Machine Learning Approaches for Spectrum Management in Cognitive Radio Networks

Ahmed Mohammed Mikaeil

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.74599

Abstract

Cognitive radio (CR) provides a better way for utilization of spectrum resource by introducing an opportunistic usage of the frequency bands that are not heavily occupied by a licensed spectrum user or a primary user (PU). In cognitive radio, the detection and estimation of PU channel availability (unoccupied spectrum) are the key challenges that need to be overcome in order to prevent the interference with licensed spectrum user and improve spectrum resource utilization efficiency. This chapter focuses on developing new ways for detecting and estimating primary user channel availability based on machine-learning (ML) techniques.

Keywords: machine learning, spectrum sensing, spectrum management, channel state estimation, cognitive radio

1. Introduction

In this chapter, we study the problem of detection of unoccupied primary user spectrum (i.e., spectrum hole). We also introduce the methods for estimating the time when primary user channel state is available, so that the secondary spectrum user can adjust their transmission strategies accordingly.

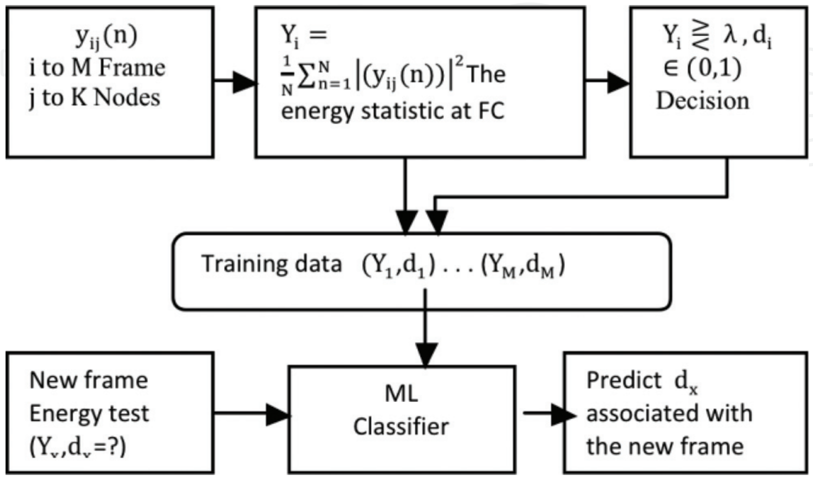
The chapter is organized in two parts. The first part of the chapter focuses on the problem of detecting the unoccupied spectrum left by the primary user. In this part, we introduce the usage of machine-learning (ML) techniques as a fusion algorithm in cooperative spectrum sensing based on energy detector [1, 2]. In particular, we train a machine-learning classifier (i.e., K-nearest neighbor (KNN), support vector machine (SVM), Naive Bayes (NB), and Decision tree (DT)) over a set containing energy test statistics of PU channel frames along with their corresponding decisions about the presence or absence of PU transmission in the channel. Then, we use the trained classifier to predict the decisions for newly unseen PU channel frames [3]. The second part focuses on estimating the near future of PU channel state. In the literature, there are many proposals that have studied the problem of estimating PU channel state in cognitive radio (CR) [4–6]. However, most of these studies focused on predicting PU channel state in frequency domain by converting the received digital signals into frequency domain using fast Fourier transform (FFT). This increases the system complexity due to the FFT computations process. In the second part of the chapter, we introduce a new time-domain approach for PU channel state prediction based on time series prediction with some machine-learning prediction model. In particular, a time series is used to capture PU channel state detection sequence (PU channel "idle" or "occupied") in time domain. Then, prediction models such as the hidden Markov model (HMM) and Markov switching model (MSM) are used to predict the behavior of the time series that used capture PU channel state [7].

2. Machine-learning fusion-based cooperative spectrum sensing

In this part, we, first, define the system model for energy detection-based spectrum sensing; then, we present the method of calculating the thresholds for energy detector with different fusion rules. Second, we formulate a machine-learning classification problem and present four machine-learning classifiers to solve it. Then, we evaluate the performance of these classifiers with simulation experiments.

2.1. Energy detection-based cooperative spectrum sensing

Figure 1 shows a block diagram of the system model used for energy detection cooperative spectrum sensing based on machine-learning fusion rule. In this model, we consider a cooperative CR network with K cooperative nodes. Each node uses N samples for energy detection,



$y_{ij}(n)$
i to M Frame
j to K Nodes
$Y_i = \frac{1}{N}\sum_{n=1}^{N}\left|(y_{ij}(n))\right|^2$ The energy statistic at FC
$Y_i \gtrless \lambda, d_i \in (0,1)$ Decision
Training data $(Y_1,d_1)\ldots(Y_M,d_M)$
New frame Energy test $(Y_v,d_v=?)$
ML Classifier
Predict d_x associated with the new frame

Figure 1. Block diagram of machine-learning-based fusion rule spectrum sensing.

while M frames are used for training the machine-learning (ML) classifier. The received signal of i_{th} frame at the j_{th} cooperative node $y_{ij}(n)$, $1 \le n \le N$, $1 \le i \le M$, $1 \le j \le K$ is given by

$$y_{ij}(n) = \begin{cases} w_{ij}(n) & H0 \\ \sqrt{\gamma_{ij}}\ \ s_{ij}(n) + w_{ij}(n) & H1 \end{cases} \tag{1}$$

where $s_{ij}(n)$ is the PU signal which is assumed to follow Gaussian i.i.d random process (i.e., zero mean and $\sigma_s{}^2$ variance), $w_{ij}(n)$ is the noise which is also assumed to follow Gaussian i.i.d random process (zero mean and $\sigma_u{}^2$ variance) because $s_{ij}(n)$ and $w_{ij}(n)$ are independent. Due to the fact that all K nodes are sensing the same frame at a given time, the global decision about PU channel availability will be made at the fusion center only. Thus, the energy statistic for the i_{th} frame at the j_{th} cooperative node Y_{ij} can be represented by the energy test statistic of the i_{th} frame at the fusion center which is given by

$$Y_i = \frac{1}{N}\sum_{n=1}^{N}\left|\left(y_{ij}(n)\right)\right|^2, \qquad 1 \le i \le M \tag{2}$$

Y_i is a random variable that has chi-square distribution probability density function ($2N$ degrees of freedom for complex value and with N degrees of freedom for real value case). If we assume that the channel remains unchanged during the observation interval and there are enough number of samples observed $(N \ge 200)$ [8], then we can approximate Y_i using Gaussian distribution as follows:

$$Y_i = \begin{cases} \left(\sigma_{ij}{}^2, 2\sigma_{ij}{}^4/N\right) & H0 \\ \left(\sigma_{ij}{}^2\left(1+\gamma_{ij}\right), 2\sigma_{ij}{}^4\left(1+\gamma_{ij}\right)^2/N\right. & H1 \end{cases} \tag{3}$$

where $\sigma_{ij}{}^2$, is the standard deviation of noise samples $w_{ij}(n)$, and γ_{ij} is the observed signal-to-noise ratio (SNR) of the i_{th} frame sensed at the j th cooperative node. Assuming that the noise variance and the SNR at the node remain unchanged for all M frames, then $\gamma_{ij} = \gamma_j$ and $\sigma_{ij}{}^2 = \sigma_j{}^2$. For a chosen threshold λ_j for each frame in the probability of the false alarm, P_f as given in [9] can be written as

$$\begin{aligned} P_f(\lambda_j) &= \Pr(Y_i > \lambda_j | H0) \\ &= \frac{1}{\sqrt{2\pi\sigma_j}}\int_{\lambda_j}^{\infty} e^{-\left(\lambda_j - \sigma_j\right)^2/\sqrt{2}\sigma_j{}^2} \\ &= Q\left(\frac{\lambda_j}{\sigma_j{}^2} - 1\right) \end{aligned} \tag{4}$$

and the probability of detection P_d is given by

$$P_d(\lambda_j) = \Pr(Y_i > \lambda_j | H1) \\ = Q\left(\left(\frac{\lambda_j}{\sigma_j^2\left(1+\gamma_j\right)} - 1\right)\sqrt{\frac{N}{2}}\right) \tag{5}$$

where Q (.) is the complementary distribution function of Gaussian distribution with zero mean and unit variance. To obtain the optimal threshold λ for K cooperative sensing nodes, data fusion rules are used. The calculation of the thresholds for single user and other fusion rules is presented in subsections 2.1.1 and 2.1.2.

2.1.1. The detection threshold for single-user-based sensing

For single user, sensing the number of the cooperative nodes is one (i.e., K = 1, $\sigma_j^2 = \sigma_u^2$, $\gamma_j = \gamma_u$. From Eq. (4) and for a given probability of false alarm P_f, the single-user threshold can be written as

$$\lambda_{single} = \left(\sqrt{\frac{2}{N}} Q^{-1}(P_f) + 1\right)\sigma_u^2 \tag{6}$$

where $Q^{-1}(.)$ is the inverse of the Q (.) function, and the probability of the detection $P_{dsingle}$ can be written as

$$P_{dsingle} = Q\left(\left(\frac{\lambda_{single}}{\sigma_u^2(1+\gamma_u)} - 1\right)\sqrt{\frac{N}{2}}\right) \tag{7}$$

2.1.2. The detection threshold for data fusion-based sensing

In a data fusion spectrum sensing scheme, K nodes cooperate in calculating the threshold that is used to make the global sensing decision. There are many fusion rules used to calculate the global sensing decision threshold, which are divided into: hard fusion rules including AND, OR, and majority rule and soft fusion rules including maximum ratio combining (MRC), equal gain combining (EGC), and square law selection (SLS).

