^{(1)}, \ldots, a^{(L)}\}$) are fed into the classifier for training.

B. *Support Vector Machine*

The SVM tries to find a linearly separable hyperplane, with the help of support vectors (i.e., energy vectors that lie closest to the decision surface), by maximizing the margin of the classifier while minimizing the sum of errors. However, the training energy vectors may not be linearly separable. Therefore, we try to map the training energy vectors into a higher dimensional feature space by a non-linear mapping function, denoted by ϕ, to make the training samples linearly separable [16], [24]. Hence, the classifier should satisfy the following condition for all $l = 1, \ldots, L$:

$$\begin{aligned} &\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0 \geq 1, \text{ if } a^{(l)} = 1, \\ &\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0 \leq -1, \text{ if } a^{(l)} = -1, \end{aligned} \tag{23}$$

where $\mathbf{w}$ is the weighting vector and w_0 is the bias. The bias is used for shifting the hyperplane away from the origin.

Although we map the training energy vectors into a higher dimensional feature space, practically we cannot achieve a perfect linearly separable hyperplane that satisfies the condition in (23) for each $\mathbf{y}^{(l)}$. Hence, we modify condition (23) by introducing a slack variable $\delta^{(l)}$ for possible classification errors as follows:

$$a^{(l)}[\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0] \geq 1 - \delta^{(l)}, \tag{24}$$

where $\delta^{(l)} \geq 0$ for $l = 1, \ldots, L$. For marginal classification errors, the slack variable lies in $0 \leq \delta^{(l)} \leq 1$, whereas $\delta^{(l)} > 1$ for misclassification. Hence, the optimization problem for maximizing the margin of classifier while minimizing the sum of errors can be written as

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \xi \sum_{l=1}^{L} I_{\{\delta^{(l)} > 1\}} && (25) \\ \text{subject to} \quad & a^{(l)}[\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0] \geq 1 - \delta^{(l)}, && (26) \\ & \quad \text{for } l = 1, \ldots, L, && (27) \\ & \delta^{(l)} \geq 0, \quad \text{for } l = 1, \ldots, L, && (28) \end{aligned}$$

where $\|\mathbf{w}\|^2 = \mathbf{w} \cdot \mathbf{w}$, ξ is a soft margin constant [16], and $I_{\{X\}}$ is the indicator function which is one if X is true; and is zero, otherwise.

The optimization problem defined in (25)–(28) is non-convex due to $I_{\{\delta^{(l)} > 1\}}$ in the objective function. Since $\delta^{(l)} > 1$ for misclassification, $\sum_{l=1}^{L} \delta^{(l)}$ gives a bound on the number of the misclassified training energy vectors. Therefore, $\sum_{l=1}^{L} \delta^{(l)}$ can be used to measure the number of the training energy vectors which are misclassified by the decision surface $\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0 = 0$ as well as the number of the training energy vectors that are correctly classified but they lie in the slab $-1 < \mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0 < 1$. Hence, we can rewrite the optimization problem as a convex optimization problem as follows:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \xi \sum_{l=1}^{L} \delta^{(l)} && (29) \\ \text{subject to} \quad & a^{(l)}[\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0] \geq 1 - \delta^{(l)}, && (30) \\ & \quad \text{for } l = 1, \ldots, L, && (31) \\ & \delta^{(l)} \geq 0, \quad \text{for } l = 1, \ldots, L. && (32) \end{aligned}$$

The Lagrangian of (29)–(32) can be written as

$$\begin{aligned} \Lambda(\mathbf{w}, w_0, \boldsymbol{\delta}; \boldsymbol{\lambda}, \boldsymbol{\gamma}) = {} & \frac{1}{2}\|\mathbf{w}\|^2 + \xi \sum_{l=1}^{L} \delta^{(l)} \\ & - \sum_{l=1}^{L} \lambda^{(l)} \big\{ a^{(l)}[\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0] \\ & \qquad - 1 + \delta^{(l)} \big\} \\ & - \sum_{l=1}^{L} \gamma^{(l)} \delta^{(l)}, \end{aligned} \tag{33}$$

where $\lambda^{(l)}$ and $\gamma^{(l)}$ are Lagrangian multipliers. By applying the Karush-Kuhn-Tucker (KKT) conditions, we can obtain

$$\mathbf{w} = \sum_{l=1}^{L} \lambda^{(l)} a^{(l)} \phi(\mathbf{y}^{(l)}) \tag{34}$$

