International Journal of Research Studies in Computer Science and Engineering (IJRSCSE)
Volume 7, Issue 1, 2020, PP 1-12
ISSN 2349-4840 (Print) & ISSN 2349-4859 (Online)
DOI: http://dx.doi.org/10.20431/2349-4859.0701001
www.arcjournals.org

A Robust Jamming Signal Classification and Detection Approach Based on Multi-Layer Perceptron Neural Network

Sahar Ujan*, Mohammad Hossein Same, Rene Jr Landry

École de technologie supérieure, 1100 Notre-Dame Street West, Montreal, QC H3C 1K3, Canada

Corresponding Author:** ***Sahar Ujan, *École de technologie supérieure, 1100 Notre-Dame Street West, Montreal, QC H3C 1K3, Canada*

Abstract: *The Quality of Service (QoS) and security of a wireless communication system can be improved using a robust Radio Frequency Interference (RFI) detection method, which could in turn lead to an effective mitigation process. Due to the vast variety of RFI signals in existence, it is crucial to deploy a spectrum signal detection technique dealing with all kinds of potential interference (non-intentional or intentional, known as jamming). To solve this problem, a robust multi-class Multi-Layer Perceptron (MLP) neural network technique has been developed in this paper to recognize jamming signals in a real-time digital video broadcasting scenario based on DVB-S2 standards. It is assumed that the communication signal can be combined with one of the three major kinds of interference, namely, Continuous Wave Interference (CWI), Multiple CWI (MCWI) and Chirp Interference (CI). The proposed algorithm can either classify the interference under its related group or recognize the Signal of Interest (SoI). Besides analyzing different learning approaches (such as online learning, full-batch, and mini-batch), the Principal Component Analysis (PCA) technique is deployed to select more informative features to enhance the classifier's performance. More importantly, a new methodology is presented to generate a real-time jamming signal dataset that is entirely different from what is provided in other similar studies. Furthermore, the robustness of the trained classifier is assessed to detect unknown signals at different Signal to Noise Ratios (SNR). In addition, the performance of the proposed design is compared with the well-known Support Vector Machine (SVM) Machine Learning (ML) technique in terms of both jamming signal classification and detection.*

Keywords: *Jamming classification, Multi-Layer Perceptron, Principal Components Analysis, Support Vector Machine,*

1. INTRODUCTION

Jamming is the intentional transmission of RFI in different wireless network systems. The main objective of a jammer is to corrupt radio communication networks, disrupting the receiver and preventing it from properly demodulating the received signal; the result is a degraded SoI reception [1]. Although jamming attacks can be quickly deployed without using any complicated devices, they are hard to detect, mitigate and/or eliminate [3]. Consequently, RFI can be considered as the most significant threat to communication networks. Designing a highly accurate detection technique capable of dealing with different types of signal jammers is thus an important research avenue [2]. As thoroughly discussed in [3], various known jamming signals exist, each requiring different mitigation techniques based on its specific characteristics. Classification-based techniques likely constitute the most efficient detection approach when it comes to tackling this problem. Another challenge in designing a detection technique is the ability to deal with weak jamming signals [3]. Recently, Artificial-Intelligence (AI)-based methods have been used for jamming detection. One such example is [2], which presents the Random Forest classifier (RFC) for jamming detection in a IEEE 802.11 wireless network, and compares the RFC performance to other classifiers, including the Decision Tree, Adaptive Boosting (AdaBoost), the Support Vector Machine (SVM) and Expectation Maximization (EM). As the results illustrate, the Random Forest and SVM slightly outperform other techniques. In [4], besides being used to analyze various feature selection techniques, SVM is deployed for GNSS Spoofing Jamming Recognition at different SNRs in the 1 to 10 dB range. An Artificial Neural Network (ANN) was proposed in [5] for jammer detection in wideband radios. To this end, spectral correlation was used as the feature extraction technique. According to the obtained results, the proposed technique performed efficiently on SNR values down to -3 dB. In [6], the K-means algorithm is used for jamming attack

detection in a pair of RF communicating vehicles. Moreover, the author presents some new features to enhance the accuracy of the classifier. Recently, in a few related works [7-12], MLP with different designs are presented to recognize different kinds of jamming attacks and Automatic Modulation Classification (AMC) [13, 14]. The focus of this work is not only to classify SoI and different types of jamming, but also to develop a jamming predictive model based on MLP to detect new instances generated in various channel conditions. In addition, a robust feature selection-based PCA is deployed for dimensionality reduction while preserving more informative features. More significantly, a new methodology is presented to generate a real-time jamming signal dataset entirely different from what is provided in other similar studies. Finally, the performance of the designed MLP is compared with SVM, which is a known robust technique for RFI classification in the literature. As the results show, our proposed model performs almost similarly to SVM.

2. JAMMING DETECTION METHODOLOGY

In the present research, our case study consists in analyzing jamming classification approaches in a real-time digital video broadcasting scenario based on DVB-S2 standards. To this end, our proposed methodology includes two main sections: 1) real-time data acquisition, and 2) classification. Both section follow.

