Artificial Neural Networks, Support Vector Machine and Energy Detection for Spectrum Sensing based on Real Signals

Mohammed Saber[1,2], Abdessamad El Rharras[1], Rachid Saadane[1], Hatim Kharraz Aroussi[2] and Mohammed Wahbi[1]

[1]Laboratory Engineering system, SIRC/LAGeS-EHTP, Hassania School of Public Works, Casablanca, Morocco
[2]Laboratory Information Modeling and Communication Systems (IMCS), Ibn Toufail University, Kenitra, Morocco

Abstract: A Cognitive Radio (CR) is an intelligent wireless communication system, which is able to improve the utilization of the spectral environment. Spectrum sensing (SS) is one of the most important phases in the cognitive radio cycle, this operation consists in detecting signals presence in a particular frequency band. In order to detect primary user (PU) existence, this paper proposes a low cost and low power consumption spectrum sensing implementation. Our proposed platform is tested based on real world signals. Those signals are generated by a Raspberry Pi card and a 433 MHz Wireless transmitter (ASK (Amplitude-Shift Keying) and FSK (Frequency-Shift Keying) modulation type). RTL-SDR dongle is used as a reception interface. In this work, we compare the performance of three methods for SS operation: The energy detection technique, the Artificial neural network (ANN) and the support vector machine (SVM). So, the received data could be classified as a PU or not (noise) by the ED method, and by training and testing on a proposed ANN and SVM classification model. The proposed algorithms are implemented under MATLAB software. In order to determine the best architecture, in the case of ANN, two different training algorithms are compared. Furthermore, we have investigated the effect of several SVM functions. The main objective is to find out the best method for signal detection between the three methods. The performance evaluation of our proposed system is the probability of detection (P_d) and the false alarm probability (P_{fa}). This Comparative work has shown that the SS operation by SVM can be more accurate than ANN and ED.

Keywords: Cognitive radio, spectrum sensing, energy detection, artificial neural networks (ANN), support vector machine (SVM), Raspberry Pi 3, 433 MHz Wireless transmitter, RTL-SDR.

1. Introduction

The already existed radio resources are becoming increasingly in demand, because of the wireless communication emergence (wireless technologies and its various services). Under a static frequency spectrum allocation, a study carried out by the Federal Communications Commission (FCC) [1] has shown that the frequency spectrum use is not regular; according to the different times and the geographical position; some frequencies bands are not occupied or partially occupied and others are highly demanded [1, 2], the activity is concentrated on cellular radio and FM bands. The unused frequencies have been termed as spectrum holes. A spectrum hole is a region of spatiotemporal frequencies assigned to a licensed user (primary users 'PU'), but, at a particular time and specific geographic location, the band is not being utilized by the PU wherein a secondary and unique use is possible. As a conclusion of this previous study, FCC recommends an efficient spectrum management and access to enhance the spectrum efficiency, and specifies that unlicensed devices should have the capability of identifying free bands.

In order to reduce the waste of spectrum resources and increase the spectral efficiency, the concept of cognitive radio (CR) was introduced and proposed by Joseph Mitola in 1999 [3, 4]. The CR is defined as an intelligent device that senses the spectrum holes and makes it available for unlicensed users (secondary users (SU)). In CR the SUs can take advantage of these spectral holes dynamically and opportunistically, without causing any harmful interference to PUs. CR is a concatenation of software defined radio and artificial intelligence, with the aim of providing an efficient spectrum usage, also defined by FCC as: "A radio or system that senses its operational electromagnetic spectrum environment and can dynamically and autonomously adjust its radio operating parameters to modify system operation". Figure 1 illustrates a basic cognition cycle model as introduced by Haykin [5].

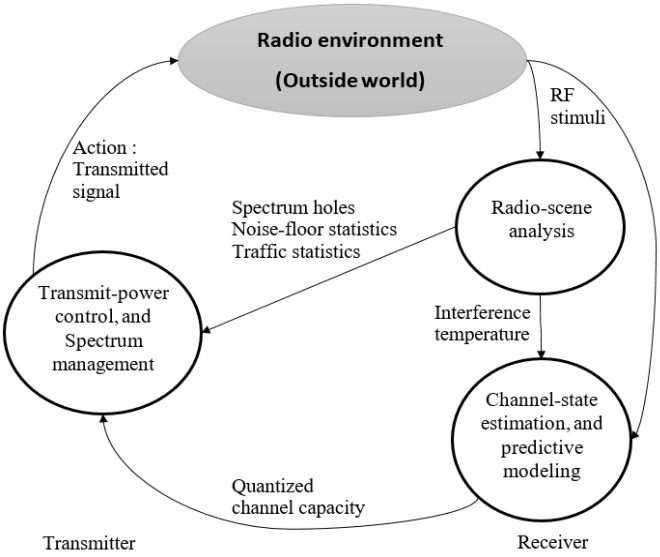

Figure 1. Basic cognitive cycle. (The figure focuses on three fundamental cognitive tasks.)

Spectrum sensing (Radio-scene analysis) is an essential capability of CR. The objective is to sense the spectrum holes in order to obtain the state of the band (free/occupied). The various spectrum sensing methods are discussed in [6]; Matched filter detection technique [7, 8], Energy detection techniques [9, 10] and cyclostationary feature detection technique [11, 12]. The energy detection (ED) remains the most used method for spectrum sensing [13], due to its simple implementation and not requiring any information about the PU signal. Therefore, in this work, we are interested in the ED method, in which the energy of the received signal is measured and compared with a predetermined threshold, which presents the noise energy present in the channel. If the

signal energy exceeds the threshold, we declare the presence of the PU, otherwise it is absent.

During the last years soft computing techniques like artificial neural networks (ANN) and support vector machine (SVM), have become extremely successful discriminative approaches to pattern classification [14-20]. In our context, we propose an implementation of ANN and SVM for SS operation to detect the PU signal; we focus on different ANN training algorithms and SVM functions that can be applied on the set of input data patterns. Our proposed platform as described in Fig. 2: transmits a real ASK/FSK signals using the Raspberry Pi 3 card and the 433 MHz transmitter. The transmitted signals are received by RTL-SDR dongle connected to MATLAB software environment. The received signals are treated as features and fed into a detector to sense the PU availability, using energy detection method, ANN and SVM.