$$\sum_{l=1}^{L} \lambda^{(l)} a^{(l)} = 0 \tag{35}$$

$$\lambda^{(l)} = \xi - \gamma^{(l)}. \tag{36}$$

It is noticeable that $\gamma^{(l)} \geq 0$ and $0 \leq \lambda^{(l)} \leq \xi$. The vector of $\lambda^{(l)}$'s is known as a support vector. Hence, we can obtain the dual problem in terms of the support vector as follows:

$$\text{maximize} \quad \sum_{l=1}^{L} \lambda^{(l)} - \frac{1}{2} \sum_{i=1}^{L} \sum_{j=1}^{L} \lambda^{(i)} \lambda^{(j)} a^{(i)} a^{(j)} \{\phi(\mathbf{y}^{(i)}) \cdot \phi(\mathbf{y}^{(j)})\} \tag{37}$$

$$\begin{aligned} \text{subject to} \quad & \sum_{l=1}^{L} \lambda^{(l)} a^{(l)} = 0 && (38) \\ & 0 \leq \lambda^{(l)} \leq \xi, \quad \text{for } l = 1, \ldots, L. && (39) \end{aligned}$$

The KKT conditions uniquely characterize the solution of the primal problem as (34)–(36), and the dual problem as two active constraints $\lambda^{(l)} \big\{ a^{(l)}[\mathbf{w} \cdot \phi(\mathbf{y}^{(l)}) + w_0] - 1 + \delta^{(l)} \big\} = 0$ and $\gamma^{(l)} \delta^{(l)} = (\xi - \lambda^{(l)}) \delta^{(l)} = 0$. We can solve the optimization problem in (37)–(39) by using standard techniques to solve a quadratic program. Let $\widetilde{\lambda}^{(l)}$ denote the solution of (37)–(39).

Finally, the nonlinear decision function can be obtained as

$$d(\mathbf{x}) = \operatorname{sgn}\left(\sum_{l=1}^{L} \widetilde{\lambda}^{(l)} a^{(l)} \kappa(\mathbf{x}, \mathbf{y}^{(l)}) + w_0 \right), \tag{40}$$

where sgn is the sign function and $\kappa(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$ is the kernel function. Some of the commonly used kernel functions are linear, polynomial, and Gaussian radial basis functions [16]. After the classifier obtains the decision function, the classifier can categorize the test energy vector $\mathbf{y}^*$ as

$$\widehat{a} = d(\mathbf{y}^*). \tag{41}$$

Note that the bias w_0 can be derived by solving the optimization problem (29)–(32) after finding the optimal $\mathbf{w}$. However, in this paper, we adjust w_0 to control the tradeoff between the false alarm and the misdetection probabilities.

Remark: If L denotes the number of training energy vectors and L_s denotes the total number of support vectors, then the expected error rate is bounded by $\mathrm{E}[error] \leq \mathrm{E}[L_s]/L$ [17]. The expectation is taken over the training set generated from the distribution of energy vectors. Note that this bound is independent of the dimension of the feature space that is determined by ϕ. Furthermore, if we can map the original feature space to the higher dimensional feature space (i.e., the feature space of ϕ) so that it separates the training energy vectors by using a small number of support vectors, the expected error becomes lower. Hence, it is important to select a kernel function which reduces the number of support vectors.

C. Weighted K-Nearest-Neighbor

The weighted K-nearest-neighbor (KNN) is a classification technique based on the majority voting of neighbors. For a given test energy vector $\mathbf{y}^*$, the KNN classifier finds K neighboring training energy vectors among $\overline{\mathbf{y}} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(L)}\}$ based on a particular distance measure. We define $\Delta(\mathbf{x}, \mathbf{y})$ as a distance between the energy vectors $\mathbf{x}$ and $\mathbf{y}$. To find the neighboring training energy vectors, the KNN classifier calculates $\Delta(\mathbf{y}^*, \mathbf{y}^{(l)})$ for all $\mathbf{y}^{(l)}$'s and sorts the training energy vectors in the ascending order of $\Delta(\mathbf{y}^*, \mathbf{y}^{(l)})$. Then, the KNN classifier selects the first K training energy vectors as neighbors. Let $\Phi(\mathbf{y}^*, \overline{\mathbf{y}})$ denote the set of neighbors of $\mathbf{y}^*$ among $\overline{\mathbf{y}}$. To determine the class of $\mathbf{y}^*$, we count the number of neighbors that belong to the channel available class and the channel unavailable class, respectively. The number of neighbors in the channel available class ($a = 1$) and the channel unavailable class ($a = -1$) is defined as

$$\nu(a; \mathbf{y}^*, \overline{\mathbf{y}}) = |\{l = 1, \ldots, L | a^{(l)} = a, y^{(l)} \in \Phi(\mathbf{y}^*, \overline{\mathbf{y}})\}|. \tag{42}$$