2.1.2.1. AND fusion rule

The AND rule decides that the signal is present if all users have detected the signal. For a system with K cooperative nodes with the same false alarm probability P_f cooperating using AND rule, the fusion center threshold can be expressed as

$$\lambda_{AND} = \left(\sqrt{\frac{2}{N}} Q^{-1}\left(P_f^{\frac{1}{K}}\right) + 1\right)\sigma_u^2 \tag{8}$$

And the detection probability P_{dAND} can be written as

$$P_{dAND} = \left(Q\left(\left(\frac{\lambda_{AND}}{\sigma_u{}^2(1+\gamma_u)} - 1 \right) \sqrt{\frac{N}{2}} \right) \right)^K \tag{9}$$

2.1.2.2. OR fusion rule

The OR rule decides that a signal is present if any of the users detect a signal. The fusion center threshold for K cooperative nodes cooperate using OR fusion rule which can be expressed as

$$\lambda_{OR} = \left(\sqrt{\frac{2}{N}} Q^{-1}\left(\left(1 - \left(1 - P_f\right)^{\frac{1}{K}}\right) + 1 \right) \sigma_u{}^2 \tag{10}$$

And the detection probability P_{dOR} is

$$P_{dOR} = \left(1 - (1 - Q\left(\left(\frac{\lambda_{OR}}{\sigma_u{}^2(1+\gamma_u)} 1 \right) \sqrt{\frac{N}{2}} \right) \right)^K \tag{11}$$

2.1.2.3. Maximum ratio combination (optimal MRC) fusion rule

In soft combination fusion K, cooperative nodes with noise variances $\left\{\sigma_{11}{}^2, \sigma_{22}{}^2, \ldots, \sigma_{MK}{}^2\right\}$ and instantaneous SNRs $\{\gamma_{11}, \gamma_{22}, \ldots, \gamma_{MK}\}$ send their i_{th} frame energy test statistics $Y_{ij} = \frac{1}{N} \sum_{n=1}^{N} \left| \left(y_{ij}(n)\right) \right|^2, 1 \le j \le K$ to the fusion center. The fusion center, weighs and adds them together after receiving these energy statistics as follows:

$$Ys_i = \sum_{j=1}^{K} w_j \, Y_{ij} \qquad , 1 \le i \le M \tag{12}$$

An assumption is made that SNRs and noise variances at the sensing node will remain unchanged for all the frames during the training process (i.e., $\gamma_{ij} = \gamma_j$, $\sigma_{ij}{}^2 = \sigma_j{}^2$). For soft optimal linear combination, we need to find the optimum weight vector w_j that maximizes the detection probability. For additive white Gaussian noise (AWGN) channels, the fusion threshold for MRC fusion rule is written as

$$\lambda_{MRC} = \left(\sum_{j=1}^{K} w_j \sigma_j{}^2 \right) Q^{-1}\left[P_f \right] + \sum_{j=1}^{K} w_j \sigma_j{}^2) \tag{13}$$

And the detection probability P_{dMRC} is given by

$$P_{dMRC} = Q\left(\left(\frac{\lambda_{MRC}}{\sum_{j=1}^{K} \left(1+\gamma_j\right) w_j \sigma_j{}^2} - 1 \right) \sqrt{\frac{N}{2}} \right) \tag{14}$$

where the weighting coefficient vector $\mathrm{w_j}\{\mathrm{w}_1, \mathrm{w}_2 \ldots \mathrm{w_K}\}$ can be obtained by:

$$\mathrm{w_j} = \mathrm{sign}\left(\mathrm{g^T w_0}\ \right)\ \mathrm{w_0}$$

where

$$w_0 = \frac{{L_{H1}}^{-1/2}\left[{L_{H1}}^{-1/T}\right]^T g}{\left\|{L_{H1}}^{-1/2}\left[{L_{H1}}^{-1/2}\right]^T g\right\|}$$

where

$$L_{H1} = 2\ \mathrm{diag}\ \left({\sigma_1}^4(1+\gamma_1)^2, \ldots..{\sigma_k}^4(1+\gamma_K)^2\right)/N$$

$$g = \left[{\sigma_1}^2\gamma_1, {\sigma_2}^2\gamma_2, {\sigma_3}^2\gamma_3, {\sigma_4}^2\gamma_4, \ldots., {\sigma_K}^2\gamma_K\right]^T$$

2.1.2.4. Equal gain combination (EGC) fusion rule

Equal weight linear combining employs straightforward averaging of the received soft decision statistics. In the equal gain combination, the received energies are equally weighted and then added together. The calculation of the threshold λ_{EGC} and the detection probability $\mathrm{P_{dEGC}}$ follow Eqs. (13) and (14), respectively; the weighting vector is $\left\{w_j = w_1, \ldots w_K\right\}$ where $w_1 = w_2 = w_3 \ldots = w_K = 1/\sqrt{K}$ [10].

2.1.2.5. Square law selection (SLS) fusion rule

Here, the fusion center selects the node with the highest SNR $\gamma_{SLS} = \mathrm{MAX}\ \left(\gamma_1, \gamma_2, ..\gamma_k\right)$ and considers the noise variance ${\sigma_{SLS}}^2$ associated with that node. Then the fusion center threshold is calculated as follows:

$$\lambda_{SLS} = \left(\sqrt{\frac{2}{N}}Q^{-1}\left(1-(1-P_f)^{\frac{1}{K}}\right)+1\right){\sigma_{SLS}}^2 \tag{15}$$

And the detection probability P_{dSLS} is

$$P_{dSLS} = 1 - \left((1 - Q\left(\left(\frac{\lambda_{SLS}}{{\sigma_{SLS}}^2(1+\gamma_{SLS})} - 1\right)\sqrt{\frac{N}{2}}\right)\right)^K \tag{16}$$

2.2. Machine-learning classification problem formulation

The i_{th} frame energy test statistic (Y_i for hard fusion or Ys_i for soft fusion rule) given in Eq. (2) or (12) is compared to the sensing threshold to calculate the decision $\mathrm{d_i}$ associated with i_{th} frame in the training data set as follows:

$$d_i = \begin{cases} 1 & Y_F \geq \lambda \\ -1 & Y_F < \lambda \end{cases} \qquad 1 \leq i \leq M \tag{17}$$

where $\lambda \in (\lambda_{single}, \lambda_{and}, \lambda_{OR}, \lambda_{MRC}, \lambda_{EGC}, \lambda_{SLS})$, $Y_F \in \{ Y_i, Ys_i\}$, M is the number of frames in the training set and "−1 " represents the absence of primary user on the channel, and "1" represents the presence of the primary user transmission on the channel. The output of Eq. (17) gives a set of pairs $(Y_i, d_i), i = 1, 2 \ldots M, d_i \in (-1, 1)$ that represent frame energy test statistics and their corresponding decisions. If we want to detect the decision (i.e., the class label) $\mathrm{d_x}$ associated with a new frame energy test statistic Y_x, we can use one of the following machine-learning classifiers to solve this classification problem.

2.2.1. K-nearest neighbors (KNN) classifier

For K-nearest neighbors classifier,K nearest points to Y_x are used to predict the class label $\mathrm{d_x}$ which corresponds to $\mathrm{Y_x}$ [11]. For $\mathrm{K} = 1$, the Euclidian distance d_{st} between Y_x and the training data points can be computed as

$$d_{st}(i) = \sqrt{(Y_x - Y_i)^2} = | Y_x - Y_i| \quad i = 1, 2 \ldots M \tag{18}$$

and, the new Y_x is classified with the label $d_x = d_{in}$, where d_{in} is the point that achieves the minimum Euclidian distance between d_{st} and Y_x.

2.2.2. Naïve Bayes classifier

Under the assumption that $d_i = -1$ and $d_i = 1$ are independent, the prior probabilities for $d_i = -1$ and $d_i = 1$ given training example $(Y_i, d_i), i = 1, 2, \ldots, M$ can be calculated, and the class-conditional densities (likelihood probabilities) can also be estimated from the set $[Y_1, Y_2, \ldots, Y_k].[\mathrm{Y_1, Y_2, \ldots, Y_k}]$ in which the new $\mathrm{Y_x}$ is expected to fall in. And, the probability that the new $\mathrm{Y_x}$ to be a member of either $d_i = -1$ or $d_i = 1$ class is calculated using Naïve Bayes assumption and Bayes rule [12] as follows:

$$class(Y_x) = \underset{d_i}{argmax} \Pr(d_i) \prod\nolimits_{j=1}^{k} \Pr(Y_j/d_i) \tag{19}$$

where the prior probabilities are given to

$$\Pr(d_i = -1) = \frac{number\ of\ Y_i\ with\ \ class\ label\ ''1''}{total\ number\ of\ class\ labels}$$

$$\Pr(d_i = 1) = \frac{number\ of\ Y_i\ with\ a\ class\ label\ ''0''}{total\ number\ of\ class\ labels}$$

Whereas the class-conditional densities "likelihood probabilities" can be estimated using Gaussian density function by:

$$\Pr\left(Y_j/d_i\right) = \frac{1}{\sigma_j\sqrt{2\pi}} e^{\frac{-\left(Y-\mu_j\right)}{2\sigma_j}}, \qquad Y_1 < Y < Y_k, \sigma_j > 0,$$

where μ_j, σ_j are mean and variance of the set $[Y_1, Y_2, \ldots, Y_k]$. Eq. (19) means that Naïve Bayes classifier will label the new Y_x with the class label d_i that achieves the highest posterior probability.