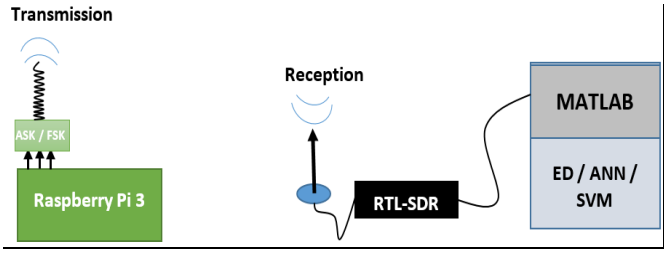

Figure 2. The proposed system

The rest of this paper is organized as follows: Section II presents the ASK and FSK radio signal transmitter based on Raspberry Pi 3 card and 433 MHz Wireless transmitter. Section III describes the mathematical formulation of classical energy detection method and explains the MATLAB implementation of this detector. Section IV gives more detailed description of the artificial neural network and the ANN detector architecture. Section V presents the theoretical foundations of SVM, the mathematical formulation and the implementation of SVM is presented. Section VI presents the performance evaluation results for the proposed SS implementation. Lastly, Section VII concludes the paper.

2. The used Data base

2.1 Database generation:

In order to make our work more authentic, we will generate our own signals $x(t)$ using a Raspberry Pi 3 card and a 433 MHz Wireless transmitter (ASK/FSK).

Figure 3. Raspberry Pi 3 card

The Raspberry Pi 3 (fig. 3) card is a low-cost, basic computer that was originally intended to help spur interest in computing among school-aged children. It allows running multiple variants of free operating systems GNU / Linux and compatible software. One of its great points is that it has GPIO connectors (General Purpose Input Output). This GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes. In this work, we will use the GPIO 4 connector (pin 7) for transmitting signal data to 433 MHz transmitter.

Figure 4. The 433 MHz ASK/FSK transmitter devices

The 433 MHz ASK/FSK transmitter (fig. 4) module is a small electronic device, it is used to transmit radio signals between two devices, and widely used in remote control, wireless data transfer applications, mobile robots and burglar alarms. The Specifications of the used 433MHz device, are as follows: the receiver operating voltage: 3V to 12V, the receiver operating current is 5.5mA, the operating frequency is 433 MHz, the Transmission Distance: 3 meters (without antenna) up to 100 meters, the modulating technique: ASK (Amplitude shift keying) / FSK (Frequency-Shift Keying), the data transmission speed: 10Kbps, and has a Low cost and small package.

2.2 Database acquisition:

The receiver reconstructs the message sent from the captured signal by the inverted processing operations done on transmission. Those processes can be performed using a single RTL-SDR hardware and MATLAB software. RTL-SDR (fig. 5) or Software Defined Radio is a radio communication system, wherein the traditional hardware components (e.g. mixers, filters, amplifiers, modulators/demodulators, detectors, etc.) are instead implemented by means of software on an embedded system [21]. RTL-SDR is capable of receiving any signal in its frequency range. This range varies depending on the type of device used. In this work, the used dongle has a frequency capability of approximately 25MHz-1750MHz.

Figure 5. RTL-SDR dongle

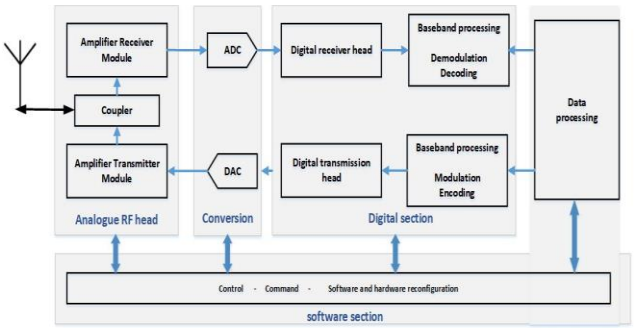

Figure 6. Synoptic diagram of SDR card

Fig. 6 shows the block diagram of the different processing stages of an RTL-SDR dongle. It is composed of:

- A configurable analog RF head, consisting of filters, couplers, mixers, intermediate frequency local oscillators, broadband and low noise power amplifiers,
- An analog / digital (ADC) and digital / analogue (DAC) conversion stage,
- A programmable digital section for shaping the spectrum, adapting and digital baseband processing,
- Software section providing control and software configuration of the various stages.

3. The Energy Detection

3.1. Energy detection Model

The problem of spectrum sensing operation can be mathematically formulated as follows:

$$\begin{cases} y(t) = n(t) & : H_0 \\ y(t) = \varepsilon * x(t) + n(t) & : H_1 \end{cases} \quad \text{Where} \quad 0 < \varepsilon \leq 1 \qquad (1)$$

Where:

$y(t)$: The received signal.

$x(t)$: The signal to be detected, deterministic or random, but unknown.

$n(t)$: The presented noise in the channel.

H0 is the hypothesis that the PU is not transmitting, therefore $x(t) = 0$, while H1 is the hypothesis that the PU is using the channel for transmission. In the energy detection technique, we receive the signal $y(t)$, next we measure its energy by (2). The detection of the primary signal is done by comparing the measured energy with a threshold λ that presents the noise energy.

$$E = \frac{1}{T}\sum_{n=1}^{N}[Y(n)]^2 \qquad \begin{cases} E > \lambda & H_1 \\ E < \lambda & H_0 \end{cases} \qquad (2)$$

3.2. Energy detector implementation

The energy detection implementation is presented in Fig. 7. First, the received signal y (t), by RTL SDR dongle, is digitized by an analog to digital convertor (ADC) then passes through a band-pass filter (BPF), with a center frequency f0 and bandwidth W, using the transfer function (3) to select the desired band.

$$H(f) = \begin{cases} \frac{2}{\sqrt{N_0}} & |f - f_0| \leq W \\ 0 & |f - f_0| > W \end{cases} \qquad (3)$$

Then the filtered signal is transformed into frequency domain through the fast Fourier transform (FFT) block, and the signal energy is measured using (4).

$$E = \frac{1}{N}\sum_{i=1}^{N}|Y(f_i)|^2 \qquad (4)$$

Finally, the estimated energy E is compared with a threshold λ (the noise energy) to decide if a signal is present (H_1) or not (H_0).