The KNN classifier categorizes $\mathbf{y}^*$ as the channel unavailable class (i.e., $\widehat{a} = -1$) if and only if

$$\frac{\nu(-1; \mathbf{y}^*, \overline{\mathbf{y}})}{\nu(1; \mathbf{y}^*, \overline{\mathbf{y}})} \geq \varphi, \tag{43}$$

where φ is a constant that controls the tradeoff between the false alarm and the misdetection probabilities.

The distance $\Delta(\mathbf{x}, \mathbf{y})$ can be calculated in various ways. In this paper, we adopt a weighted distance measure where each component of the energy vector is weighted by a certain weight factor. The weight factor for the nth component of the energy vector is denoted by ω_n. To calculate ω_n, we draw an ROC curve by using the nth components of the training energy vectors (i.e., $y_n^{(1)}, \ldots, y_n^{(L)}$) and the corresponding channel availabilities (i.e., $a_n^{(1)}, \ldots, a_n^{(L)}$). Then, the weight factor ω_n is equal to the area-under-the-ROC-curve (AUC) of the nth components of the training energy vectors. If we consider the squared Euclidean distance, the distance measure is given as

$$\Delta(\mathbf{x}, \mathbf{y}) = \sum_{n=1}^{N} \{\omega_n (x_n - y_n)\}^2. \tag{44}$$

On the other hand, if we adopt the city block distance, we have

$$\Delta(\mathbf{x}, \mathbf{y}) = \sum_{n=1}^{N} |\omega_n (x_n - y_n)|. \tag{45}$$

Remark: If the number of neighbors (i.e., K) is fixed and the number of training energy vectors approaches infinity, then all the K neighbors converge to $\mathbf{y}^*$. Hence, the label of each of the K-nearest-neighbors is a random variable which takes the value of a with probability $\Pr[A = a | \mathbf{Y} = \mathbf{y}^*]$ [17]. When the number of neighbors, K, increases, the proportion of each label of the neighbors approaches the Bayesian a posteriori probability. Hence, the error probability of the KNN classifier becomes closer to that of the Bayesian classifier with increasing K. In practice, we can have only a limited number of training energy vectors. On the other hand, we want to reduce errors by increasing the number of training energy vectors. This tradeoff forces us to select a reasonable value for K.

VI. Performance Evaluation

A. Parameters

In this study, unless otherwise specified, we consider that the SUs participating in cooperative spectrum sensing (CSS) are located in a 5-by-5 (25 SUs) grid topology in a 4000 m × 4000 m area as shown in Fig. 4. The values of important simulation parameters are as follows: the bandwidth w is 5 MHz, the sensing duration τ is 100 μs, the noise spectral density η is -174 dBm, and the path-loss exponent α is 4. We assume that the shadow fading and the multi-path fading components are fixed as $\psi_{m,n} = 1$ and $\nu_{m,n} = 1$. The transmit power of each PU is 200 mW. We consider two PUs having fixed locations with coordinates (500 m, 500 m) and (-1500 m, 0 m). The probability that a PU is in the active state is 0.5 and the state of each PU is independent of that of the other PU. The proposed algorithms are implemented by using Matlab 7.10.0 (R2010a) in a 64-bit computer with a core i7 processor (clock speed of 2.8 GHz) and 4 GB RAM.

B. Training Duration for Different Classifiers

The average training durations for different classifiers according to the size of training energy vectors are shown in Table I. The GMM shows relatively high training duration (1.12796 seconds for 1000 samples) among the unsupervised classifiers whereas the supervised SVM classifier with the polynomial kernel takes the highest training duration (1.65817 seconds for 1000 samples) among all classifiers. The average training time for the KNN classifier is measured as the uploading time of the training energy vectors to the classifier and it is approximately 50 μs for 1000 energy vectors. Hence, the KNN classification has the capability of changing the training energy vectors more promptly as compared to all other classifiers.

