

Security Threats to Cognitive Radio Signal Classifiers

Timothy R. Newman
Wireless @ Virginia Tech
Virginia Polytechnic and State Institution
Blacksburg, VA 24073
Email: trnewman@vt.edu

T. Charles Clancy
Electrical and Computer Engineering
University of Maryland
College Park, MD 20742
Email: tcc@umd.edu

Abstract—Cognitive approaches to spectrum sensing and differentiation of primary and secondary users can utilize machine learning to improve their performance and robustness in dynamic environments. However when the fundamental concepts of inferencing and classification are applied to signals analysis, some flaws become apparent.

An attacker can manipulate both the feature extraction algorithms and the classifier engines to affect their output. The result is misclassification of the intended signals. In the worst case malicious secondary users can be misclassified as primary users, giving them unfettered access to the spectrum.

This paper describes these attacks for various types of supervised and unsupervised learning, and presents simulation results for a few specific cases. We demonstrate the need for careful engineering when designing these systems to mitigate the efficacy of these types of attacks.

I. INTRODUCTION

Signal classification is a valuable feature for wireless communication systems, specifically for future communication systems where wireless devices may need to avoid a select group of primary user signals. In dynamic spectrum access (DSA) systems, signal classification techniques typically combine signal processing and pattern matching techniques to allow the secondary user to authenticate a primary user signal. This can be done by extracting specific features of the signal through a variety of possible signal processing techniques and then matching these characteristics to a pattern of a known primary user. These features can range from the spectral shape of the primary users signal to high order cyclostationary features of the signal [1].

Using classification techniques such as neural networks or Markov models can help reduce the computational complexity of the system by moving a significant amount of the classification offline. Using these types of classifiers, a limited number of signatures are created offline that correspond to known signals. The classification techniques are then trained offline to recognize the predetermined patterns. Neural networks have been shown to provide a reliable classification technique when used in conjunction with cyclic signal analysis methods [2], [3], [4], [5].

Previous work has not taken into account the possibility of malicious secondary users that may or may not have intimate knowledge of the internal structure of the signal classifier. The ability to force the signal classifier to falsely identify a signal as a primary user could allow a malicious user to manipulate the communication of other wireless systems. We investigate the possibility of a malicious user that is able to construct a signal with the specific goal of causing the signal classifier to provide a false positive result when observing the malicious users signal. For attacks specific to neural networks we look at the possibility of attacking the internal structure of the network, assuming it is known to the attacker, by providing specific signal information in order to force the multilayer network to output a false positive.

The remainder of this paper is organized as follows. Section 2 describes models for signal classification. Section 3 analyzes the impact of malicious elements can have on this signal classification model. Section 4 provides results of simulations and experimentation. Section 5 suggests ways to mitigate the efficacy of these attacks. Section 6 concludes.

II. SIGNAL CLASSIFICATION MODEL

In this paper we consider signal classification algorithms that seek to answer the following question: is the observed complex baseband signal $x(t)$ a primary user or a secondary user? If $x(t)$ is a primary user, then the band it occupies, and possibly even several adjacent bands, is off-limits to use by secondary users. If $x(t)$ is a secondary user, then these bands may be shared by other secondary users.

Later in the paper we evaluate techniques a malicious or greedy agent may use to fool such a signal classifier, convincing it that the malicious user is a primary user, therefore giving it unrivaled spectrum access. This is called a *Primary User Emulation* (PUE) attack [6]. In this section, however, we focus on and develop the signal classification model.

A. Bayesian Approach

Our goal is to distinguish between two classes: primary P and secondary S . Let $\mathcal{C} = \{P, S\}$ be the set of classes. Mathematically the correct class \hat{C} is the one maximizing the following conditional probability:

$$\hat{C} = \arg \max_{C \in \mathcal{C}} P(C|x(t)) \quad (1)$$

Using Bayes Rule, this is equivalent to:

$$\hat{C} = \arg \max_{C \in \mathcal{C}} \frac{P(x(t)|C)P(C)}{P(x(t))} \quad (2)$$

The value of $P(x(t))$ is not part of the maximization, and therefore can be removed, resulting in:

$$\hat{C} = \arg \max_{C \in \mathcal{C}} P(x(t)|C)P(C) \quad (3)$$

The *a priori* probability $P(C)$ is based solely on the ratio of primary to secondary users in the environment. Thus if through prior knowledge we can estimate that ratio r of the total signals in the environment are primary users, we can compute $P(C)$ as

$$P(C) = \begin{cases} r & C = P \\ 1 - r & C = S \end{cases} \quad (4)$$

The *a posteriori* probability $P(x(t)|C)$ is more difficult to compute. In particular $x(t)$ is a vector of large dimension, and the joint probability distribution across an arbitrarily large-dimension $x(t)$ and the two classes P and S is difficult to compute.