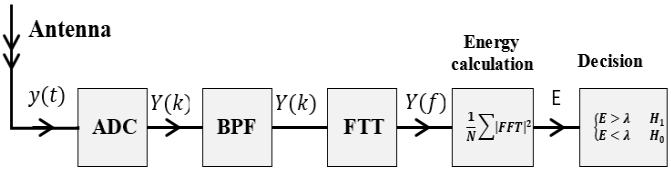

Figure 7. Block diagram of a frequency domain energy detector

4. Artificial Neural Networks

4.1. The biological neuron

The biological neuron is a special biological cell that processes information. According to [22], there are huge number of neurons in the human brain, approximately 10^{11}, each neuron is connected to 10^3 up to 10^4 other neurons. In total, approximately 10^{14} to 10^{15} interconnections. As shown in the fig. 8, a typical neuron mainly consists of the following three parts:

- The dendrites, which are the inputs of the neuron, collect the electrical information from the nervous system.
- The soma that processes this information and sends back an electrical signal of impulse type.
- The axon through which the outgoing signal is transmitted to neighboring neurons.

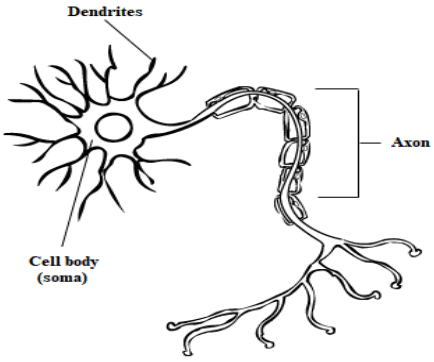

Figure 8. The biological neuron

4.2. The formal neuron

A formal neuron (fig. 9) is a mathematical function; it is conceived as a model of biological neuron. Formal neurons are elementary units in an artificial neural network. The following diagram represents the general mathematical model of a formal neuron [23]:

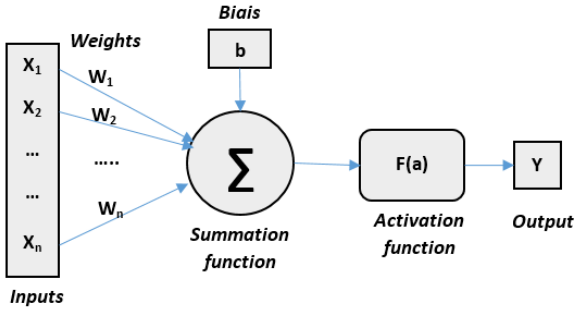

Figure 9. Mathematical model of the formal neuron

The formal neuron that is given in the figure above has n inputs denoted as $\{X_1, X_2, \ldots, X_n\}$. Each line that connects these inputs to the summation junction is assigned a weight denoted as $\{W_1, W_2, \ldots, W_n\}$. The net input y_{in} can be calculated as follows:

$$y_{in} = x_1.w_1 + x_2.w_2 + x_3.w_3 + \ldots + x_n.w_n + b \qquad (5)$$

The activation function $F(a)$ is One of the most important parts of a neuron. Several activation functions can be considered (threshold function, linear function, sigmoid function …). In this work, we have chosen a sigmoid function (fig. 10), for its nonlinearity that makes it possible to approximate any function. Finally, the output $\boldsymbol{y}$ of the neuron is given in the following formula:

$$y = F(y_{in}) = F\left(\sum W_i * X_i + b\right) \quad (6)$$

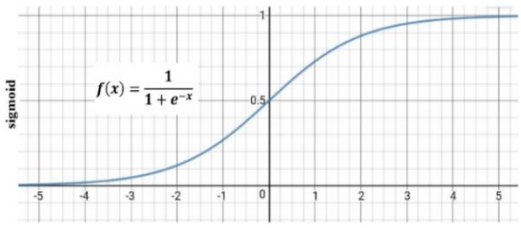

Figure 10. sigmoid function

4.3. Multi-Layer Perceptron

The multilayer perceptron (MLP) (fig. 11) is a class of feedforward artificial neural network that has at least three layers of nodes. It generates a set of outputs $\{y_1, y_2, \dots, y_m\}$ from a set of inputs $\{X_1, X_2, \dots, X_n\}$. Except for the input nodes, each node is a neuron that uses a nonlinear activation function.

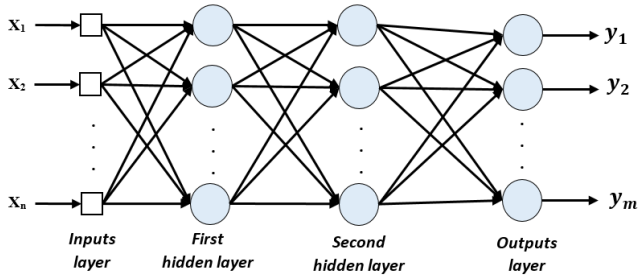

Figure 11. MLP model

A neural network is trained with input and target pair patterns with the ability of learning. MLP can separate data that is not linearly distinguishable [24]. It is especially trained using a supervised learning technique called back-propagation (PB) algorithm [25], which aims at minimizing the global error measured at the output layer by the relation bellow:

$$e(t) = y_d(t) - y_m(t) \quad (7)$$

Where $y_d(t)$ denotes the desired output, and $y_m(t)$ the measured output of the neuron.