C. Average Classification Delay for Different Classifiers

Table II shows the time taken for deciding the channel availability for different classifiers. The classification delay of Fisher linear discriminant, K-means clustering, and GMM classifiers does not change with different batch of training energy vectors. More specifically, the number of decision parameters does not change with the number of training energy vectors even though the values of the decision parameters

TABLE I
AVERAGE TRAINING DURATION (IN SECONDS) FOR DIFFERENT CLASSIFIERS (5×5 SUS)

Classification Methods	Number of Training Samples					
	100	200	300	400	500	1000
Fisher	0.01001	0.01035	0.01067	0.01091	0.01144	0.01346
K-means	0.09202	0.09319	0.09363	0.09455	0.09536	0.11704
GMM	0.0309	0.06621	0.17373	0.24281	0.35527	1.12796
SVM-Linear	0.01114	0.01426	0.01792	0.02114	0.0268	0.06289
SVM-Poly.	0.04986	0.31983	0.46806	0.85701	1.03886	1.65817

TABLE II
AVERAGE CLASSIFICATION DELAY (IN SECONDS) FOR DIFFERENT CLASSIFIERS (5×5 SUS)

Classification Methods	Number of Training Samples					
	100	200	300	400	500	1000
Fisher	5.3×10^{-6}	5.3×10^{-6}	5.3×10^{-6}	5.3×10^{-6}	5.3×10^{-6}	5.3×10^{-6}
K-means	1.9×10^{-5}	1.9×10^{-5}	1.9×10^{-5}	1.9×10^{-5}	1.9×10^{-5}	1.9×10^{-5}
GMM	3.8×10^{-5}	3.8×10^{-5}	3.8×10^{-5}	3.8×10^{-5}	3.8×10^{-5}	3.8×10^{-5}
SVM-Linear	1.92×10^{-5}	3.24×10^{-5}	3.87×10^{-5}	4.45×10^{-5}	4.86×10^{-5}	5.67×10^{-5}
SVM-Polynomial	1.02×10^{-5}	1.12×10^{-5}	1.25×10^{-5}	1.32×10^{-5}	1.53×10^{-5}	2.81×10^{-5}
KNN-Euclidean	4.68×10^{-5}	5.82×10^{-5}	7.72×10^{-5}	8.73×10^{-5}	1.17×10^{-4}	2.98×10^{-4}
KNN-Cityblock	4.62×10^{-5}	5.73×10^{-5}	7.63×10^{-5}	8.57×10^{-5}	1.16×10^{-4}	2.84×10^{-4}

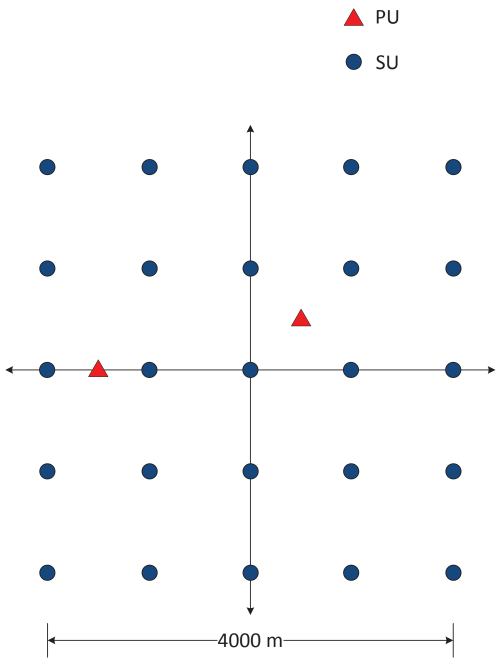


Fig. 4. The CR network topology used for simulation.