3. REAL-TIME DATA ACQUISITION

As shown in Fig 1, the data used in this study is a real-time video stream, which is modulated and processed by GNU radio and transmitted using a Universal Software Radio Peripheral (USRP-N210) [15]. In GNU radio, the modulation type and amplitude of the transmitted signal can be easily adjusted [16]. A SatCom Emulator (RTLogic T400) [17] is used for modeling a real-time communication channel. The programmatic control of the channel simulator is facilitated over an Ethernet connection using a control protocol or optional plugin to STK software. The Channel Simulator produces IF/RF signals with extracting signal behavior for any scenario. The Kratos STK plugin provides real-time, phase-continuous control of the channel simulator when playing STK scenarios. Further, the jamming signals generated are transmitted using a NanoBee modem and are combined to SoI by a combiner. Finally, the combined signal is received by a MegaBee modem.

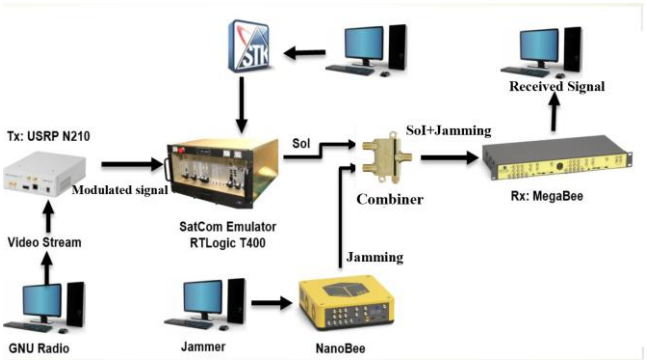

Fig1. *Real-time RFI data acquisition scenario*

In this study, the effect of the following three jamming signals on the signal classification efficiency is analyzed.

- **Continuous Wave Interference (CWI)**

$$CWI = \exp(j2\pi f_{cw} t) \quad (1)$$

Where f_{cw} and t represent the center frequency and the duration of interference, respectively.

- **Multi Continuous Wave Interference (MCWI)**

In this work, we consider a two-tone CW, defined as:

$$MCWI = \exp(j2\pi f_{c1} t) + \exp(j2\pi f_{c2} t) \quad (2)$$

Where f_{c1} and f_{c2} are the center frequencies of each wave.

- **Chirp Interference (CI)**

The CI was generated according to [18] as follows:

$$CI = \exp(2\pi \frac{k}{2} t^2 + 2\pi f_0 t) \quad (3)$$

Where $k = \frac{f_1 - f_0}{T}$ and the signal sweeps from f_0 to f_1 with the sweeping duration equal to T. The center frequencies were considered to be changed randomly.

Fig 2 demonstrates our test bench environment at the LASSENA Laboratory at École de technologie supérieure, in Montreal, Canada. As can be seen, there are five computers in use: PC 1 is connected to a NanoBee modem, and is used to generate jamming signals in MATLAB (Eq.1-3). PC 2 demonstrates the Power Spectral Density (PSD) of the signals received by the MegaBee modem (as shown in Fig 4 & Fig 5). The GNU radio is running on PC 3, and is connected to USRP N210 through an Ethernet cable. PC 4 and PC 5 include STK software for controlling the channel emulator's key parameters.

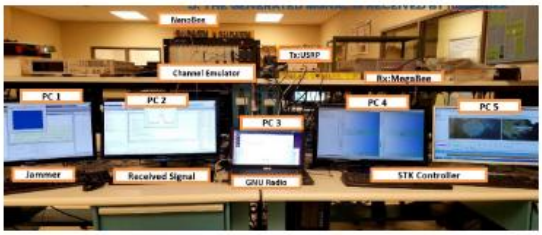

Fig2. *Test bench at LASSENA laboratory*

The shift in frequency due to the Doppler Effect is estimated within STK, and can be modeled in real time using the RT-Logic channel simulator. In this paper, we considered a direct satellite-to-ground communication in the Low Earth Orbit (LEO) scenario, in which the satellite is orbiting around the earth, and therefore is moving relative to the ground station. Fig 3 demonstrates the direct connection between the ground station (Telesat-Montreal-Teleport) and the satellite (CUBESAT-42707).

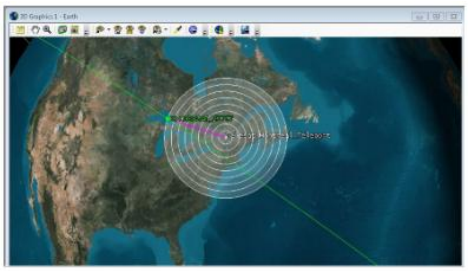

Fig 3. *Satellite-to-ground station in the STK software*

Fig 4 illustrates the PSD of the received signals in MATLAB. The generated signals' spectrum was validated using a signal analyzer, as shown in Fig 5.