2.2.3. Support vector machine (SVM) classifier

For a given training set of pairs $(Y_i, d_i), i = 1, 2\ldots M$, where $Y_i \in R$, and $d_i \in (+1, -1)$, the minimum weight w and a constant b that maximize the margin between the positive and negative class (i.e., $w\ Y_i + b = \pm 1$) with respect to the hyper-plane equation $w\ Y_i + b = 0$ can be estimated using support vector machine classifier by performing the following optimization [13].

$$\min_{w,\, b} \left(\frac{\|w\|^2}{2} \right) \quad , \qquad where\ \|w\|^2 = w^T\ w \tag{20}$$

subject to $d_i(w\ Y_i + b) \geq 1 \qquad i = 1, 2, \ldots, M.$

The solution of this quadratic optimization problem can be expressed using Lagrangian function as

$$L(w, b, \alpha) = \frac{\|w\|^2}{2} - \sum_{i=1}^{M} \alpha_i(d_i(w\ Y_i + b) - 1), \alpha_i \geq 0 \tag{21}$$

where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\mathrm{M})$ is the Lagrangian multipliers. IF we let $L(w, b, \alpha) = 0$, we can get $w = \sum_{i=1}^{M} \alpha_i\ d_i\ Y_i$ and $\sum_{i=1}^{M} \alpha_i d_i = 0$, and by substituting them into Eq. (21), the dual optimization problem that describes the hyper-plane can be written as

$$\min_{\alpha} \left(\frac{1}{2} \sum_{i=1}^{M} \sum_{j=1}^{M} d_i d_j \left(Y_i\ Y_j\right) \alpha_i \alpha_j - \sum_{i=1}^{M} \alpha_j \right), \alpha_j \geq 0 \tag{22}$$

From expression (22), we can assess α and compute w using $w = \sum_{i=1}^{M} \alpha_i\ d_i\ Y_i$. Then by choosing $\alpha_i > 0,$ from the vector of $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_\mathrm{M})$ and calculating b from b $= d_j - \sum_{i=1}^{M} \alpha_i d_i \left(Y_i\ Y_j\right)$, we classify the new instance Y_x using the following classification function

$$class(Y_x) = \mathrm{sign}\left(\sum_{i=1}^{M} \alpha_j d_j (Y_i\ Y_x) + b \right) \tag{23}$$

which means that the classification of new Y_x can be expressed as dot product of Y_x and the support vectors.

2.2.4. Decision tree (DT) classifier

For the training of a set of pairs of sensing decision $(Y_i, d_i), i = 1, 2, \ldots, M$, $d_i \in (-1, 1)$, the decision tree classifier creates a binary tree based on either impurity or node error splitting rule in order to split the training set into separate subset. Then, it repeats the splitting rule recursively for each subset until the leaf of the subset becomes pure. After that, it minimizes the error in each leaf by taking the majority vote of the training set in that leaf [14]. For classifying a new example Y_x, DT classifier selects the leaf where the new Y_x falls in and classifies the new Y_x with the class label that occurs most frequently among that leaf.

2.3. Performance discussion

Figure 2 shows the receiver operating characteristic (ROC) curves for single-user soft and hard fusion rules under Additive White Gaussian Noise (AWGN) channel. In order to generate this figure, we assume a cognitive radio system with 7 cooperative nodes (i.e., K = 7) operate at SNR γ_u = −22 dB. The local node decisions are made after observing1000 samples (i.e., energy

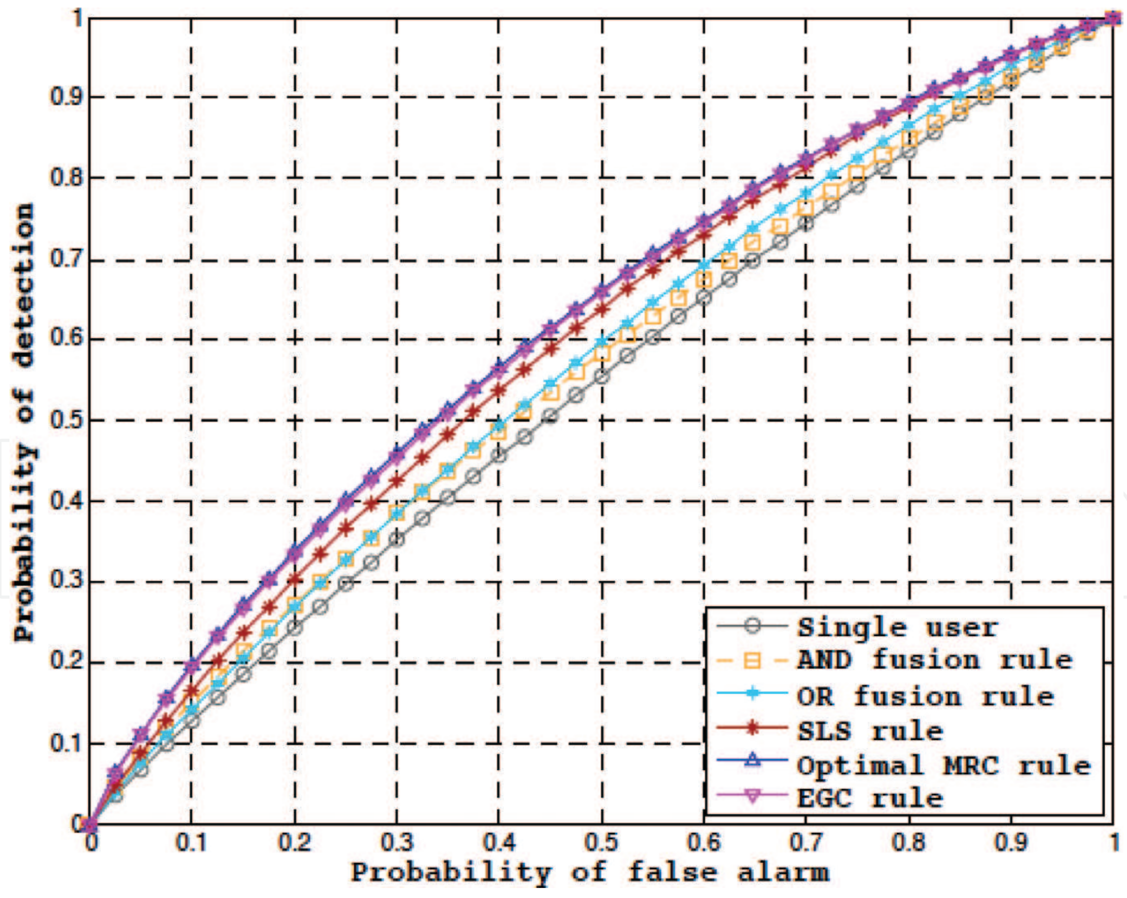


Figure 2. ROC curves for the soft and hard fusion rules under the case of AWGN receiver noise, $\sigma_u{}^2 = 1, \gamma_u = -22$ dB, $K = 7$ users and energy detection over N=1000 samples.

detection samples $N = 100$). For soft fusion rules, the SNRs γ_j for the nodes are equal to $\{-24.3, -21.8, -20.6, -21.6, -20.4, -22.2, -21.3\}$ and the noise variances σ_j^2 are $\{1, 1, 1, 1, 1, 1, 1\}$. We use a false alarm probability P_f varied from 0 to 1 increasing by 0.025. The simulation results show that soft EGC and optimal MRC fusion rules perform better than other soft and hard fusion rules even though that soft EGC fusion rule does not need any channel state information from the nodes.

Figure 3 shows the ROC curve depicting the performance of SVM classifier in classifying 1000 new frames after training it over a set containing $M = 1000$ frames. The thresholds used for training SVM classifier (i.e., single-user threshold, AND, OR, MRC, SLS, and EGC fusion rule threshold) are obtained numerically by considering the cognitive system used to generate **Figure 2**; however, here, we set the false alarm probability to $P_f = 0.1$.