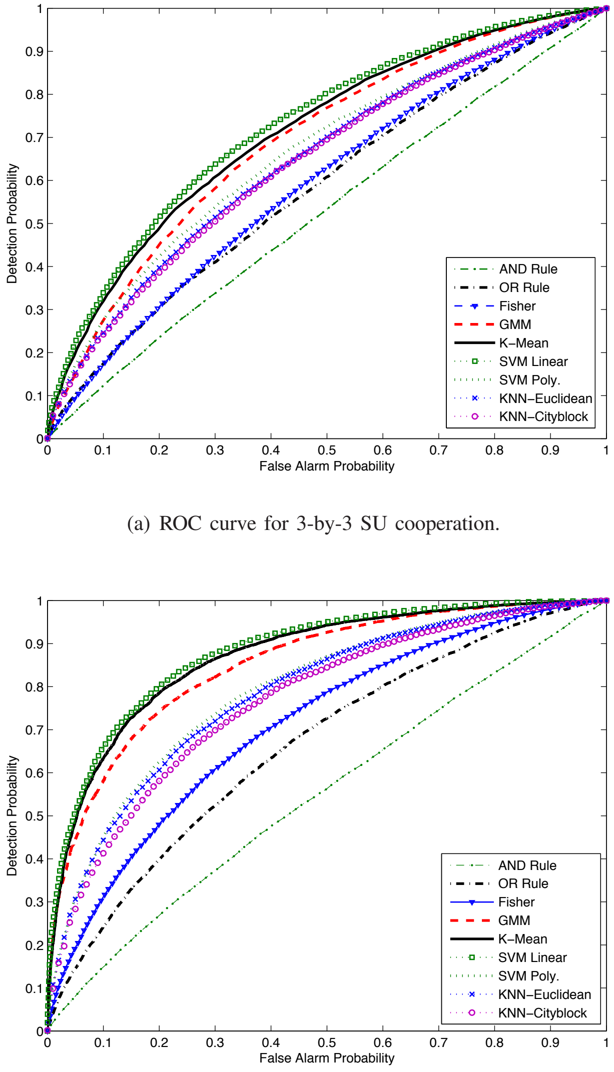
change slightly with the number of training energy vectors. Table II clearly shows that the K-means classifier has the capability to detect channel availability more promptly in comparison to the other unsupervised learning approach (i.e., GMM). For supervised learning, the Fisher linear discriminant shows the lowest classification delay. It is important to note that, for the KNN classifier, the classification delay is relatively high even though its training time is found to be the lowest.

D. Detection Probability for Different Classifiers

Fig. 5 compares the performance of different proposed CSS schemes in terms of receiver operating characteristic (ROC) curves for different sets of cooperating SUs when there is only a single PU at (500 m, 500 m). In particular, Figs. 5(a) and 5(b) show the ROC curves when 3 × 3 SUs (i.e., 9 SUs) and 5 × 5 SUs (i.e., 25 SUs) participate in CSS, respectively. These figures clearly reveal that the performances of the proposed classifiers improve with the increasing number of SUs. It is important to notice that all the proposed CSS schemes outperform the existing CSS techniques such as those based on the Fisher linear discriminant analysis, AND-rule, and OR-rule. Fig. 5 depicts that the SVM with the linear kernel outperforms the other CSS schemes. The SVM-Linear classifier achieves high detection probability by mapping a feature space to a higher dimension with the help of the linear kernel. Interestingly, Fig. 5 clearly shows that the unsupervised K-means classifier achieves the performance comparable to the SVM-Linear classifier. The simple weighted KNN scheme achieves comparatively higher detection probability than the existing CSS techniques due to the exploitation of localized information.

Fig. 6 shows the performance of different CSS schemes in terms of the ROC curve when there are two PUs at (500 m, 500 m) and (−1500 m, 0 m). This figure clearly depicts that the SVM classifier with the linear kernel outperforms the other supervised and unsupervised CSS schemes. The computational complexity of the SVM-Linear classifier can be compensated by its high detection capability and comparatively low training and classification delay. Hence, the SVM classifier with the linear kernel is well-suited for CSS requiring high accuracy. Further, this figure reveals that the K-means clustering scheme outperforms the other unsupervised CSS schemes even in the multiple PU case.

In Fig. 7, the detection probabilities for the different CSS schemes are plotted against the transmission power of a PU. The results are obtained for a target false alarm probability



(a) ROC curve for 3-by-3 SU cooperation.

(b) ROC curve for 5-by-5 SU cooperation.

Fig. 5. The ROC curves when a single PU is present. We use 500 training energy vectors to train each classifier.