The BP algorithm uses an iterative supervised learning procedure, where the MLP is trained with a set of predefined inputs and outputs. The global error $E_g(t)$ is calculated by equation (8), this error can be minimized by the gradient descent technique [22].

$$E_g(t) = \frac{1}{2}\sum_{i=1}^{n}\left(y_{d,i}(t) - y_{m,i}(t)\right)^2 \quad (8)$$

There are several training algorithms that can be used to train an MLP network. In this paper, we will present a qualitative comparison between two training algorithms: quasi newton and conjugate gradient. Wherein the used training functions are respectively: trainlm (Levenberg Marquardt (LM)) and trainscg (Scaled Conjugate Gradient (SCG)). All these algorithms are trained by the same data set acquired from the implementation described in section II.

4.4. ANN implementation

We have proposed an ANN detector for spectrum sensing (Fig. 12), the ANN detector architecture is similar at the ED detector (Fig. 7), where the energy calculation block is replaced by an ANN block (Fig. 13).

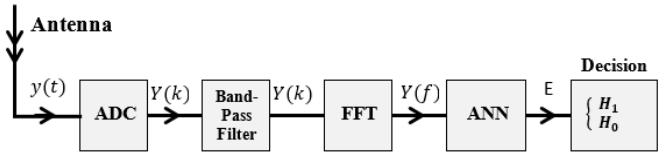

Figure 12. ANN detection diagram

The number of input layer neurons n, is fixed by the points number (features) in the captured signal FFT vector. which is 1024 and we use one (1) neuron in the output layer (H_1 or H_0):

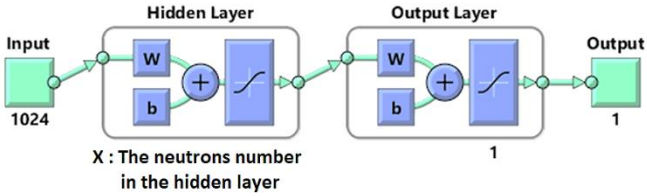

Figure 13. The ANN architecture

5. Support vector machines

Support vector machines (SVMs) are a new discriminating techniques in the theory of statistical learning, introduced in 1995s by V. Vapnik in his book " The Nature of Statistical Learning Theory "[26]. SVMs are machine learning methods that can be used to separate two classes of data [27, 28] by finding an optimal hyperplane 'H_0'. This technique is essentially used for binary classification, but possible to classify samples with multiple classes. In addition, it can be used to solve both linear and nonlinear classification or regression problems. In SVMs, each input instance x, is represented by a pair (x_i, y_i), where $x_i \in \mathbb{R}^n$ is the data instance, and $y_i \in \{1, -1\}$ is the binary class label (positive or negative). The training data can be defined as:

$$D = \{(x_i, y_i) \in \Re^n \ , \ i = 1, \dots, m\} \quad (9)$$

The hyperplane 'H' can be defined by equation (10). Where the vector 'ω' is the weighing vector that defines the boundary of different classes of data, and 'b' is a scalar threshold.

$$H : (\omega . x) + b \quad (10)$$

The objective of SVM classification is to predict the value of y_i for new data points x_i. There are two types in SVM classification: Linearly and non-linearly separable classification.

5.1. Linearly separable classification:

In this section, we present the general method of constructing the optimal hyperplane (OH), which separates data belonging to two different linearly separable classes. Fig. 14 gives a visual representation of the OH (H_0) in the case of linearly separable data, which is satisfying in the following conditions:

$$\begin{cases} \omega * x_i + b \geq 1 & if \quad y_i = 1 \\ \omega * x_i + b \leq -1 & if \quad y_i = -1 \end{cases} \quad (11)$$

That is equivalent to the next representation:

$$y_i(\omega \cdot x_i + b) \geq 1 \quad , \quad i = 1, \dots \dots, m \quad (12)$$

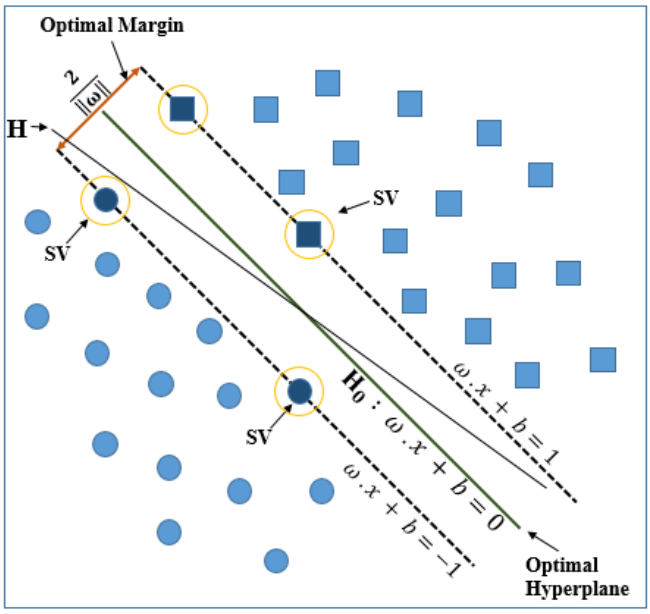

Figure 14. Separator hyperplanes: H is any hyperplane, Ho is the optimal hyperplane and M is the margin that represents the distance between the different classes and Ho (SV are the Supports Vectors)

The optimal hyperplane H_0 is a hyperplane that maximizes the margin M, which represents the smallest distance between the different data of the two classes and H_0. Maximizing the margin M is equal to maximizing the sum of the distances between the two classes relative to H_0. The margin 'M' has the following mathematical expression:

$$M = \min_{x_i | y_i = 1} \frac{\omega \cdot x + b}{\|\omega\|} - \max_{x_i | y_i = -1} \frac{\omega \cdot x + b}{\|\omega\|} \quad (13)$$

$$= \frac{1}{\|\omega\|} - \frac{-1}{\|\omega\|}$$

$$M = \frac{2}{\|\omega\|} \quad (14)$$

The optimal hyper plane can be obtained by maximizing the equation (14). Which is equivalent to minimizing (15):

$$\min_{\omega} \frac{\|\omega\|^2}{2} \quad (15)$$

The Equation (15) can be solved as a quadratic optimization problem by Lagrangian function:

$$L(\omega, b, \alpha) = \frac{1}{2}\|\omega\|^2 - \sum_{i=1}^{m} \alpha_i \left[y_i (\omega \cdot x_i + b) - 1 \right] \quad (16)$$

Where $\alpha_i = (\alpha_1, \dots, \alpha_m) > 0$ is a lagrangian multiplier factors.