From **Figure 3** and **Table 1**, we can notice that when training SVM classifier with anyone of the following thresholds: single user, OR, MRC, SLS, or EGC, it can detect 100% positive classes. We can also notice that training with EGC threshold can provide 90% precession in classifying the positive classes with 10% harmful interference, whereas training SVM with AND threshold can precisely classify the positive classes by 97.8%. **Table 1** shows the classification accuracy of

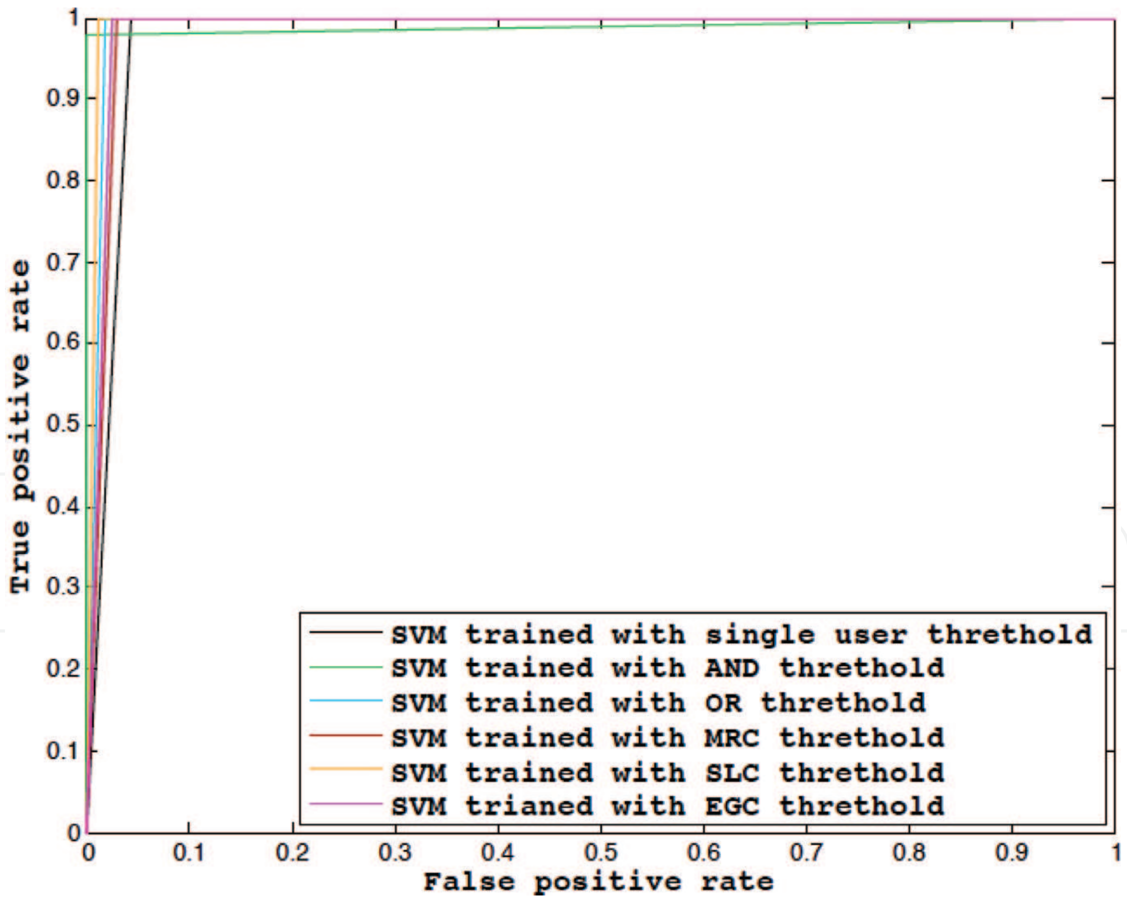


Figure 3. ROC curves shows the performance of SVM classifier in predicting the decisions for 1000 new frames after training it over a set containing1000 frames when single user, AND, OR, MRC, SLS, and EGC thresholds are used for training process.

Threshold	Single user (%)	AND rule (%)	OR rule (%)	MRC rule (%)	SLS rule (%)	EGC rule (%)
SVM						
Accuracy	96.1	98.3	98.1	97.6	98.9	98.0
Precession	77.7	100	53.7	89.4	74.4	90.1
Recall	100	97.8	100	100	100	100

Table 1. The accuracy, precession and the recall of SVM classifier.

SVM classifier (i.e., the proportion of all true classifications over all testing examples) and the precession of classification (i.e., proportion of true positive classes over all positive classes) as well as the recall of classification (i.e., the effectiveness of the classifier in identifying positive classes).

Figure 4 shows ROC curves showing the comparison of four machine-learning classifiers: K-nearest neighbor (KNN), support vector machine (SVM), Naive Bayes and Decision tree when used to classify 1000 frames after training them over a set containing 1000 frames with single-user threshold (Note: the same system used to generate the simulation of **Figure 3**. is considered

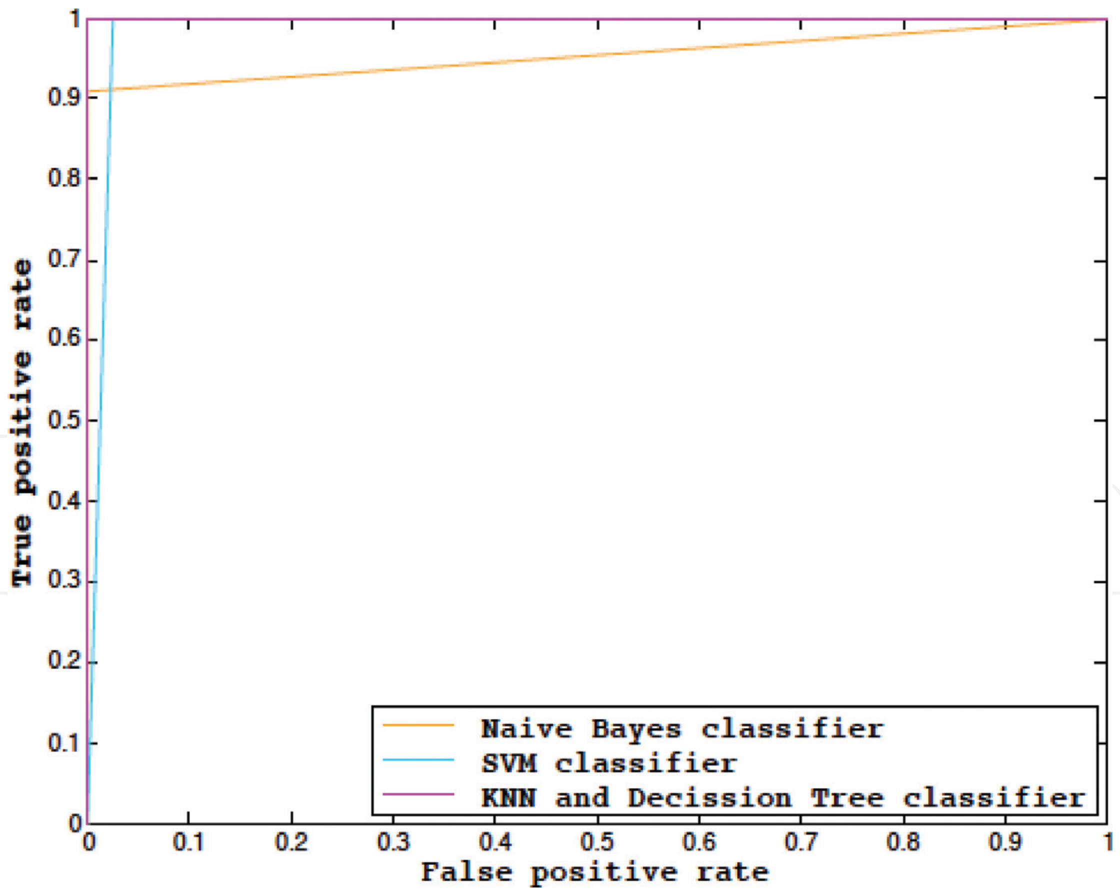


Figure 4. ROC curves shows a comparison of four machine learning classifiers: KNN, SVM, naive Bayes, and decision tree in classifying 1000 frames after training them over a set with 1000 frames using single user scheme threshold.

for computing the single-user threshold). We can notice from both **Figure 4**. and **Table 2** that KNN and decision tree classifier perform better than Naïve Bayes and SVM classifier in terms of the accuracy of classifying the new frames.

Table 3 shows the accuracy, precession, and the recall for decision tree classifier when used to classify 3000 frames after training it over a set containing 1000 frames for the same cognitive system used to generate **Figure 3**. The single-user threshold is used for training the classifier. The simulation was run with different number of samples for energy detection process. It is clear from the table that decision tree can classify all of the 3000 frames correctly or achieve 100% detection rate using only 200 samples for the energy detection process. And, due to the fact that the sensing time is proportional to the number of samples taken by energy detector, a less number of samples used for energy detection leads to less sensing time. Thus, when we use machine-learning-based fusion, such as decision tree or KNN, we can reduce the sensing time from 200 to 40 μs for 5 MHz bandwidth channel as an example, while we still achieve 100% detection rate of the spectrum hole.

Classifier	**Accuracy (%)**	**Precession (%)**	**Recall (%)**
KNN	100	100	100
Decision Tree	100	100	100
Naïve Bayes	98.9	100	91.2
SVM	97.6	83.9	100

Table 2. The accuracy, precession and recall of KNN, SVM, NB, and DT classifiers used in classifying 1000 new frames after being trained with 1000 frames.

Number of samples	**Accuracy (%)**	**Precession (%)**	**Recall (%)**
200	100	100	100
400	100	100	100
600	100	100	100
800	100	100	100
1000	100	100	100

Table 3. The accuracy, precession, and recall for decision tree classifier used in classifying 3000 frames for different number of samples.