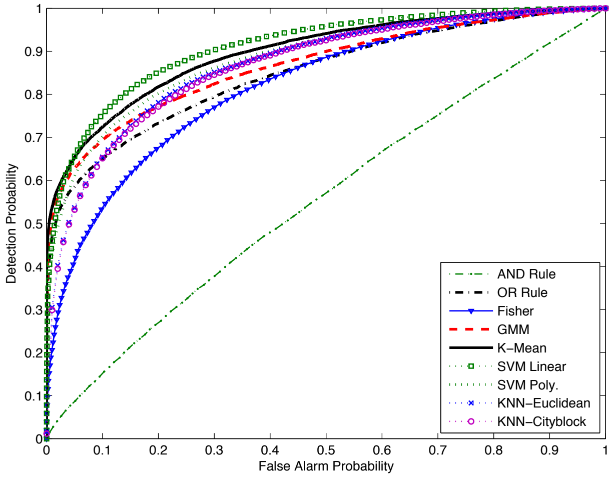


Fig. 6. The ROC curves when there are two PUs. We use 500 training energy vectors to train each classifier.

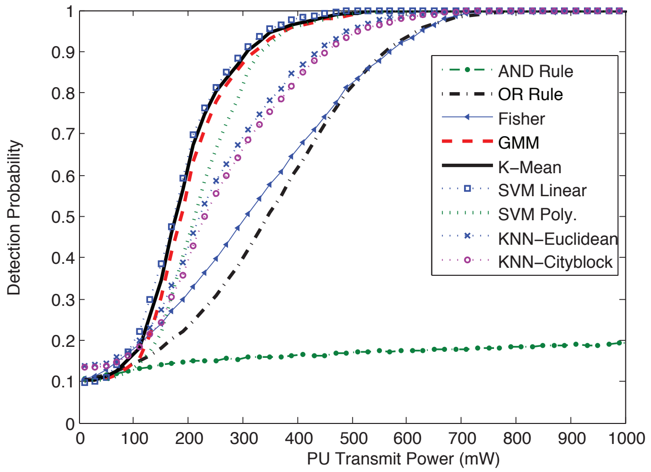


Fig. 7. The detection probability according to the transmission power of a PU when the false alarm probability is 0.1 and there are 25 (5×5) SUs.

of 0.1 when there is a single PU at (500 m, 500 m). This figure shows that all the proposed CSS schemes outperform the existing CSS schemes in all range of the transmission power of a PU. Especially, it is worth noting that the unsupervised learning schemes (i.e., K-means and GMM) achieve performance which is comparable to that of the supervised learning schemes even when the transmission power of a PU is very low.

E. Summary of Results

The main results of the analysis can be summarized as follows:

- The unsupervised K-means clustering is a promising approach for CSS due to its higher PU detection capability and lower training and classification delay. Moreover, its detection probability is very close to the best performing classifier, i.e., the SVM-Linear classifier.
- Compared to all other classifiers, the supervised SVM classifier with the linear kernel performs well in terms of the detection probability. The computational complexity of the SVM-Linear classifier is compensated by its higher detection capability and comparatively lower training and classification delay.
- In terms of updating the training energy vectors on-the-fly, the KNN performs extremely well (training duration for 1000 sample is approximately 50 μs). However, its classification delay is relatively higher than other classifiers.

A qualitative comparison among the different classifiers is shown in Table III.

VII. Conclusion

In this paper, we have designed cooperative spectrum sensing (CSS) mechanisms for cognitive radio (CR) networks based on unsupervised and supervised learning techniques.

TABLE III
COMPARISON AMONG DIFFERENT CSS CLASSIFIERS

Classification Methods	Training Duration	Classification Delay	ROC Performance
Fisher Linear Discriminant	Low	Normal	Low
K-Means	Normal	Low	High
GMM	High	Low	High
SVM-Linear	Low	Normal	High
SVM-Poly.	High	Low	Normal
KNN-Euclidean	Low	High	Normal
KNN-Cityblock	Low	High	Normal

We have proposed to use unsupervised classifiers such as K-means clustering and Gaussian mixture model (GMM) for CSS, whereas the support vector machine (SVM) and weighted K-nearest-neighbor classifiers have been proposed for CSS under supervised learning. The received energy level measured at the secondary users (SUs) are considered as a feature for determining the channel availability. We quantify the performance of the classifiers in terms of the training duration, the classification delay, and the ROC curves. The proposed SVM classifier achieves the highest detection performance compared to the other CSS algorithms by mapping the feature space into the higher dimensional space with the help of kernel functions, namely, linear kernel and polynomial kernel functions. Further, the unsupervised K-means clustering scheme achieves the performance very close to the supervised SVM-Linear classifier in terms of the ROC performance. In particular, the weighted KNN cognitive classifier requires very small amount of time for training the classifier. Hence, the weighted KNN classifier is well suited for CSS which requires to update training energy vectors on-the-fly.