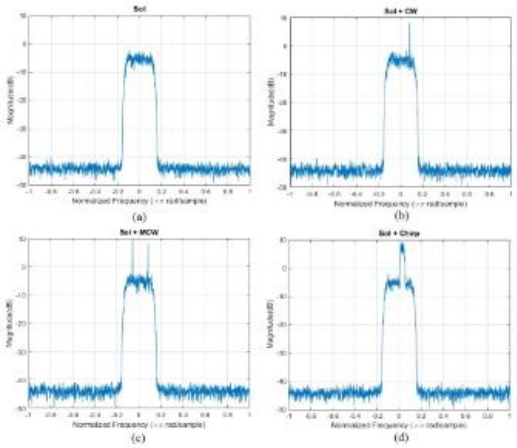

Fig4. *PSD of the received signals in MATLAB: a) SoI, b) SoI+CWI, c) SoI+MCWI, d) SoI+CI*

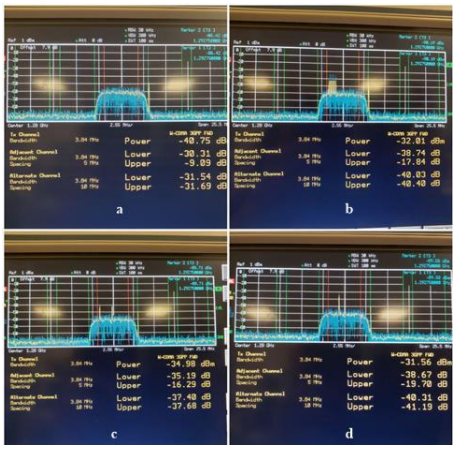

Fig5. *PSD of the received signals using the signal analyzer; a) SoI, b) SoI+CI, c) SoI+MCWI, d) SoI+CWI*

Measurements: As shown in Table 1, the power of the received signals (in dBm/Hz) was measured using a signal analyzer in a 3.84 MHz Bandwidth (BW):

Table1. *Measured power of received signals*

SoI	SoI+CWI	SoI+MCWI	SoI+CI
- 40 (dBm)	-33 dBm	-35 dBm	-32 dBm

The Jamming-to-Signal Ratio (JSR) is computed as:

$$JSR = Power_{SOI} - Power_{Jamming} \quad (4)$$

Therefore, the measured JSR for CWI, MCWI and CI is 7 dB, 5 dB and 8 dB, respectively. Table 2 shows the generated real-time dataset specification:

Table2. *Real-time dataset specification*

Total number of samples	4800
Length of each generated signal	32488 (8ms)
Sampling frequency	40×10^6 Hz
Modulation types	QPSK, 8APSK, 16APSK and 32 APSK

Notably, for each type of signal (SoI, CWI, MCWI and CI), the number of generated samples per modulation type is 300.

Supervised classification steps

As shown in Fig 6, we intend to design an intelligent receiver that can precisely recognize the type of the received signal. To this end, we use a supervised classification procedure which follows three steps: 1) feature extraction, 2) feature selection, and 3) classification. Each step will be covered in the rest of this section.

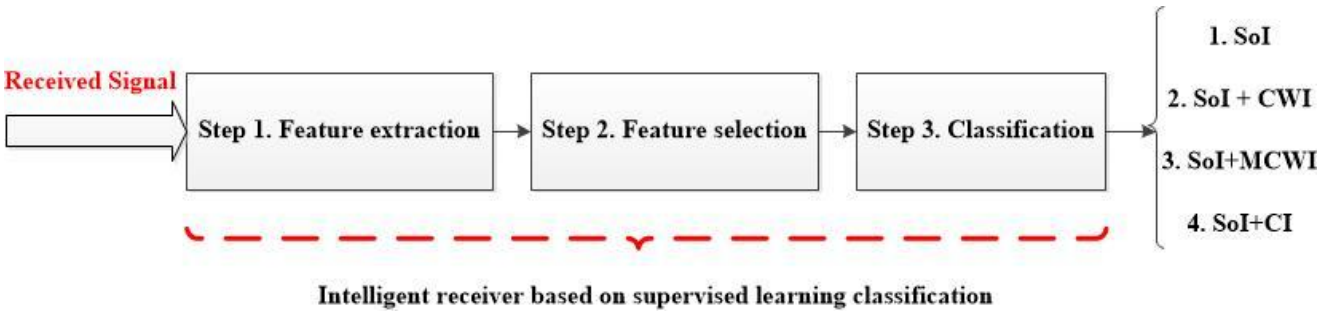

Fig6. *Proposed design of an intelligent receiver based on supervised learning classification*

Feature Extraction: As discussed in [19-21], there are various RFI features presented in similar RFI classification works. However, in this work, the following features were extracted from each received signal x with size 1 by n:

- **Mean**

$$\mu(x) = \frac{1}{N}\sum_{i=0}^{n-1} x(i) \quad (5)$$

- **Standard Deviation (σ):**

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n} |x(i) - \mu(x(i))} \quad (6)$$