3. Prediction of PU channel state based on hidden Markov and Markov switching model

In this part, the system model for forecasting the near future of PU channel state is divided into three models: (1) the model detecting the PU channel state (i.e., PU signal present or PU signal) which follows the conventional single-user energy detection (i.e., fusion techniques mentioned in Section 2.1 can also be considered here); (2) the model that generates a time series to capture PU channel state based on the detection sequence; and (3) the model for predicting the generated time series used to capture PU channel state based on hidden Markov model (HMM) and Markov switching model (MSM). The block diagram in **Figure 5**. illustrates these three models.

The PU channel state detection model can be written using Eq. (4); by giving probability of false alarm P_f, the detection threshold for single-user energy detector can be written as:

$$\lambda = \left(\sqrt{\frac{2}{N}}Q^{-1}(P_f) + 1\right){\sigma_u}^2 \tag{24}$$

where $Q^{-1}(.)$ is the inverse of the Q (.) function.

And the decision of the sensing (i.e., PU detection sequence) over the time can be written as follows:

$$D_t = \begin{cases} {''0''} & \text{PU absent} \quad Y_t < \lambda \\ {''1''} & \text{PU present} \quad Y_t \geq \lambda \end{cases} \quad 1 \leq t \leq T \tag{25}$$

3.1. Time series generation model

Given PU channel state detection sequence over the time (i.e., PU absent, PU present), if we denote the period that the PU is inactive as "idle state," and the period that PU is active as "occupied state," our goal now is to predict when the detection sequence D_t will change from one state to another (i.e., "idle" to "occupied "or vice versa) before that happens so that the secondary user can avoid interfering with primary user transmission. For this reason, we generate a time series z_t to map each state of the detection sequence D_t (i.e., "PU present" and "PU absent") into another observation space using two different random variable distributions for each state (i.e., $z_t \in \{v_1, v_2 \dots v_L\}$ represents PU absent or idle state and $z_t \in \{v_{L+1} \dots . v_M\}$ represents PU occupied or present), the time series z_t can be written as

$$z_t \in \begin{cases} \{v_1, v_2, \dots, v_L\} & Y_t < \lambda \\ \{v_{L+1}, ..v_M\} & Y_t \geq \lambda \end{cases} \quad 1 \leq t \leq T \tag{26}$$

Now, supposing that we have given observations value $O = \{O_1, O_2, O_t, \dots O_T\}$, $O_t \in \{v_1, v_2 \dots v_M\}$ and a PU channel state at time step t, $X_t \in s_i$, $i = 1, 2 \dots . K$, $s_i \in \{0, 1\}$ (i.e., 0 for

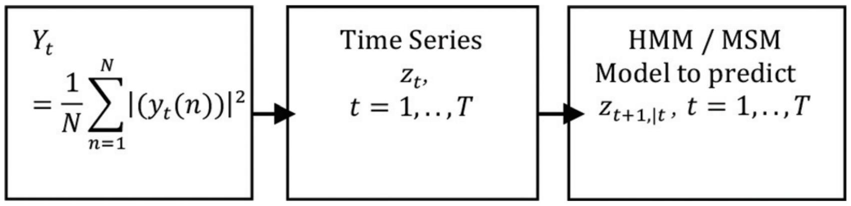

$Y_t = \frac{1}{N}\sum_{n=1}^{N}|(y_t(n))|^2$
Time Series
z_t, $t = 1,..,T$
HMM / MSM
Model to predict
$z_{t+1,|t}$, $t = 1,..,T$

Figure 5. Block diagram of PU channel state prediction model.

PU idle and 1 PU occupied state), and we want to estimate the channel state at one time step ahead of the current state X_{t+1}. We can solve this problem using hidden Markov model Viterbi algorithm [15].

3.2. Primary users channel state estimation based on hidden Markov model

The generic HMM model can be illustrated by **Figure 6.—in this figure,** $X = \{ X_1, X_t, \dots X_T\}$ represents the hidden state sequence, where $X_t \in \{s_1, s_2, \dots, s_K\}$, K represents the number of hidden states or Markov chain and $O = \{ O_1, O_t, \dots, O_T\}$ represents the observation sequence where $O_t \in \{ v_1, v_2, \dots, v_M\}$ and M is the number of the observations in the observation space. A and B represent the transition probabilities matrix and the emission probabilities matrix, respectively, while π denotes the initial state probability vector. HMM can be defined by $\theta = (\pi, A, B)$ (i.e., the initial state probabilities, the transition probabilities, and emission probabilities) [15].

Initial state probabilities for HMM can be written as

$$\pi = (\pi_1, \pi_2, \pi_i \dots \quad \pi_K)$$
$$\pi_i = \ P(X_1 = s_i), \quad i = 1, 2, \dots, K \tag{27}$$

For a HMM model with two hidden states $i = 2$,

$$\pi = (\pi_1 \pi_2)$$

And the transition probabilities can be written as,

$$A = (a_{ij})_{K\times K}$$
$$a_{ij} = P\left(X_{t+1} = s_j | X_t = s_i\right) \quad , i, j = 1, \dots.., K \tag{28}$$

where a_{ij} is the probability that next state equal s_j when current state is equal to s_i. For HMM model with two states, the matrix A can be written as

$$A = \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix}$$

The emission probabilities matrix for HMM model is written as

$$B = (b_{jm})_{K\times M}$$
$$b_j(m) = P(O_t = v_m | X_t = s_j) \triangleq b_j(O_t), \quad j = 1 \dots K \ , m = 1, \dots..M \tag{29}$$

B and b_j represent the probability that current observation is v_m when current state is s_j. For example, in an HMM model with $M = 6$ and $K = 2$, B is written as

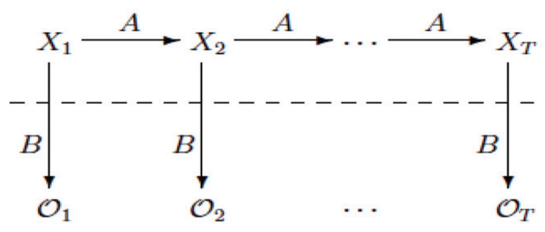


Figure 6. Hidden Markov model.

$$B = \begin{pmatrix} b_{11} & b_{12}\, b_{13} & b_{14} & b_{15}\, b_{16} \\ b_{21} & b_{22}\, b_{23} & b_{24} & b_{25}\, b_{26} \end{pmatrix}$$

Now, for the problem we describe in subsection (3.1), if we assume that HMM parameters $\theta = (\pi, A, B)$ and the observations value $O = \{ O_1, O_2\, O_t, \dots O_T\}$ are given. If we assume that the maximum probability of state sequence t that end at state i to be equal to $\delta_t(i)$ where

$$\delta_t(i) = \max_{X_1, \dots, X_{t-1}} \{P(X_1, \dots, X_t = s_i; O_1, \dots, O_t|\theta\ )\} \tag{30}$$

And we let $\psi_t(i)$ to be a vector that stores the arguments that maximize Eq. (30), we can write Viterbi algorithm to solve the problem mentioned in subsection (3.1) as follows:

1) step 1 initializes $\delta_t(i)$ and $\psi_t(i)$.

$$\delta_t(i) = \pi_i b_i(O_1)$$
$$\psi_t(i) = 0, \quad i = 1, \dots, K \tag{31}$$

2) step 2 iterates to update $\delta_t(i)$ and $\psi_t(i)$.

$$\delta_t(j) = \max_{1 \le i \le K} [\delta_{t-1}(i) a_{ij}] b_j(O_t) \quad , t = 2, \dots, T, \quad j = 1, \dots, K \tag{32}$$

$$\psi_t(j) = \underset{1 \le i \le K}{\operatorname{argmax}} [\delta_{t-1}(i) a_{ij}] \quad , t = 2, \dots, T, j = 1, \dots, K \tag{33}$$

3) step 3 terminates the update and calculates the likelihood probability P^* and the estimated state $q_T{}^*$ at time T as

$$P^* = \max_{1 \le i \le K} [\delta_T(i)] \tag{34}$$

$$q_T{}^* = \underset{1 \le i \le K}{\operatorname{argmax}} [\delta_T(i)] \tag{35}$$

In the above case, HMM parameters $\theta = (\pi, A, B)$ are unknown and need to be estimated. We estimate these parameters statistically using Baum-Welch algorithm [16].