B. Feature Extraction

To make computation of a probability more tractable, we reduce the dimensionality of the complex-valued $x(t)$ by projecting it into a feature space through transform $F: \mathbb{C}^\infty \rightarrow \mathbb{R}^N$. We can then estimate as follows:

$$P(x(t)|C) \approx P(F(x(t))|C) \quad (5)$$

Feature selection then becomes very important. Examples of features could include $x(t)$'s occupied bandwidth, variance and kurtosis of its magnitude and phase, baud rate estimate from a cyclostationary analysis, and so on. In general, these features should be computationally light-weight to minimize system complexity.

C. Classifiers

Given the feature vector associated with $x(t)$ the goal is now to determine whether those features are more consistent with class P or S . In Bayesian analysis, these are known as likelihood values, $L_C(F(x(t)))$ for $C \in \mathcal{C}$, which are proportional to the probabilities of interest, i.e.

$$L_C(F(x(t))) \propto P(F(x(t))|C) \approx P(x(t)|C) \quad (6)$$

Classification engines can be used to compute these likelihood values, given appropriate training data. Approaches to classification include constructs such as neural networks, support vector machines [7], K -nearest neighbor, K -means clustering [8], and self-organizing maps [9]. Some of these techniques use *supervised learning*, where a training data would be available providing examples of primary and secondary users. Others use *unsupervised learning*, where no training data is available, and classifiers use clustering and decision-feedback approaches to identify and delineate classes in feature space.

D. Supervised Learning

Examples of supervised learning include the K -nearest neighbor algorithm, support vector machines, and neural networks. In this section we briefly describe each, and the implications on signal classification.

K -nearest neighbor is by far the simplest algorithm. Assume we have annotated training data $(x_i(t), C_i)$, i.e. time-domain signal samples, and an indication of whether that signal is a primary or secondary user. Also assume that we have the same number of primary and secondary users represented in our training data. For each training sample, we compute the associated feature vector $F_i = F(x_i(t))$.

Each feature vector is a point in an N -dimensional space. Given candidate signal $x(t)$ we compute $F(x(t))$, and then find the K points in $\{F_i\}_i$ closest to $F(x(t))$ under some distance metric, such as the L-2 norm. If r of these K neighbors are primary users, then the output likelihoods

$$\begin{aligned} L_P(F(x(t))) &= \frac{r}{K} \\ L_S(F(x(t))) &= 1 - \frac{r}{K} \end{aligned} \quad (7)$$

One drawback of K -nearest neighbors is that it requires a large repository of training data. Support vector machines can reduce this storage requirement if you assume typical feature vectors for primary and secondary users inhabit disjoint areas of the feature space. The goal is to bisect the feature space with a plane (defined by an orthogonal "support" vector), and any signals with feature point lying on side of the plane are deemed primary users, and points on the other side of the plane are deemed secondary users.

The art of support vector machines is finding the plane which minimizes the probability of misclassification, and there is abundant literature on this topic. In general, however, our plane is defined by an N -dimensional vector, and a point on that vector, giving $N + 1$ degrees of freedom in its selection. Algorithms seek to iteratively find values defining the plane that maximize the average distance between the training points and their projection onto the plane.

Yet another approach to classification is neural networks. Here the idea is to mimic the way in which the human brain recognizes things by building a set of interconnected neurons. Each neuron computes a weighted sum of its input and uses that as its output to another neuron. The network has N inputs, representing the N feature dimensions, and 2 outputs, representing $L_C(F(x(t)))$ for $C \in \mathcal{C}$. Training data $\{F_i, C_i\}_i$ is used to select neuron weights subject to output vector $[1, 0]$, representing P , and $[0, 1]$, representing S .