- **Skewness (I/Q)**

As in [19], the skewness of a signal is computed as:

$$skewness = \frac{E(x-\mu(x))^3}{\sigma^3} \quad (7)$$

- **Real Signal Kurtosis (RSK)**

This feature was proposed in [21], where the kurtosis of the I and Q parts is computed, and finally the RSK is obtained by averaging the computed kurtosis (I/Q):

$$kurtosis = \frac{E(x-\mu(x))^4}{\sigma^4} \quad (8)$$

$$RSK = \frac{kurtosis_I + kurtosis_Q}{2} \quad (9)$$

- **Average Power**

According to [22], the average power of each signal over its length is calculated as:

$$AveragePower = \frac{1}{n}\sum_{i=1}^{n} |x(i)|^2 \quad (10)$$

Average Power of the Discrete Wavelet Coefficients

The wavelet transform is known to be an extremely useful tool for feature extraction as a prior step to a highly precise real-time signal recognition [23]. Since we face signals which are corrupted by abrupt changes in real situations, it is crucial to relate to the occurrence of an event in time. The wavelet transform utilizes a small wave called the *mother wavelet*, which has finite energy to analyze signals. It allows localization in both the time and frequency domains using translations and dilations of the mother, respectively [23]. The Discrete Wavelet Transform (DWT), which is also known as a Multi-Resolution Analysis (MRA), characterizes a signal by passing it through an analysis filter bank. This bank consists of a low pass and a high pass filter at each decomposition stage. When a signal passes though these filters, it splits into two bands at each stage. The low pass filter, which corresponds to averaging information, extracts coarse information of the signal. For its part, the high pass filter, which operates as a differentiation process, extracts detailed information on the signal. The outputs of the filtering operation, which are decimated by two, are called the approximation (A) and the detailed (D) coefficients, respectively [24, 25]. In this work, DWT-based features are computed as follows:

- Firstly, each observation (x) is decomposed by up to 4 levels (in this case, levels higher than 4 are less informative) using the 10th Daubechies wavelet (db10). Notably, different types of the wavelet were tested, and the best classification result was obtained using db10.
- Secondly, the average power of the 4th approximation and details of the 1st to the 4th level are computed by applying Eq. (10).
- Therefore, we have a 4800 by 10 dataset in which each column represents one of the extracted features. In all, there are 1200 observations per class. Of note, the above features have never been used together in any similar work, and the type of signals in the dataset is different from that seen in other related studies. Thus, this dataset can be considered new in evaluating the performance of ML- based classifiers.

Feature selection: Feature selection is a crucial prior step in the classification to reduce the size of the high dimensional patterns. The most significant benefit of this process is that it decreases the computational complexity of the implementation [26]. It must be borne in mind that reducing the number of representative features may come at the expense of classification accuracy, but the trick in dimensionality reduction is to trade a little precision for simplicity [27]. To this end, we deployed the Principal Component Analysis (PCA) approach for selecting the most informative features.

- **Principal Component Analysis (PCA)**

PCA has been used in a wide range of computer science applications, such as computer vision, pattern recognition and machine learning [26]. Most related studies focus on deploying PCA for feature extraction – such that projecting a dataset from many correlated coordinates onto fewer uncorrelated coordinates called principal components - while preserving information as much as possible [28]. However, in [26], PCA was presented for the first time as an efficient feature selection method with lower computational complexity in comparison to other approaches, such as the genetic algorithm, colony optimization, etc.

Generally, PCA for feature extraction is implemented in three steps, as follows [27]:

Step 1: Mean subtraction: It is an important step to ensure that the first principal component describes the direction of the maximum variance [26].

Step 2: Computation of the covariance matrix: The covariance matrix is a representation of the linear dependency between two values, and is mathematically defined for a zero-mean dataset X as[28]:

$$Cov\,(X) = \sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T = X^T X \tag{11}$$

Step 3: Calculation of eigenvectors and eigenvalues of the covariance matrix: This step is used to determine the principal components of the data. Therefore, the highest eigenvalues are related to the most uncorrelated eigen- vectors which are considered as the principal components. PCA mainly projects as much information as possible toward the first component, while the second component receives as much as possible of the remaining data. Since a feature component is less important for feature extraction, it can also be interpreted that this feature is less informative in the original space [28].

In this paper, we implemented PCA to select more significant features based on Singular Value Decomposition (SVD), which gives us the same kind of information as does the Eigen decomposition. Accordingly, the principal components of a given covariance matrix are computed as follows [28, 29]:

$$SVD\,(XX^T) = U\,\Lambda\,V^T \tag{12}$$

where Λ is a diagonal matrix including the square roots of the eigenvalues of XX^T known as singular values, in descending order [30], and U and V are orthogonal matrices including orthonormal eigenvectors chosen from XX^T(Left singular vectors) and X^TX (Right singular vectors), respectively. According to [26],the effectiveness of the extracted features is defined by computing the contribution to each extracted features (C_i) based on the right singular vectors (V) corresponding to the first m largest eigenvalues as follows:

$$C_i = \sum_{k=1}^{m} |V_{ki}| \tag{13}$$

where V_{ki} denotes the ith element of V_k,$k = 1,2,..,N$, (N is the number of singular vectors) and $|V_{ki}|$ refers to the absolute value of V_{ki}.