More importantly, the unsupervised K-means clustering and the supervised SVM with the linear kernel are promising approaches for CSS in CR networks. In practical implementation, it is important to obtain accurate training energy vectors since inaccurate training of the classifier results in inaccurate decision. The proposed CSS approaches can be further improved to gradually train the classifiers by using the energy vectors obtained one-by-one. This facilitates the classifiers to adapt to the varying environment without training all over again.

REFERENCES

[1] E. Hossain and V. K. Bhargava, *Cognitive Wireless Communication Networks*. Springer, 2007.

[2] H. Huang, Z. Zhang, P. Cheng, and P. Qiu, "Opportunistic spectrum access in cognitive radio system employing cooperative spectrum sensing," in Proc. of *IEEE Vehicular Technology Conference (VTC'09- Spring 2009)*, pp. 1–5, 26-29 April 2009.

[3] T. Yucek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *IEEE Commun. Surveys Tutorials*, vol. 11, no. 1, pp. 116–130, First Quarter 2009.

[4] C. Li and C. Li, "Opportunistic spectrum access in cognitive radio networks," *IEEE International Joint Conference on Neural Networks (IJCNN'08)*, pp. 3412–3415, 1-8 June 2008.

[5] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Physical Communications*, vol. 4, no. 1, pp. 40–62, March 2011.

[6] D. Cabric, S. M. Mishra, and R. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in Proc. of *38rd Asilomar Conf. on Signals, Systems, and Computers, Pacific Grove*, CA, Nov. 2004.

[7] A. Sahai, R. Tandra, S. M. Mishra, and N. Hoven, "Fundamental design tradeoffs in cognitive radio systems," in Proc. of *TAPAS'06*, Boston, MA, Aug. 2006.

[8] A. Ghasemi and E. S. Sousa, "Spectrum sensing in cognitive radio networks: The cooperation-processing tradeoff," *Wireless Commun. and Mobile Comput.*, vol. 7, no. 9, pp. 1049–1060, Nov. 2007.

[9] E. C. Peh, Y.-C. Liang, Y. L. Guan, and Y. Zeng, "Optimization of cooperative sensing in cognitive radio networks: A sensing-throughput tradeoff view," *IEEE Trans. Veh. Technol.*, vol. 58, no. 9, pp. 5294–5299, Nov. 2009.

[10] J. Unnikrishnan and V. V. Veeravalli, "Cooperative sensing for primary detection in cognitive radio," *IEEE J. Sel. Topics Signal Process.*, vol. 2, no. 1, pp. 18–27, Feb. 2008.

[11] J. Ma, G. Zhao, and Y. Li, "Soft combination and detection for cooperative spectrum sensing in cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 11, pp. 4502–4507, Dec. 2008.

[12] G. Ding, Q. Wu, J. Wang, and X. Zhang, "Joint cooperative spectrum sensing and resource scheduling for cognitive radio networks with soft sensing information," *IEEE Youth Conference on Information Computing and Telecommunications (YC-ICT)*, pp. 291–294, 28-30 Nov. 2010.

[13] Z. Quan, W. Ma, S. Cui, and A. H. Sayed, "Optimal linear fusion for distributed detection via semidefinite programming," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2431–2436, Apr. 2010.

[14] G. Ganesan and Y. G. Li, "Cooperative spectrum sensing in cognitive radio - part I: Two user networks," *IEEE Trans. Wireless Commun.*, vol. 6, no. 6, pp. 2204–2213, Jun. 2007.

[15] W. Zhang and K. B. Letaief, "Cooperative spectrum sensing with transmit and relay diversity in cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4761–4766, Dec. 2008.

[16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning J.*, vol. 20, no. 3, pp. 273–297, 1995.

[17] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. 2nd Edition, Wiley, New York, 2001.

[18] A. Agostini and E. Celaya, "Reinforcement learning with a Gaussian mixture model," in Proc. *Int. Joint Conf. on Neural Networks (IJCNN'10)*, Barcelona, Spain, pp. 3485–3492, 2010.

[19] O. L. Mangasarian and D. R. Musicant, "Active support vector machine classification," in *Advances in Neural Information Processing Systems 13*, Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, Editors, pp. 577–583, MIT Press, 2001.