3.2.1. Hidden Markov model parameters estimation using Baum-Welch algorithm

If we assume that we have given some training observations with length L $\{O_1, O_2\, O_t, \ldots O_L\}$ and want to approximate HMM parameters $\theta = (\pi, A, B)$ from them, we can use maximum likelihood estimation. In order to do that, we define $\gamma_t(i)$ to be the probability of being in state s_i at time t, given t O_t ,$t = 1, 2 \ldots L$. $\gamma_t(i)$ is written as

$$\gamma_t(i) = P(X_t = s_i \mid O_1, \ldots, O_L, \theta) \tag{36}$$

We also define $\zeta_t(i, j)$ to be the probability of being in state s_i at time t and transiting to state s_j at time $t + 1$, given O_t ,$t = 1, 2 \ldots L$. $\zeta_t(i, j)$ is written as

$$\zeta_t(i, j) = P\big((X_t = s_i; X_{t+1} = s_j \big| O\, O_1, \ldots, O_L, \theta\ \big) \tag{37}$$

Given $\gamma_t(i)$ and $\zeta_t(i, j)$, the anticipated number of transitions from state s_i during the path is written as

$$E\big(\gamma_t(i)\big) = \sum_{t=1}^{L-1} \gamma_t(i) \tag{38}$$

and the anticipated number of transitions from state s_i to state s_j during the path is written as

$$E(\zeta_t(i, j)\ ) = \sum_{j=1}^{L-1} \zeta_t(i, j) \tag{39}$$

Given E $(\zeta_t(i, j))$ and E $\big(\gamma_t(i)\big)$, we can extract the model parameters $\theta = (\pi, A, B)$ from the training sequence as given in [16] using the step listed below

1- for $i = 1, 2, 3 \ldots K$, let $\widehat{\pi}_i =$ expected frequency in state s_i at time $(t = 1)$

$$\pi_i = \gamma_1(i) \tag{40}$$

2- for $i = 1, 2, 3 \ldots K$ and $j = 1, 2, 3 \ldots K$, compute

$$\widehat{a}_{ij} = \frac{\text{Expected number of transitions fromstate } s_i \text{ to state } s_j}{\text{Expected number of transitions from state } s_i}$$

$$= \frac{E(\zeta_t(i, j))}{E\big(\gamma_t(i)\big)} = \frac{\sum_{j=1}^{L-1} \zeta_t(i, j)}{\sum_{t=1}^{L-1} \gamma_t(i)} \tag{41}$$

3- for $i = 1, 2, 3 \ldots K$ and $j = 1, 2, 3 \ldots K$, compute

$$\widehat{b}_i(m) = \frac{\text{Expected number of times in state } s_j \text{ and observing } v_m}{\text{Expected number of times in state } s_j}$$
$$= \frac{\sum_{\substack{t=1 \\ O_t = v_m}}^{L} \gamma_t(i)}{\sum_{t=1}^{L} \gamma_t(i)} \tag{42}$$

The estimation algorithm can be summarized in the following steps:

1. Get your observations $O_1\, O_2, \ldots O_L$,
2. Set a guess of your first θ estimate θ (1), k = 1

$$\text{Update } k = k + 1$$

3. Compute θ (k) based on $O_1\, O_2, \ldots O_L$ and

$$\gamma_t(i), \zeta_t(i,j) \quad \forall 1 \le t \le L, \quad \forall 1 \le i \le K, \forall 1 \le j \le K$$

4. Compute $E(\gamma_t(i))$ and $E(\zeta_t(i,j))$ from Eqs. (38) and (39)
5. Compute according to 5 the new estimate of a_{ij}, $b_i(k)$, π_i, and call them θ (k + 1)
6. Go to 3 if not converged.

The prediction for a one-step ahead PU channel state can be done based on the trained parameters {π, A, B } with the help of Eqs. (31), (34), and (35) by setting T = 1.

3.3. Primary users channel state estimation based on Markov switching model

An alternative way to estimate PU channel state is to use Markov switching model (MSM). For the time series in Eq. (26), we assume that z_t obeys two different Gaussian distributions $N \sim (\mu_{z0}, \sigma_{z0}{}^2)$ or $N \sim (\mu_{z1}, \sigma_{z1}{}^2)$ based on the sensed PU channel state "PU channel idle" or "PU occupied." We can rewrite Eq. (26) as follows:

$$z_t \sim \begin{cases} (\mu_{z0}, \sigma_{z0}{}^2) & Y_t < \lambda \\ (\mu_{z1}, \sigma_{z1}{}^2) & Y_t \ge \lambda \end{cases} \quad 1 \le t \le T \tag{43}$$

It is obvious that Eq. (43) represents a two-state Gaussian regime switching time series which can be modeled using MSM [17]. In order to estimate the switching time of one state ahead of the current state for this time series, we need to derive MSM regression model for the time series and estimate its parameters.

3.3.1. Derivation of Markov switching model for Gaussian regime switching time series

A simple Markov switching regression model to describe the two-state Gaussian regime switching time series is given in Eq. (43). This model can be written by following Ref [17] as

$$z_t = \mu_{s_t} + \epsilon_t \qquad \epsilon_t \sim \left(0, {\sigma_{s_t}}^2\right) \tag{44}$$

where μ_{s_t} is an array of predetermined variables measured at time t, which may include the lagged values of z_t, ϵ_t is the white noise process, $s_t = \{0, 1\}$ is a hidden Markov chain which has a mean and standard deviation over the time equal to $\mu_{st} = \mu_0(1 - s_t) + \mu_1\, s_t$ and $\sigma_{st} = \sigma_0(1 - s_t) + \sigma_1\, s_t$, respectively (the state variable s_t follows first order Markov chain (i.e., two-state Markov chain as in [18])). Given the past history of s_t, the probability of s_t taking a certain value depends only on s_{t-1}, which takes the following Markov property:

$$P(\, s_t = j|\, s_{t-1} = i) = P_{ij} \tag{45}$$

where $P_{ij}(i; j = 0; 1)$ denotes the transition probabilities of $s_t = j$, given that $s_{t-1} = i$. Clearly, the transition probabilities satisfy $P_{i0} + P_{i1} = 1$. We can gather the transition probabilities P_{ij} into a transition matrix as follows:

$$\begin{aligned} P &= \begin{pmatrix} P(s_t = 0|\, s_{t-1} = 0) & P(\, s_t = 0|\, s_{t-1} = 1) \\ P(\, s_t = 1|\, s_{t-1} = 0) & P(\, s_t = 1|\, s_{t-1} = 1) \end{pmatrix} \\ &= \begin{pmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{pmatrix} \end{aligned} \tag{46}$$

The transition matrix P is used to govern the behavior of the state variable s_t , and it holds only two parameters (P_{00} and P_{11}). Assuming that we do not observe s_t directly, we only deduce its operation from the observed behavior of z_t. The parameters that need to be estimated to fully describe the probability law governing z_t are the variance of the Gaussian innovation σ_0, σ_1 , the expectation of the dependent variable μ_0, μ_1, and the two-state transition probabilities P_{00} and P_{11}.

3.3.2. Markov switching model parameters estimation via maximum likelihood estimation

There are many ways to estimate the parameters for the Markov switching model. Among these ways are Quasi-maximum likelihood estimation (QMLE) and Gibbs sampling. In this section, we focus on maximum likelihood estimation (MLE).

If we denote ψ_{t-1} = $\{z_{t-1}, z_t, z_{t+1} \ldots z_1\}$ to be a vector of the training data until time $t - 1$ and denote θ =$\{\sigma_0, \sigma_1, \mu_0, \mu_1, P_{00}, P_{11}\}$ to be the vector of MSM parameters, then ψ_L = $\{z_{t-1}, z_t, \ldots, z_L\}$ to a vector of the available information with the length L sample (see **Figure 7**.). In order to evaluate the likelihood of the state variable s_t based on the current trend of z_t, we need to assess its conditional expectations $s_t = i$, $(i = 0, 1)$ based on ψ and θ . These conditional expectations include prediction probabilities $P\,(s_t = i|\psi_{t-1}; \theta)$, which are based on the information prior to time t, the filtering probabilities $P\,(s_t = i|\psi_t; \theta)$ which are based on the past and current information, and finally the smoothing probabilities $P\,(s_t = i|\psi_L; \theta)$ which are based on the full-sample information L. After getting these probabilities, we can obtain the log-likelihood function as a byproduct, and then we can compute the maximum likelihood estimates.

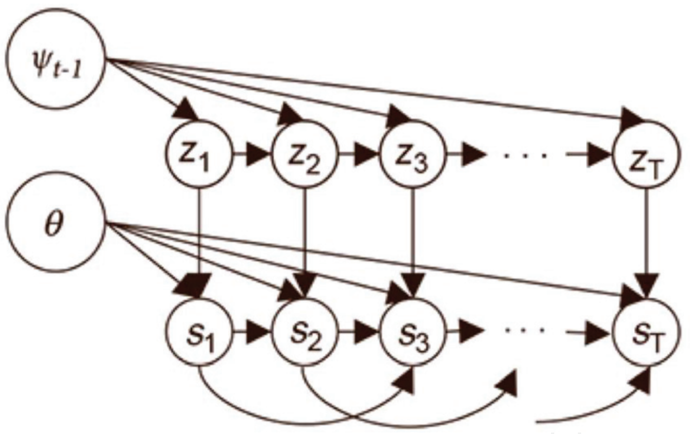


Figure 7. Markov switching model.