Then sorting C_i in descending order, the effectiveness of each feature is calculated as follows:

$$Effectiveness = \frac{C_i}{\sum_{i=1}^{N} C_i} \tag{14}$$

Fig 7 shows the extracted features that are more informative. As can be seen, the most informative features are related to the average power of the wavelet coefficient.

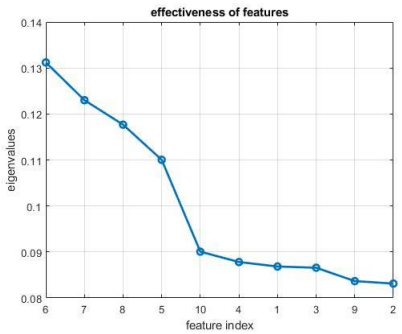

Fig7. *Effectiveness of the extracted features using PCA*

Classification Phase

The data pre-processing step comes right before classification. Here, the dataset is firstly normalized by applying Eq. 15 on each column (i) so that the features are scaled in the 0 to 1 range [31], and then the classes are encoded into a set of numbers as shown in Table 3 since AI-based techniques can only deal with numbers in the computations:

$$NormalizedData = \frac{dataset\ (:,) - Min(i)}{Max(i) - Min(i)} \quad (15)$$

Table3. *Label encoding for each class*

Classes	SoI	SoI+CWI	SoI+MCWI	SoI+CI
Label encoding	1	2	3	4

As mentioned earlier, in this study, an efficient classification technique based on MLP is presented for different types of jamming detection, after which MLP performance is compared to that of SVM [29, 32].

Multi-Layer Perceptron (MLP)

MLP, as a feedforward, multi-layer and nonlinear sigmoid neural network based on the supervised gradient descent algorithm known as Back Propagation (BP). It is used to optimize the error function by updating the coefficients at each layer [33]. The advantage of MLP includes all the benefits provided by a nonlinear structure, in addition to a simple implementation, which make it an appropriate choice in Cognitive Radio (CR) implementation [34].

Network Architecture

The proposed MLP contains three layers, namely, input, hidden and output. The number of input neurons depends on the size of the feature vector, and there is one neuron in the output to represent the corresponding class label. The number of neurons in the hidden layer is variable. As will be shown in the simulation results, the network is trained with different numbers of hidden layer neurons. More importantly, the activation function of the first layer is the logarithmic sigmoid, and the linear function is employed for the output layer. It should be mentioned that the network was trained using different activation functions, but in this case, the most efficient design is done using the aforementioned functions.

The mathematical relationship between the output and input of each layer in the k^{th} iteration is defined as in Eq. (16) below by assuming a dataset $[p_i, y_i]$ where $i = 1, 2, ..., Total\ number\ of\ observations$ and p_i & y_i are the feature vector and its corresponding label, respectively. The output of the first layer before applying the activation function is:

$$n_1(k) = W_1 * p(i,:) + b_1 \quad (16)$$

The first layer's output is obtained by applying the logarithmic sigmoid function:

$$a_1(k) = \frac{1}{1+\exp(n_{1)}} \quad (17)$$

The output of the second layer is:

$$n_2(k) = a_1(k) * W_2 + b_2 \quad (18)$$

The classifier's output is computed by deploying the linear function:

$$a_2(k) = n_2 \quad (19)$$

Moreover, the Mean Square Error (MSE) of the classifier output is calculated as:

$$Error(k) = y_{target} - a_2 \quad (20)$$

$$MSE = E[(Error(k))^2] \quad (21)$$

Learning phase

The foremost to be undertaken before starting the learning procedure involves splitting the dataset into three subsets, namely, training, and testing and validation data. Because the most supervised classifiers are sensitive to data used for training, classification results will also vary depending on the training dataset. Therefore, to ignore the bias introduced into classification results, it is better to apply a random

selection to choose the training data [35]. It should be recalled that the size of the weight coefficient (W) is determined based on the number of neurons in the previous and current layers. For instance, if the number of neurons in the input and hidden layers is 10 and 15, respectively, w1 is considered a 15 by 10 matrix. Furthermore, the coefficients are updated using the Back Propagation (BP) technique as follows:

$$w_i^m\ (k+1) = w_i^m(k) - \alpha \frac{\partial MSE}{\partial w_i^m} \tag{22}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial MSE}{\partial b_i^m} \tag{23}$$

Where j, α are the number of layers to which the coefficients belong and the learning rate, respectively. Choosing a small value for α enhances the classification accuracy while increasing the training time. Furthermore, as the *MSE* is not in a direct relation with coefficients, the chain rule is applied to compute the derivative of MSE to each of the coefficients [33].