By deriving the equation (16) we obtain:

$$\omega = \sum_{i=1}^{m} \alpha_i \, y_i x_i \quad (17)$$

$$\sum_{i=1}^{m} \alpha_i \, y_i = 0 \quad (18)$$

Substituting (17) and (18) into Equation (16), the optimal separating hyperplane can be obtained by solving the following dual representation of the optimization problem:

$$\min_{\alpha} \quad \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j (x_i x_j) \alpha_i \alpha_j - \sum_{j=1}^{m} \alpha_j \quad (19)$$

$$\text{Subject to} \quad \sum_{i=1}^{m} \alpha_i \, y_i = 0 \quad , \quad \alpha_i > 0$$

By solving this dual Lagrange function (19), α is evaluated. Consequently, ω is evaluated out from (17), and b can be easily calculated from (20):

$$b = y_j - \sum_{i=1}^{m} \alpha_i y_i (x_i x_j) \quad (20)$$

Knowing that the classification function is defined by (21), where sgn is a Signum function.

$$class(x) = sgn(\omega \cdot x_i + b) \quad (21)$$

Finally, we can classify an unknown data x by utilizing the following function:

$$class(x) = sgn\left[\sum_{i=1}^{m} \alpha_i y_i (x_i x) + b \right] \quad (22)$$

5.2. Non-linearly separable classification:

If the classes of data are mixed (not linearly) (fig. 15), it is impossible to linearly separate the training data in the original space. In order to solve this non- linearly separable problem, the slack variables ξ_i, which cause little change around training data, are introduced. Then the training vectors must satisfy:

$$y_i (\omega \cdot x_i + b) \geq 1 - \xi_i \quad , \quad i = 1, \dots \dots, m \quad (23)$$

In the case when the data points are not linearly separable in the data space, the SVM handles this by using a kernel function to map the data to a transformed feature space (a higher dimensional space) where a hyperplane is then used to gain linearly separation. For this, we project the original data into the feature space using non-linear functions. In this new space we build the OH that separates the transformed data. The main problem here is how to manipulate the transformation of all input vectors in the feature space, so as to avoid an increase in the cost in number of free parameters. Let the set D' be the image (the transformed feature space) of the set D defined in previous section, by the transformation:

$$D' = \{\psi(x_i), y_i) \in \Re^n \times \{-1,1\}, i = 1, \dots, m | \, p \geq n \} \quad (24)$$

Structuring an optimal hyperplane, defined by the weight vector ω and the bias b is solved as a quadratic optimization problem that maximizes the margin between the classes (The first part of Eq. 25), and minimizes the errors (The second part of Eq. 25). It can be expressed as:

$$min \left[\frac{1}{2}\|\omega\|^2 + C \sum_{i=1}^{m} \xi_i \right] \quad (25)$$

$$\text{Subject to} \quad \begin{cases} y_i (\omega \cdot x_i + b) \geq 1 - \xi_i \\ \xi_i \geq 0 \, ; \; i = 1, \dots \dots, m \end{cases}$$

Where C is a soft margin constant that is used to control the training error rate by different values. For high values of C, the optimization uses a smaller-margin hyperplane if that hyperplane classified all the training points correctly. Conversely, a very small value of C will make the optimizer use a higher-margin separating hyperplane in the case when the hyperplane misclassifies more points.

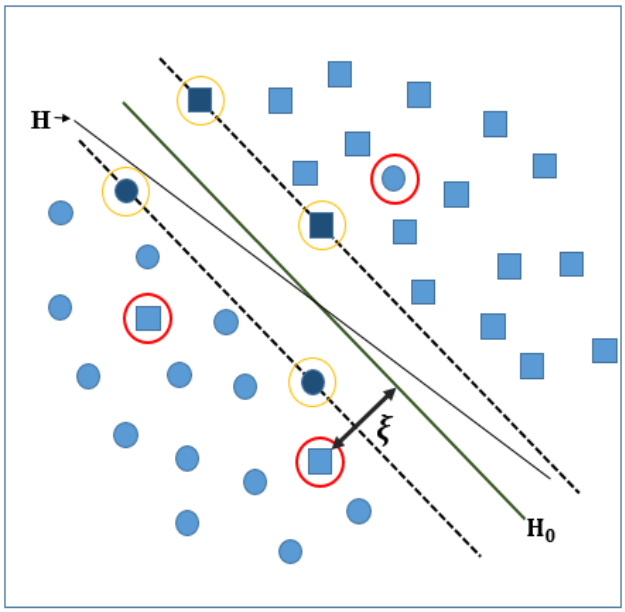

Figure 15. Separator hyperplanes in the case of non-linearly separable (ξ is the slack variables)

Using Lagrange multiplier technique, the above optimization problem (25) can be reformulated as:

$$L(\omega, b, \xi, \alpha, \beta) = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\beta_i\xi_i - \sum_{i=1}^{m}\alpha_i\,[y_i(\omega^T\psi(x_i) + b) - 1 + \xi_i] \quad (26)$$

Where α_i and β_i are positive Lagrangian multipliers parameters that can be found by solving the quadratic programming problem (26). The obtained vector α is known as a support vector. By applying Karush-Kuhn-Tucker (KKT) conditions, which is a theorem that plays an important part in the theory of optimization, to (26), we can obtain:

$$\omega = \sum_{i=1}^{m}\alpha_i y_i \psi(x_i) \quad (27)$$

$$\sum_{i=1}^{m}\alpha_i y_i = 0 \quad (28)$$

$$\alpha_i = C - \beta_i \quad (29)$$

It is noticeable that $0 \le \alpha_i \le C$ and $\beta_i \ge 0$.