Normally, the density of z_t conditional on ψ_{t-1} and $s_t = i$, $(i = 0, 1)$ is written as

$$\mathcal{F}(z_t |\, s_t = i, \psi_{t-1}; \theta) = \frac{1}{\sqrt{2\pi\sigma_{s_t}}} e^{-\frac{\left(z_t - \mu_{s_t} \right)^2}{2\sigma_{st}{}^2}} \tag{47}$$

where $\mathcal{F}$ represent the probability density function. Given the prediction probabilities P $(s_t = i \,|\psi_{t-1}; \theta)$, the density of z_t conditional on ψ_{t-1} can be obtained from

$$\begin{aligned} &\mathcal{F}(z_t |\psi_{t-1}; \theta) = \\ &= \ \mathrm{P}\,(s_t = 0|\psi_{t-1}; \theta)\, \mathcal{F}(z_t |\, s_t = 0\ \psi_{t-1}; \theta) \\ &\quad + \mathrm{P}\,(s_t = 1|\psi_{t-1}; \theta)\, \mathcal{F}(z_t |\, s_t = 1\ \psi_{t-1}; \theta) \end{aligned} \tag{48}$$

For $i = 0; 1$, the filtering probabilities of s_t are given by:

$$\begin{aligned} &\mathrm{P}\,(s_t = i|\psi_t; \theta) \\ &= \frac{\mathrm{P}\,(s_t = i|\psi_{t-1}; \theta)\, \mathcal{F}(z_t |\, s_t = i\ \psi_{t-1}; \theta)}{\mathcal{F}(z_t |\psi_{t-1}; \theta)} \end{aligned} \tag{49}$$

The prediction probabilities are:

$$\begin{aligned} &\mathrm{P}\,(s_{t+1} = i|\psi_t; \theta) \\ &= \ \mathrm{P}\,(s_t = 0, s_{t+1} = i|\psi_t; \theta) + \mathrm{P}\,(s_t = 1, s_{t+1} = i|\psi_t; \theta) \\ &= \mathrm{P}_{0i}\, \mathrm{P}\,(s_t = 0|\psi_t; \theta) + \mathrm{P}_{1i}\, \mathrm{P}\,(s_t = 1|\psi_t; \theta) \end{aligned} \tag{50}$$

where $\mathrm{P}_{0i} = \mathrm{P}\,(s_{t+1} = i|s_t = 0)$ and $\mathrm{P}_{1i} = \mathrm{P}\,(s_{t+1} = i|s_t = 1)$ are the transition probabilities. By setting the initial values as given in [19] assuming the Markov chain is presumed to be ergodic:

$$P\left(s_0 = i|\psi_0\right) = \frac{1 - P_{jj}}{2 - P_{ii} - P_{jj}}$$

we can iterate the Eqs. (49) and (50) to obtain the filtering probabilities $P\left(s_t = i|\psi_t; \theta\right)$ and the conditional densities $\mathcal{F}\left(\left. z_t\right| s_t = 0\ \psi_{t-1}; \theta\right)$ for $t = 1, 2, \ldots..T$. Then we can compute the logarithmic likelihood function using

$$\log\left(L\left(\widehat{\theta}\right)\right) = \sum_{t=1}^{T}\sum_{i=1}^{2}\log(\mathcal{F}\left(\left. Z_t\right| S_t = i, \psi_{t-1}; \theta\right) \times P\left(S_t = i|\psi_t; \theta\right)) \tag{51}$$

where $L\left(\widehat{\theta}\right)$ is the maximized value of the likelihood function. The model estimation can finally be obtained by finding the set of parameters $\widehat{\theta}$ that maximize the Eq. (51) using numerical-search algorithm. The estimated filtering and prediction probabilities can then be easily calculated by plugging $\widehat{\theta}$ into the equation formulae of these probabilities. We adopt the approximation in Ref [20] for computing the smoothing probabilities $P\left(s_t = i|\psi_L; \theta\right)$

$$\begin{aligned} P\left(s_t = i\right| s_{t+1} = j, \psi_L; \theta\right) &\approx P\left(s_t = i\right| s_{t+1}, \psi_t; \theta\right) \\ &= \frac{P\left(s_t = i, s_{t+1}|\psi_{t-1}; \theta\right)}{P\left(s_{t+1} = j|\psi_t; \theta\right)} \\ &= \frac{P_{0i}\, P\left(s_t = i|\psi_t; \theta\right)}{P\left(s_{t+1} = j|\psi_t; \theta\right)} \end{aligned}$$

And, for $i; j = 0; 1,$ smoothing probabilities is expressed as:

$$\begin{aligned} &P\left(s_t = i|\psi_L; \theta\right) \\ &= P\left(s_{t+1} = 0|\psi_L; \theta\right) P\left(s_t = 1\right| s_{t+1} = 0, \psi_L; \theta\right) \\ &+ P\left(s_{t+1} = 1|\psi_L; \theta\right) P\left(s_t = i\right| s_{t+1} = 1, \psi_L; \theta\right) \\ &= P\left(s_t = i|\psi_t; \theta\right) \times \left(\frac{P_{i0}\, P\left(s_{t+1} = 0|\psi_L; \theta\right)}{P\left(s_{t+1} = 0|\psi_t; \theta\right)} + \frac{P_{i1}\, P\left(s_{t+1} = 1|\psi_L; \theta\right)}{P\left(s_{t+1} = 1|\psi_t; \theta\right)}\right) \end{aligned} \tag{52}$$

Using $P\left(S_L = i|\psi_L; \theta\right)$ as the initial value, we can iterate the equations regressively for filtering and prediction probabilities along with the equation above to get the smoothing probabilities for $t = L - 1, \cdot\cdot, k + 1$.

3.4. Results and discussions

Figure 8 shows the training detection sequence which we generate as a training observation using randomly distributed PU channel state "idle and occupied" over T = 250 ms simulation

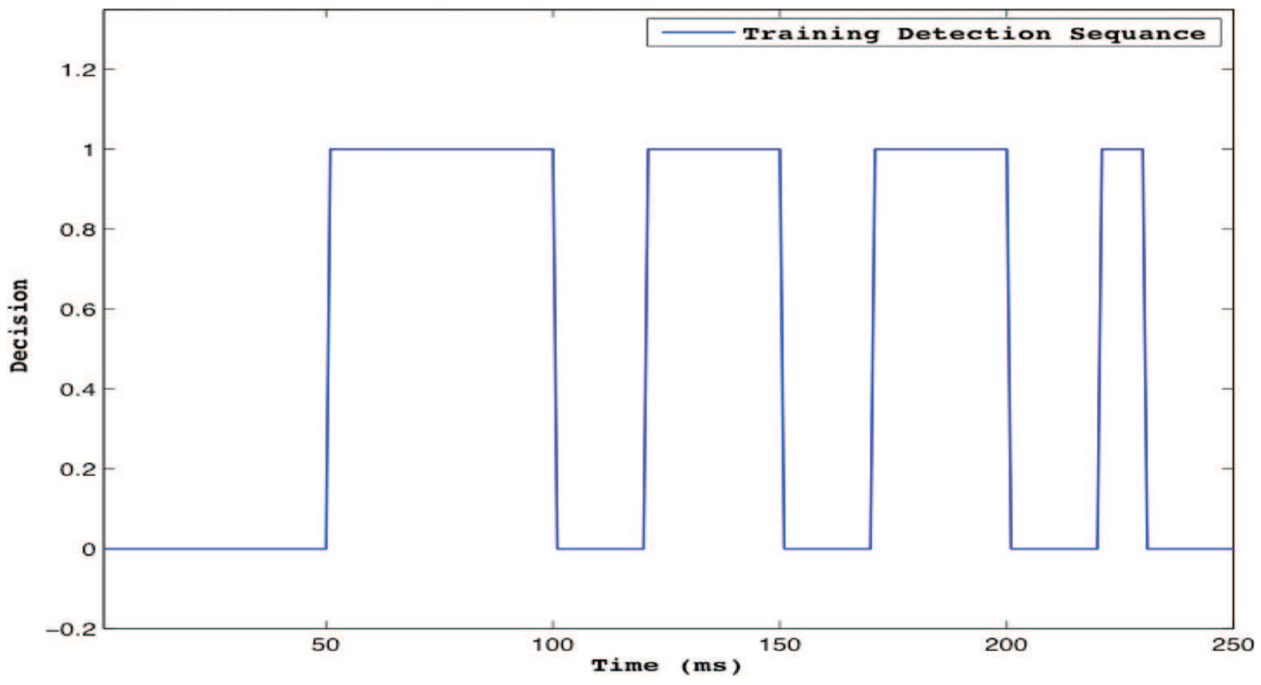


Figure 8. The training detection sequence for HMM and MSM.

time. We use this training observations to train *Baum-Welch algorithm* in order to estimate HMM model parameters $\theta = (\pi, A, B)$, assuming that the first estimate of θ (1) is:

$$\pi_1 = (1 \quad 0)$$

$$A_1 = \begin{pmatrix} 0.85 & 0.15 \\ 0.10 & 0.90 \end{pmatrix}$$

$$B_1 = \begin{pmatrix} 0.17 & 0.16 & 0.17 & 0.16 & 0.17 & 0.17 & 0.17 \\ 0.60 & 0.08 & 0.08 & 0.08 & 0.08 & 0.08 & 0.08 \end{pmatrix}$$

Figure 9a shows the performance of HMM algorithm in estimating the PU channel states (i.e., PU idle or PU occupied) of the time series that capture the detection sequence for a single-user cognitive radio network. **Figure 9a** contains three plots; the top plot shows the randomly distributed PU channel states over time $T = 500$ ms . The middle plot shows the generated time series following the distribution $z_t \in \{1, 2, 3\}$ for idle states and $z_t \in \{4, 5, 6\}$ for occupied states (note: we can construct the observation space from these two distributions as $O_t \in \{1, 2, 3, 4, 5, 6\}$, $t = 1, 2 \ldots 500$ ms). The bottom plot shows performance of HMM algorithm in forecasting the time series generated to capture PU detection sequence.