As is fully explained by Algorithm 1, for the learning process, we can analyze three gradient descent-based learning techniques by defining the Batch Size (BS) parameter as follows [30]:

Online learning: This is also known as the Stochastic Gradient descent (SGD) approach, which applies the BP and updates the coefficients for each example in the training dataset [36].

Full-batch learning: In this mode, the BP is applied for each sample in the training dataset, but the model is only updated after all training examples have been evaluated. Therefore, this update is done at the end of each training epoch, which is one cycle through the entire training dataset [36].

Mini-batch learning: This algorithm splits the training dataset into small batches that are used to calculate the gradient and update the coefficients (the size of the BS is variable, depending on the application on hand).

In this research, the performance of the proposed techniques is evaluated using the accuracy parameter computed by specifying the number of True Detection ($N_{T\,D}$) and the number of False Detection ($N_{F\,D}$) [8]:

$$Precision = \frac{N_{TD}}{N_{TD}+N_{FD}} \times 100 \tag{24}$$

Algorithm 1: Jamming Classification Pseudo-Code

Input: Generated dataset

Output: Classification precision

Initialization: 1) Shuffle dataset

2) Initialize the coefficients randomly

3) Split the dataset into training (70%) / validation (20%) / testing (10%) subsets

4) Define the batch size *LOOP Process*

1: **for** k = 1 to *Maximum epoch* **do**

2: for kk=0: floor (No. of training data /batchSize)

3: **for** i = 1 to *batch size* **do**

4:Compute the output of each layer (Eq.16-19)

5:Calculate the error (Eq.20)

6: Compute the Mean Square Error (MSE) of the training phase (Eq.21)

7: Apply the Back Propagation technique

8: **end for**

9: Update the coefficients (Eq.22, 23)

10: end for

11: **for** j = *1 to Number of validation data*

do

12: Calculate the output of the classifier using the updated coefficients.

13: Compute the error (Eq.20)

14: Compute the MSE of the validation phase (Eq.21)

15: **end for**

16: **Stopping rule:** if the validation error is increasing, the learning is stopped.

17: **end for**

18: **for** l = 1 to *number of test data* **do**

19: Compute the classifier output

20: Measure the distance between the classifier output and class labels using Euclidean distance.

21: Choose the minimum distance as the corresponding label.

22: **end for**

23: Compute the classification accuracy (Eq. 24).

4. Simulation Results and Analysis

The simulation results are divided into three sections. Firstly, the classification phase of the proposed approaches is analyzed, and then the impact of the PCA on the classification performance is assessed, and finally, the effect of different noise powers on the detection accuracy is demonstrated. It should be noted that SVM was implemented using ML toolbox, and all the simulations were performed in MATLAB (Version R2019b). The computer system had a Core i5-5257U CPU operating at 2.70 GHz, with RAM= 8 GHz. In addition, the dataset used for classification was generated at AWGN power -145 dBm, which is approximately equal to SNR = 9 dB.

Performance Analysis of Classification Phase

In this section, the results of the different designs of MLP and SVM are evaluated for classification. As shown in Table 4, for MLP, the highest classification accuracy is achieved with 30 neurons in the hidden layer and online learning mode (BS =1). HLN refers to Hidden Layer Neurons. The results below represent the average accuracy (in percentage) of six executions.

Table4. *RFI classification results using different designs of MLP*

HLN	BS=1	BS=100	BS=500	BS=1500	BS=3360
10	94.09	90.14	89.92	73.96	51.60
20	99.65	90.13	94.78	86.44	45.83
30	**100**	99.65	96.53	86.44	52.80

Table 5 presents the results of deploying SVM with different structures (one vs. one and one vs. all) and kernels (Polynomial, Gaussian and Linear). As the results show, the most precise performance is obtained using one vs. one with Polynomial kernel.

Table5. *RFI classification results using different structures of SVM*

	Polynomial	Gaussian	Linear
One vs. one	**100**	99.03	98.77
One vs. all	99.99	98.78	87.45

From the classification results, depending on the chosen structure of the classifier, both classifiers could achieve an average accuracy of 100%.

PCA-based feature selection

In this section, the effect of the PCA-based feature selection approach on the classification is analyzed. As can be seen in Fig 7, we evaluated the performance of the classifiers using different NF (Numbers of Features) equal to 2, 5 and 8. As is shown in Table 6, the proposed MLP can perform efficiently using only 5 features, with an accuracy of 97.05 %.

Table6. *Effect of PCA-based feature selection on the proposed MLP using HLN=10*

NF	BS=1	BS=100	BS=500	BS=1500	BS=3360
2	48.95	44.74	22.39	18.21	20.52

5	96.43	**97.05**	94.79	73.26	54.40
8	9.75	96.53	92.36	64.05	50.34

As illustrated in Table 7, SVM achieves an average accuracy of 99.12%, with only five extracted features.

Table7. *Effect of PCA-based feature selection on the proposed SVM using one vs one structure*

NF	Polynomial	Gaussian	Linear
2	68.91	62.49	50.84
5	**99.12**	97.39	90.62
8	98.95	97.39	90.62

According to the results, SVM+PCA slightly outperforms the proposed MLPNN.