By calculating the derivatives with respect to ω, b and ξ, the dual representation of the optimization problem in term of support vectors can be obtained as follows [28].

$$\max \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j\, y_i y_j K(x_i, x_j) \quad (30)$$

Subject to

$$\begin{cases} \sum_{i=1}^{m}\alpha_i\, y_i = 0 \\ 0 \le \alpha_i \le C\ ;\ i = 1, \dots\dots, m \end{cases} \quad (31)$$

Where $K(x_i, x_j) = \psi(x_i)\psi(x_j)$ is the kernel function. Hence, by solving the optimization function defined in (30), the resulting nonlinear classification function can be obtained as follows:

$$class(x) = sgn\left[\sum_{x_i \in VS}\alpha_i\, y_i K(x_i, x) + b\right] \quad (32)$$

$\psi(x_j)$ The most common kernel, that are widely used because of their efficiency in mapping the input data to higher dimensional space in order to reduce the computational load, are illustrated as follows (u and v are vectors) [28].

– The linear kernel:

$$K(u, v) = u.v \quad (33)$$

– The Polynomial kernel:

$$K(u, v) = [(u.v) + 1]^d \quad (34)$$

– The RBF kernel (Radial Basis Function):

$$K(u, v) = exp\left(-\frac{\|u - v\|^2}{2\sigma^2}\right) \quad (35)$$

5.3. SVM implementation:

Support Vector Machine (SVM) model is a powerful supervised machine learning, it is used for efficient classification with high precision in various applications like object detection, speech recognition, bioinformatics, image classification, medical diagnosis and others [29]. In our context, we have proposed an SVM detector diagram for signal classification (spectrum sensing) (Fig. 16). Our SVM detector have the same steps of the ANN detector (fig. 12), we have just changed the ANN block by an SVM block.

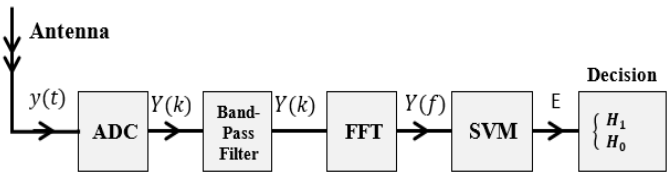

Figure 16. SVM detection diagram

The SVM kernel choice is critical to define the flexibility and classification power. The most used kernels are: linear, polynomial degree 'p', and Gaussian. In MATLAB software, we can find three principle SVM functions for classification [30]: "fitcsvm", "fitclinear" and "fitckernel".

– The *Fitcsvm* function (fit a classifier using SVM), trains a support vector machine (SVM) model for binary classification (two classes), on a low-dimensional or moderate-dimensional predictor data set.

– The *fitclinear* (Fit linear classification) function trains linear classification models for two-class (binary) learning with high-dimensional. This function minimizes the objective function using techniques that reduce computing time (e.g., stochastic gradient descent).

– The *fitckernel* (fit classifier kernel) function trains a binary Gaussian kernel classification model for nonlinear classification. it is more practical for big data applications that have large training sets, but can also be applied to smaller data sets. This function maps data in a higher dimensional space then fits a linear model in the new space by minimizing the regularized objective function.

6. Implementation and Results

The database used in our spectrum sensing implementation, was generated by the Raspberry Pi 3 card and the 433 MHz Wireless transmitter, then captured by RTL-SDR dongle for different distances between the transmitter and the receiver.

The collected database contains 1600 signals with ASK and FSK modulation type. The proposed work is implemented in MATLAB 9.4.0 (R2018a) for a 64-bit computer with core i3 processor, clock speed 2.4 GHz, and 6 GB RAM.

Table 1 is showing how we have used our database in the learning phase and test phase:

Table 1. Data-base used in learning and testing

Signal	Learning phase	Test phase
Primary signal	600	300
Noise	400	300

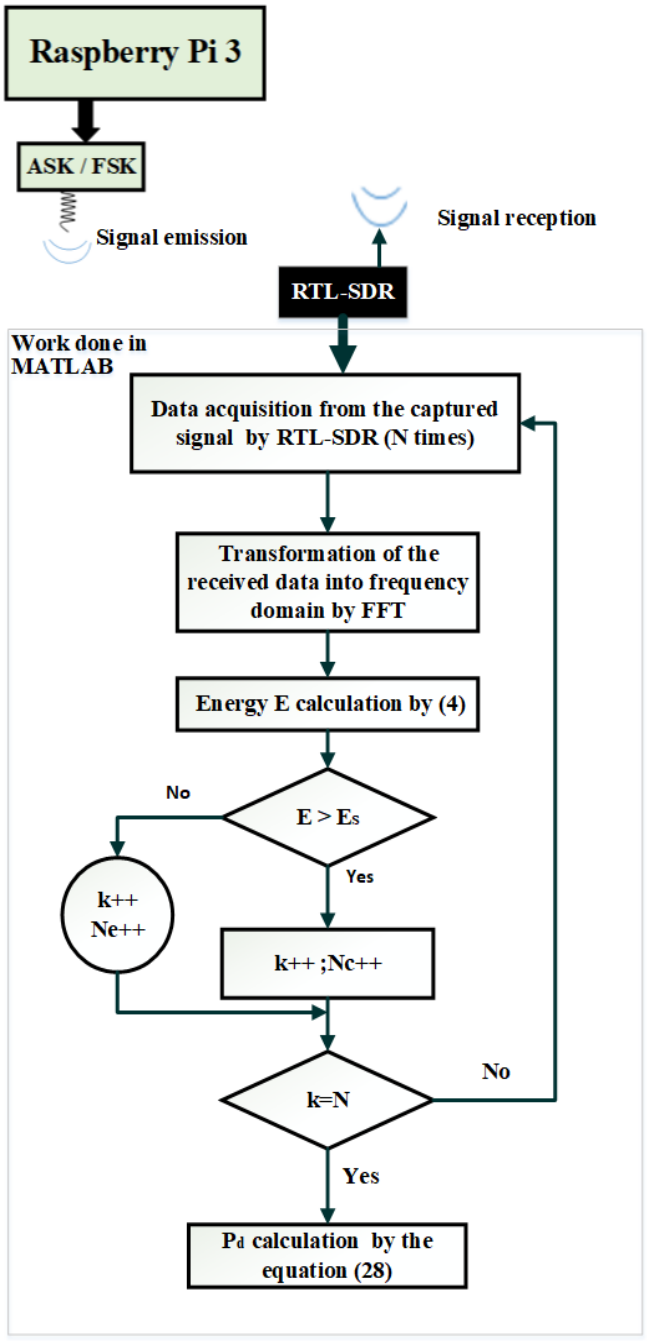

Figure 17. Chart of the energy detection operation

The performance of an energy detector can be characterized using two probabilities: P_d the probability of detection and P_{fa} the false alarm probability.

P_d : The probability of detecting a signal in the band of interest, when this signal is truly present (H_1). Failed detection causes interference with the PU. We calculated P_d by (36):

$$P_d = P\,(decision\ H_1/H_1) = \frac{Nc}{N} \times 100 \qquad (36)$$

P_{fa} : The probability when the test falsely decides that the band of interest is occupied, while it is free (H_0). The False alarm reduces the efficiency of spectrum use. We calculated P_{fa} by (37):

$$P_{fa} = P\,(decision\ H_1/H_0) = \frac{Ne}{N} \times 100 \qquad (37)$$

Where:

N_c : The number of times in which the signal is detected, while H_1;

N_e : The number of times in which the signal is detected, while H_0;

N : The number of the captured signals.

6.1. Energy Detection results:

The detection of the transmitted signal, using the energy detection method, is done in MATLAB software according to the flowchart of Fig. 17. We have as an input a frequency vector that contains 'N' signals data, which is captured by the RTL-SDR.

Following this flowchart, we calculate the probability of detection P_d . While we calculate the false alarm probability P_{fa} by stopping the emission operation and following the same previous steps using (37) instead of (36). The values of $\boldsymbol{P_d}$ and $\boldsymbol{P_{fa}}$ obtained through the energy detector (fig. 7) are shown in the table 2:

Table 2. The probability of detection and the false alarm probability of ED

$\boldsymbol{P_d}$	$\boldsymbol{P_{fa}}$
0.99	0,015

6.2. Artificial neural network results

Supervised learning methods are normally composed of two main phases: training/learning, and classification. To determine the most convenient value of the neurons number in the hidden layer, we have tested experimentally several architectures, with different hidden layer size for each training function. We measure the error on training with mean squared error (MSE). We have used Matlab Neural Network Toolbox in order to build and train our network. Basic system training parameters 1000 maximum training epochs, 6 validation checks, performance goal=0, time=Inf, min_grad=1e-010 and max_fail=10 are fixed for each training function. Table 3 summarizes the results obtained for different training functions with different architectures.

Table 3. Probability of detection and the false alarm probability in testing phase for ANN detector

Number of neurons in the hidden layer	Training function	P_d	P_{fa}
3	SCG	0.983	0.017
	LM	0.963	0.020
4	SCG	0.973	0.013
	LM	0.966	0.043
5	SCG	0.980	0.0167
	LM	0.960	0.046
6	SCG	0.970	0.0167
	LM	0.973	0.0167
7	SCG	0.963	0.0267
	LM	0.960	0.023
8	SCG	0.970	0.0131
	LM	**0.983**	**0.007**
10	SCG	0.970	0.0233
	LM	0.956	0.0267

The results show that the best performance is obtained for the training function 'trainlm' with 8 hidden neurons, in which the probabilities P_d and P_{fa} are presented in the following table.

Table 4. The probability of detection and the false alarm probability of ANN detection

P_d	P_{fa}
0.983	0.007

6.3. Support vector machines results

The SVM training phase builds a model for classifying any future data based on the Support Vectors (SVs) identified from a training dataset. The SVs are then used in the classification phase to predict the appropriate class of an input test data. In order to determine the most convenient SVM function for our SVM detector, we have compared the performance of the above three functions. The implementation results of the SVM detector (fig. 16) are presented in table 5, the following table shows the comparison of classification performance for the three SVM functions.

Table 5. Probability of detection and the false alarm probability in testing phase for SVM detector

Functions		P_d	P_{fa}
fitcsvm	linear	0.9433	0.040
	sigmoid	0.9400	0.403
	RBF	1	0.66
fitclinear		**0.9900**	**0**
fitckernel		0.9733	0

From this table we can see that the function "fitclinear" has shown high classification accuracy rates outperforming the other functions.

6.4. Discussion

Energy detection has the advantages of low complexity, ease of implementation, and extensive use of properties without knowing the signal. However, when there is a low signal noise ratio (SNR), there are many problems that decrease the performance of energy detection method, and make it hard to set the threshold. In the classic method, we use a static threshold, but as we know, the threshold depends on the environmental noise. Whereas the ANN detector and the SVM detector, give the best results and they have a stable performance in comparison with the classical energy detection method. The excellent results have been reported in applying SVM's in spectrum sensing.

7. Conclusions

In this paper, we have proposed different implementation of spectrum sensing based on real signal generated by Raspberry Pi 3 card and 433 MHz Wireless transmitter. The transmitted signal is detected in MATLAB software by RTL-SDR dongle using three methods: the energy detection method, the artificial neural (ANN) networks and support vector machines (SVM). The ANN networks implemented using the two training functions (SCG and LM), are affected according to the number of neurons in their hidden layer. The SVM algorithm is implemented using three SVM functions (fitcsvm, fitclinear and fitckernel). The proposed work is evaluated in terms of the probability of detection and the false alarm probability. The obtained results show that the spectrum sensing will be stable with ANN and SVM detectors, in comparison with the classical energy detection method. However, the SVM classifier achieves the highest detection performance compared to the other classifiers (ED and ANN).