Figure 9b shows the performance of MSM algorithm in predicting the switching process between the two PU channel states for the same PU detection sequence given in **Figure 9a** $T = 500$ ms . The top graph in **Figure 9b** shows the generated time series with the following distribution $z_t \sim (0.1, 0.5)$ for idle states and $z_t \sim (0.01, 0.2)$ for occupied states and the bottom graph shows the prediction performance using MSM. As it is clear from the figure, the prediction performance is smoother than HMM approach.

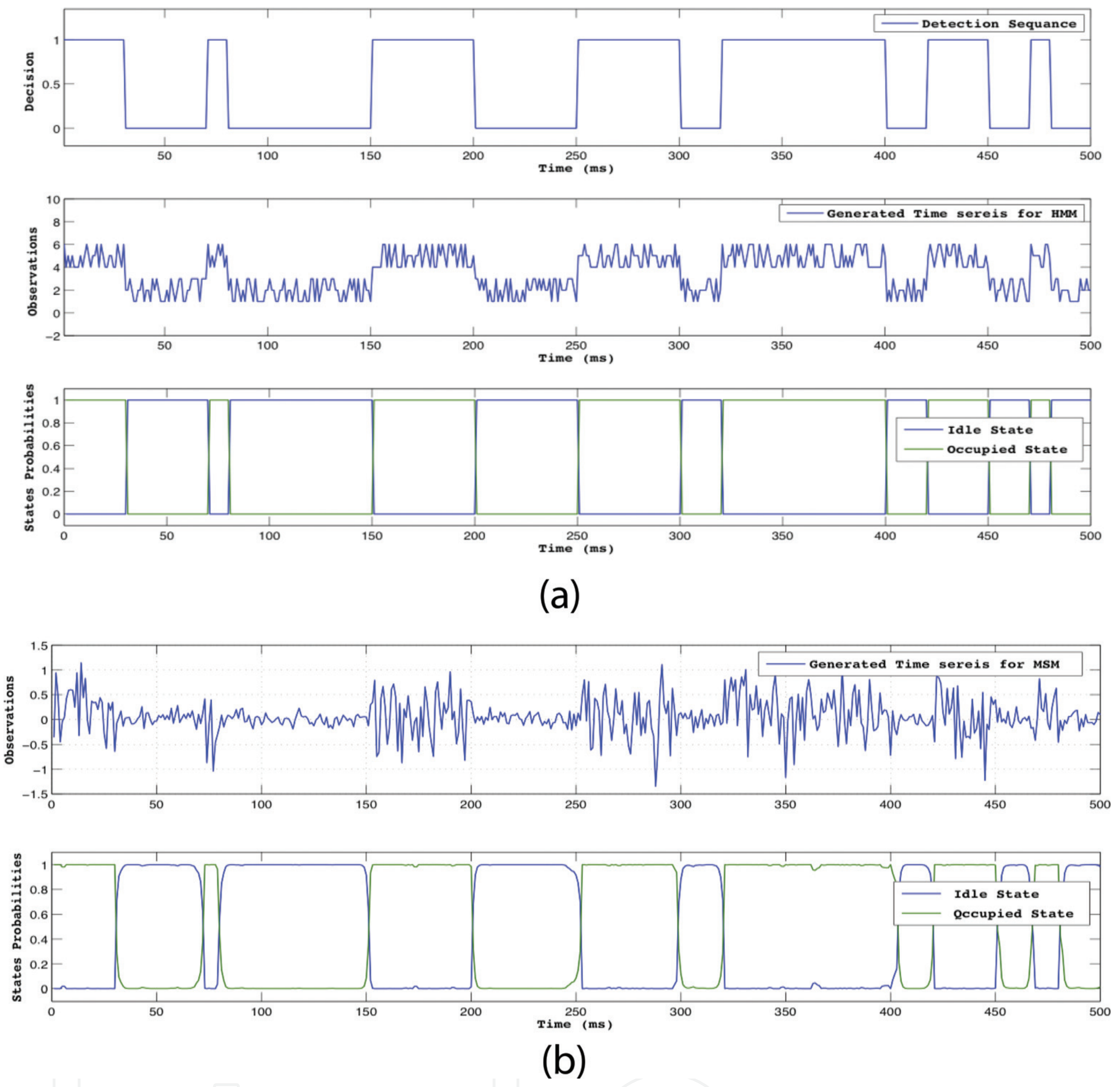


Figure 9. (a) Shows the performance of HMM algorithm in predicting the generated time series to capture PU channel state detection sequence. (b) Shows the performance of MSM algorithm in predicting the generated time series to capture the same PU detection sequence in **Figure 8**.

4. Conclusions

In this chapter, we have presented a per-frame decision-based cooperative spectrum sensing based on machine-learning classifier-based fusion approach. The simulation and numerical results have shown that the machine-learning classifier-based fusion algorithm performs same as conventional fusion rules in terms of sensing accuracy with less sensing time, overheads, and extra operations that limit achievable cooperative gain among cognitive radio users. In addition, we have also studied the problem of primary user channel state prediction in cognitive radio network and introduced Markov model and Markov Switching Model to solve this problem. We finally showed by the means of simulation that both hidden Markov model and Markov switching model perform very well in predicting the time series that capture the actual primary user channel state.

Acknowledgements

I gratefully acknowledge the funding received from Shanghai Jiao Tong University to undertake my PhD. I also thank Prof. Bin Guo for his encouragement and help on the topic.

Author details

Ahmed Mohammed Mikaeil

Address all correspondence to: ahmed_mikaeil@yahoo.co.uk

Department of Electronic Engineering, School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China

References

[1] Teguig D, Scheers B, Le Nir V. Data fusion schemes for cooperative spectrum sensing in cognitive radio networks. Communications and Information Systems Conference (MCC), 2012 Military. IEEE; 2012

[2] Zhai X, Jianguo P. Energy-detection based spectrum sensing for cognitive radio. 2007:944-947

[3] Mikaeil AM, Guo B, Bai X, Wang Z. Machine learning to data fusion approach for cooperative spectrum sensing. 2014 International Conference on Cyber Enabled Distributed Computing and Knowledge Discovery(CyberC), Shanghai,13–15 October 2014, 429-434

[4] Zhe C, Qiu RC. Prediction of channel state for cognitive radio using higher-order hidden Markov model. IEEE SoutheastCon 2010 (SoutheastCon), Proceedings of the IEEE. 2010

[5] Zhe C et al. Channel state prediction in cognitive radio, part II: Single-user prediction. Proceedings of IEEE Southeast Con. 2011

[6] Chang-Hyun P et al. HMM based channel status predictor for cognitive radio." Microwave Conference, 2007. APMC 2007. Asia-Pacific. IEEE, 2007

[7] Mikaeil AM, Guo B, Bai X, Wang Z. Hidden Markov and Markov switching model for primary user channel state prediction in cognitive radio. IEEE 2nd International Conference on Systems and informatics (ICSAI). 2014:854-859

[8] Kieu-Xuan T, Koo I. A cooperative spectrum sensing scheme using fuzzy logic for cognitive radio networks. KSII Transactions on Internet and Information Systems (TIIS). 2010; **4**(3):289-304

[9] Khaira ND, Bhadauria P. Cooperative spectrum sensing and detection efficiency in cognitive radio network. International Journal of Electronics and Computer Science Engineering (IJECSE, ISSN: 2277–1956)1. 2012;**01**:64-73

[10] Qin Q, Zhimin Z, Caili G. A study of data fusion and decision algorithms based on cooperative spectrum sensing. Fuzzy Systems and Knowledge Discovery, 2009. FSKD'09. Sixth International Conference on. Vol. 1. IEEE, 2009

[11] Bermejo S, Cabestany J. Adaptive soft k-nearest neighbor classifiers. Pattern Recognition. 2000;**33**(12):1999-2005

[12] Metsis V, Androutsopoulos I, Paliouras G. Spam filtering with naive bayes-which naive bayes? CEAS. 2006

[13] Thilina KM et al. Pattern classification techniques for cooperative spectrum sensing in cognitive radio networks: SVM and W-KNN approaches. Global Communications Conference (GLOBECOM), 2012 IEEE, 2012

[14] Barros RC et al. A survey of evolutionary algorithms for decision-tree induction. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on. 2012; **42**(3):291-312

[15] Rabiner L. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE. 1989;**77.2**:257-286

[16] Welch LR. Hidden Markov models and the Baum-Welch algorithm. IEEE Information Theory Society Newsletter. 2003;**53**(4):10-13

[17] Frühwirth-Schnatter S. Finite Mixture and Markov Switching Models: Modeling and Applications to Random Processes. Springer; 2006

[18] Kuan C-M. Lecture on the Markov switching model. Institute of Economics Academia Sinica. 2002 available at homepage.ntu.edu.tw/~ckuan/pdf/Lec-Markov_note.pdf

[19] Hamilton JD. Time Series Analysis. Vol. 2. Princeton: Princeton university press; 1994

[20] Kim C-J. Dynamic linear models with Markov-switching. Journal of Econometrics. 1994; **60.1**:1-22