Effect of Noise Power on the Detection Accuracy:

In this section, the robustness of the classifiers is evaluated in the presence of the noisy datasets with different SNR. To this end, three new datasets are generated with different powers of noise in the -125 to -135 dBm/Hz range. As discussed earlier, the AWGN for the classification phase is -140 dBm/Hz. STK models AWGN in the range of -120 to -168 dBm/Hz, which represents approximately 5.21 to 12.03 dB (measured by a signal analyzer).

Fig 8 shows the comparison result of the proposed techniques for unknown data prediction.

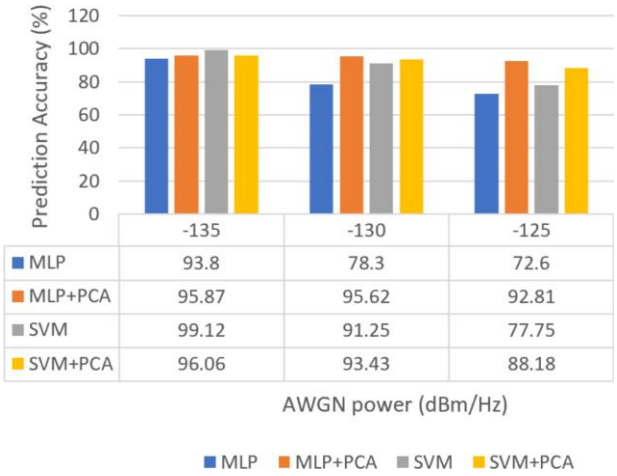

Fig8. *Comparison result of the proposed techniques for unseen data prediction*

5. Conclusion

In this work, an MLP model was developed for RFI classification in a digital video broadcasting scenario based on DVB-S2 standards in a real-time Satcom. Besides analyzing different learning modes (online, full-batch and mini-batch), PCA was deployed as a robust feature selection technique to reduce the computational cost of classification. It can be seen from the results that MLP and SVM could achieve an average accuracy of 97.05% and 99.12%, with only five features. Further, the trained models were used to predict the type of unknown datasets generated at different SNRs. As shown, the PCA-based classifications perform more precisely, and are less sensitive to noise power.

Future suggestions: In this work, we have analyzed the classification process on a fixed JSR only. Therefore, in future work, we can evaluate the classification performance to recognize weak to strong jamming signals. Moreover, to avoid the complex feature extraction phase, we can classify raw data by leveraging Deep Learning (DL)-based classifiers such as Convolutional Neural Networks (CNN).

References

[1] J. A. Jahanshahi, S. A. Ghorashi, and M. Eslami, "A support vector machine based algorithm for jamming attacks detection in cellular networks," in *2011 Wireless Advanced*, 2011: IEEE, pp. 180-184.

[2] O. Puñal, I. Aktaş, C.-J. Schnelke, G. Abidin, K. Wehrle, and J. Gross, "Machine learning-based jamming detection for IEEE 802.11: Design and experimental evaluation," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, 2014: IEEE, pp. 1-10.

[3] K. Grover, A. Lim, and Q. Yang, "Jamming and anti-jamming techniques in wireless networks: a survey," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 17, no. 4, pp. 197-215, 2014.

[4] P. Gao, S. Sun, Z. Zeng, and C. Wang, "GNSS Spoofing Jamming Recognition Based on Machine Learning," in *International Conference On Signal And Information Processing, Networking And Computers*, 2017: Springer, pp. 221-228.

[5] T. Nawaz, D. Campo, M. O. Mughal, L. Marcenaro, and C. S. Regazzoni, "Jammer detection algorithm for wide-band radios using spectral correlation and neural networks," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2017: IEEE, pp. 246-251.

[6] D. Karagiannis and A. Argyriou, "Jamming attack detection in a pair of RF communicating vehicles using unsupervised machine learning," *Vehicular Communications*, vol. 13, pp. 56-63, 2018.

[7] M. Idhammad, K. Afdel, and M. Belouch, "Dos detection method based on artificial neural networks," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 4, pp. 465-471, 2017.

[8] K. Jayakumar, T. Revathi, and S. Karpagam, "Intrusion Detection using Artificial Neural Networks with Best Set of Features," *International Arab Journal of Information Technology (IAJIT)*, vol. 12, 2015.

[9] F. Amato, N. Mazzocca, F. Moscato, and E. Vivenzio, "Multilayer perceptron: An intelligent model for classification and intrusion detection," in *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 2017: IEEE, pp. 686-691.

[10] A. Iversen, N. K. Taylor, K. E. Brown, and J. Karstad, "Classification of Communication Signals and Detection of Unknown Formats Using Artificial Neural Networks," HERIOT-WATT UNIV EDINBURGH (UNITED KINGDOM), 2006.

[11] C. Siaterlis and B. Maglaris, "Detecting DDoS attacks using a multilayer Perceptron classifier," in *Proc. 20th Int. Conf. Artif. Neural Netw., III*, 2004, pp. 118-123.