References

[1] Federal Communications Commission, Spectrum Policy Task Force, “Report of the Spectrum Efficiency Working Group,” November 2002.
[2] P. Kolodzy, "Next generation communications: Kickoff meeting," in Proc. DARPA, vol. 10, 2001.
[3] J. Mitola G. Q. Maguire, "Cognitive radio: making software radios more personal," IEEE personal communications, vol. 6, No. 4, pp. 13-18, 1999.
[4] J. Mitola Iii, "Cognitive radio for flexible mobile multimedia communications," Mobile Networks and Applications, vol. 6, No. 5, pp. 435-441, 2001.
[5] S. Haykin, "Cognitive radio: brain-empowered wireless communications," IEEE journal on selected areas in communications, vol. 23, No. 2, pp. 201-220, 2005.
[6] T. Yucek H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," IEEE communications surveys & tutorials, vol. 11, No. 1, pp. 116-130, 2009.
[7] D. Cabric, S. M. Mishra, R. W. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in Signals, systems and computers, 2004. Conference record of the thirty-eighth Asilomar conference on, vol. 1, pp. 772-776, 2004.
[8] W. Ejaz, ul Hasan, N., Lee, S. et al. , "Intelligent spectrum sensing scheme for cognitive radio networks," EURASIP Journal on Wireless Communications and Networking, vol. 1, pp. 1-10, 2013.
[9] F. F. Digham, M.-S. Alouini, M. K. Simon, "On the energy detection of unknown signals over fading channels," IEEE International Conference on Communications, Anchorage, AK, USA, vol. 5, pp. 3575-3579, 2003.
[10] V. I. Kostylev, "Energy detection of a signal with random amplitude," IEEE International Conference on Communications, New York, USA, vol. 3, pp. 1606-1610, 2002.
[11] K. Kim, I. A. Akbar, K. K. Bae, J.-S. Um, C. M. Spooner, J. H. Reed, "Cyclostationary approaches to signal detection and classification in cognitive radio," 2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, Dublin, Ireland, pp. 212-215, 2007.
[12] Sai Shankar N, C. Cordeiro and K. Challapali, "Spectrum agile radios: utilization and sensing architectures," First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, Baltimore, MD, USA, pp. 160-169, 2005.
[13] S. Atapattu, C. Tellambura, H. Jiang, "Energy detection based cooperative spectrum sensing in cognitive radio networks," IEEE Transactions on wireless communications, vol. 10, No. 4, pp. 1232-1241, 2011.
[14] S. Ali, K. A. Smith, "On learning algorithm selection for classification," Applied Soft Computing, vol. 6, No. 2, pp. 119-138, 2006.
[15] K. M. Thilina, K. W. Choi, N. Saquib, E. Hossain, "Machine learning techniques for cooperative spectrum sensing in cognitive radio networks," IEEE Journal on selected areas in communications, vol. 31, No. 11, pp. 2209-2221, 2013.
[16] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, "Gene selection for cancer classification using support vector machines," Machine learning, vol. 46, No. 1-3, pp. 389-422, 2002.
[17] P. Singh, V. Pareek, A. K. Ahlawat, "Designing an Energy Efficient Network Using Integration of KSOM, ANN and Data Fusion Techniques," International Journal of Communication Networks and Information Security (IJCNIS), vol. 9, No. 3, 2017.
[18] B. Benmammar, Y. Benmouna, A. Amraoui, F. Krief, "A parallel implementation on a multi-core architecture of a dynamic programming algorithm applied in cognitive radio ad hoc networks," International Journal of Communication Networks and Information Security (IJCNIS), vol. 9, No. 2, 2017.
[19] A. Elrharras, R. Saadane, M. Wahbi, A. Hamdoun, "Neural Networks and PCA for Spectrum Sensing in the context of Cognitive Radio," Proceedings of the Mediterranean Conference on Information & Communication Technologies, pp. 173-181, 2016.
[20] A. Elrharras, R. Saadane, M. Wahbi, A. Hamdoun, "Signal detection and automatic modulation classification based spectrum sensing using PCA-ANN with real word signals," Applied Mathematical Sciences, vol. 8, No. 160, pp. 7959-7977, 2014.
[21] M. Dillinger, K. Madani, N. Alonistioti, Software defined radio: Architectures, systems and functions. John Wiley & Sons, 2005.
[22] A. K. Jain, J. Mao, K. M. Mohiuddin, "Artificial neural networks: A tutorial," Computer, vol. 29, No. 3, pp. 31-44, 1996.
[23] M. T. Hagan, H. B. Demuth, M. H. Beale,"Neural network design," Boston, 1996.
[24] G. Cybenko, "Approximation by superpositions of a sigmoidal function," Mathematics of control, signals and systems, vol. 2, No. 4, pp. 303-314, 1989.
[25] K. Tsagkaris, A. Katidiotis, P. Demestichas, "Neural network-based learning schemes for cognitive radio systems," Computer Communications, vol. 31, No. 14, pp. 3394-3404, 2008.
[26] V. Vapnik, The nature of statistical learning theory. Springer science & business media, 2013.
[27] C. J. Burges, "A tutorial on support vector machines for pattern recognition," Data mining and knowledge discovery, vol. 2, No. 2, pp. 121-167, 1998.
[28] C. Cortes V. Vapnik, "Support-vector networks," Machine learning, vol. 20, No. 3, pp. 273-297, 1995.
[29] J. Nayak, B. Naik, H. Behera, "A comprehensive survey on support vector machine in data mining tasks: applications & challenges," International Journal of Database Theory and Application, vol. 8, No. 1, pp. 169-186, 2015.
[30] https://www.mathworks.com/help/stats/support-vector-machine-classification.html