[20] A. Shilton, M. Palaniswami, D. Ralph, and A. C. Tsoi, "Incremental training in support vector machines," *IEEE Trans. Neural Netw.*, vol. 16, pp. 114–131, 2005

[21] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in *Advances in Neural Information Processing Systems*, vol. 13, T. K. Leen, T. G. Dietterich, and V. Tresp, Editors, Cambridge, MA: MIT Press, 2001.

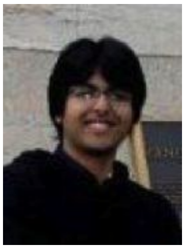
[22] K. W. Choi, E. Hossain, and D. I. Kim, "Cooperative spectrum sensing under a random geometric primary user network model," *IEEE Trans. Wireless Commun.*, vol. 10, no. 6, pp. 1932–1944, June 2011.

[23] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[24] J. S. Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK, 2004.

Karaputugala G. Madushan Thilina (S'11) received the B.Sc. degree from University of Ruhuna, Sri Lanka in 2006 and the M.Eng. degree from Asian Institute of Technology, Thailand, in 2010. He is currently working toward a Ph.D. degree at the Department of Electrical and Computer Engineering, University of Manitoba, Canada. His primary research interest includes design and analysis of new transmission strategies for relay networks, resource allocation for cooperative/cognitive relay networks, and cooperative spectrum sensing in cognitive radio networks. He received AIT fellowship and Government of Finland Scholarships during his master degree. He is currently a recipient of the University of Manitoba Graduate Fellowship (UMGF). Madushan has served as a technical program committee member for several IEEE conferences.

Kae Won Choi (M'08) received the B.S. degree in civil, urban, and geosystem engineering in 2001, and the M.S. and Ph.D. degrees in electrical engineering and computer science in 2003 and 2007, respectively, all from Seoul National University, Seoul, Korea. From 2008 to 2009, he was with Telecommunication Business of Samsung Electronics Co., Ltd., Korea. From 2009 to 2010, he was a Postdoctoral Researcher with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada. In 2010, he joined the faculty at Seoul National University of Science and Technology, Korea, where he is currently an Assistant Professor in the Department of Computer Science and Engineering. His research interests include machine-to-machine communication, device-to-device communication, cognitive radio, radio resource management, and wireless network optimization.



Nazmus Saquib received his M.Sc. degree in Electrical and Computer Engineering from the University of Manitoba, Winnipeg, MB, Canada, in 2013. He received his B.Sc. degree in Electronics and Communication Engineering from BRAC University, Bangladesh, in 2008. He was awarded the University of Manitoba Graduate Fellowship and Clarence Bogardus Sharp Memorial Scholarship during his M.Sc. For academic excellence in undergraduate studies, Saquib won the Vice Chancellor's Gold Medal from BRAC University. His research interests include radio resource management in multi-tier cellular wireless networks and cognitive radio networks.

Ekram Hossain (S'98-M'01-SM'06) is a Professor in the Department of Electrical and Computer Engineering at University of Manitoba, Winnipeg, Canada. He received his Ph.D. in Electrical Engineering from University of Victoria, Canada, in 2001. Dr. Hossain's current research interests include design, analysis, and optimization of wireless/mobile communications networks, cognitive radio systems, and network economics. He has authored/edited several books in these areas (http://home.cc.umanitoba.ca/∼hossaina). Dr. Hossain serves as the Editor-in-Chief for the *IEEE Communications Surveys and Tutorials* and an Editor for the *IEEE Journal on Selected Areas in Communications - Cognitive Radio Series* and *IEEE Wireless Communications*. Previously, he served as the Area Editor for the *IEEE Transactions on Wireless Communications* in the area of "Resource Management and Multiple Access" from 2009-2011 and an Editor for the *IEEE Transactions on Mobile Computing* from 2007-2012. Dr. Hossain has won several research awards including the University of Manitoba Merit Award in 2010 (for Research and Scholarly Activities), the 2011 IEEE Communications Society Fred Ellersick Prize Paper Award, and the IEEE Wireless Communications and Networking Conference 2012 (WCNC'12) Best Paper Award. He is a Distinguished Lecturer of the IEEE Communications Society for the term 2012-2013. Dr. Hossain is a registered Professional Engineer in the province of Manitoba, Canada.