[12] F. Haddadi, S. Khanchi, M. Shetabi, and V. Derhami, "Intrusion detection and attack classification using feed-forward neural network," in *2010 Second International Conference on Computer and Network Technology*, 2010: IEEE, pp. 262-266.

[13] A. Smith, M. Evans, and J. Downey, "Modulation classification of satellite communication signals using cumulants and neural networks," in *2017 Cognitive Communications for Aerospace Applications Workshop (CCAA)*, 2017: IEEE, pp. 1-8.

[14] E. E. Azzouz and A. K. Nandi, "Modulation recognition using artificial neural networks," in *Automatic Modulation Recognition of Communication Signals*: Springer, 1996, pp. 132-176.

[15] National Instruments. "USRP N210 Kit." http://www.ettus.com/all-products/un210-kit/ (accessed.

[16] Ettus Research. "UHD." https://kb.ettus.com/UHD (accessed.

[17] K. D. S. Solutions. "T400CS Channel Simulator." http://www.rtlogic.com/products/rf-link-monitoring-and-protection-products/t400cs-channel-simulator (accessed Aug 28, 2019).

[18] T.Smyth, "CMPT 468:Frequency Modulation (FM) Synthesis," Simon Fraser University,, School of Computing Science, , 2013,.

[19] O. Mosiane, N. Oozeer, A. Aniyan, and B. A. Bassett, "Radio Frequency Interference Detection using Machine Learning," in *IOP Conference Series: Materials Science and Engineering*, 2017, vol. 198, no. 1: IOP Publishing, p. 012012.

[20] G. Doran, "Characterizing interference in radio astronomy observations through active and unsupervised learning," 2013.

[21] S. Misra *et al.*, "Development of an On-Board Wide-Band Processor for Radio Frequency Interference Detection and Filtering," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 6, pp. 3191-3203, 2018.

[22] A. M. Llenas, J. Riihijarvi, and M. Petrova, "Performance evaluation of machine learning based signal classification using statistical and multiscale entropy features," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017: IEEE, pp. 1-6.

[23] Z. Zhu and A. K. Nandi, *Automatic modulation classification: principles, algorithms and applications*. John Wiley & Sons, 2015.

[24] E. Świercz, "Automatic Classification of LFM Signals for Radar Emitter Recognition Using Wavelet Decomposition and LVQ Classifier," *Acta Physica Polonica, A.*, vol. 119, no. 4, 2011.

[25] A. Gargoom, N. Ertugrul, and W. Soong, "Comparative study of using different mother wavelets on power quality monitoring," *Signal*, vol. 2, p. 2, 2004.

[26] F. Song, Z. Guo, and D. Mei, "Feature selection using principal component analysis," in *2010 international conference on system science, engineering design and manufacturing informatization*, 2010, vol. 1: IEEE, pp. 27-30.

[27] Zakaria Jaadi. "A STEP BY STEP EXPLANATION OF PRINCIPAL COMPONENT ANALYSIS." https://builtin.com/data-science/step-step-explanation-principal-component-analysis (accessed Aug 29, 2019).

[28] ANDRE PERUNICIC. "HOW ARE PRINCIPAL COMPONENT ANALYSIS AND SINGULAR VALUE DECOMPOSITION RELATED?" https://intoli.com/blog/pca-and-svd/ (accessed Aug 29, 2019).

[29] M. E. Wall, A. Rechtsteiner, and L. M. Rocha, "Singular value decomposition and principal component analysis," in *A practical approach to microarray data analysis*: Springer, 2003, pp. 91-109.

[30] Kaggle Inc. "Full batch, mini-batch, and online learning." https://www.kaggle.com/residentmario/full-batch-mini-batch-and-online-learning (accessed Aug 28, 2019).

[31] S. Saitta. "Standardization vs. normalization." http://www.dataminingblog.com/standardization-vs-normalization/ (accessed Aug 29, 2019).

[32] E. Alpaydin, *Introduction to machine learning*. MIT press, 2009.

[33] M. T. Hagan, H. B. Demuth, and M. H. Beale, "Neural network design, PWS Pub," *Co., Boston*, vol. 3632, 1996.

[34] 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.

[35] D. R. Amancio *et al.*, "A systematic comparison of supervised classifiers," *PloS one*, vol. 9, no. 4, p. e94137, 2014.

[36] Jason Brownlee. "A Gentle Introduction to Mini-Batch Gradient Descent and How to Configure Batch Size." https://machinelearningmastery.com/gentle-introduction-mini-batch-gradient-descent-configure-batch-size/ (accessed Aug 28, 2019).

Citation: *Sahar Ujan, et.al., (2020). "A Robust Jamming Signal Classification and Detection Approach Based on Multi-Layer Perceptron Neural Network". International Journal of Research Studies in Computer Science and Engineering (IJRSCSE), 7(1), pp.1-12. http://dx.doi.org/10.20431/2349-4859.0701001*