Mathematically, neural networks are function approximators. The classifier is a function that inputs an N -dimensional vector and outputs a 2-dimensional vector. Given known inputs and outputs in training data, it tries to interpolate that function for use in classifying future inputs. The structure of the network dictates the type and degree of functions it can interpolate. In signal classification, most researchers have considered a feed-forward perceptron network with one hidden layer.

E. Unsupervised Learning

Unsupervised learning makes no assumptions about having annotated training data. The classifier figures things out as it progresses. Initial outputs are likely to be more error-prone than later outputs. Examples of unsupervised learning include K -means clustering and self-organizing maps.

These algorithms would be useful for scanning an entire frequency band of interest, locating energy, and then feeding features from that energy into an untrained classifier. With no training phase, we *learn* from live data.

The fundamental assumption is that signals of the same class will have similar values in feature space. By looking at many values of many candidate signals, these feature-space clusters will become more apparent, and can be used to distinguish between the classes. One fundamental issue is determining which class is which, i.e. you know whether a signal falls into ‘‘Class A’’ or ‘‘Class B’’, but not if ‘‘Class A’’ is primary or secondary users. In this paper we assume that some *a priori* knowledge is available to help make that determination.

In K -means clustering, we assume our data is made up of K clusters, in our case $K = 2$, one for primary and one for secondary users. The algorithm selects K random means $\mu_1, \dots, \mu_K \in \mathbb{R}^N$. It then iteratively updates these means to minimize the variance of the points from $\{F_i\}_i$ nearest each hypothesized mean. Eventually the means converge to the cluster centers. Given the means and variances of each cluster, appropriate decision regions can be formulated. The result is similar to a support vector machine, since the decision regions would be bounded by hyper-planes.

Another approach, using the neural network model of self-organizing maps, seeks to first reduce the dimensionality of the problem before performing classification. Typically the N -dimensional feature vectors are projected into a 1- or 2-dimensional space prior to determination of decision boundaries.

For the 2-dimensional case, neurons n_j are points in a lattice (typically square or hexagonal). Each one has an N -dimensional weight vector w_j associated with it. For each training value F_i , neuron n_ℓ is selected such that

$$\ell = \arg \min_j \|F_i - w_j\|_2 \quad (8)$$

Then all weights are updated as

$$w'_j = w_j + \frac{d_{j,\ell}}{\eta} (F_i - w_\ell) \quad \forall j \quad (9)$$

Here $d_{j,\ell} \propto e^{-\|n_j - n_\ell\|_2}$, where n_j is the 2-dimensional coordinate of the neuron, and η is an evolutionary rate parameter exponentially decreasing with time.

In the end, neuron weights will begin to cluster toward feature vector values representing individual classes. By looking at relative weight densities across the lattice, boundaries between neurons representing classes can be drawn, using numerous techniques.

III. THREAT ANALYSIS

There are numerous avenues of attack for a malicious secondary user to be misclassified as a primary user.

One is to manipulate *a priori* information, e.g. the values $\{x_i(t), C_i\}$ used to train the overall system. This, however, can be difficult, depending on the origin of these values. Since class annotation requires expert analysis of the signal, detection of an attack is more likely at this stage.

The second approach is to manipulate *a posteriori* information. To accomplish this, an adversary needs to produce a value $\hat{x}(t)$ such that

$$L_P(F(\hat{x}(t)))P(P) > L_S(F(\hat{x}(t)))P(S) \quad (10)$$

or

$$\frac{L_P(F(\hat{x}(t)))}{L_S(F(\hat{x}(t)))} > \frac{P(S)}{P(P)} \quad (11)$$

There are two ways to accomplish this. The first is to produce a value $\hat{x}(t)$ whose feature-space representation is closer to that of primary users than secondary users. The second is to manipulate the classification engine, which can typically only be accomplished if unsupervised learning is being used.

Note that simply fooling the classifier is not in itself a useful goal. The adversary could simply record and rebroadcast a known primary user to achieve this goal. Given the purpose is to achieve useful communication between malicious users, a simple rebroadcast would

most likely not convey useful information. It would be optimal for the malicious user to determine a specific set of features, from the set of all $x(t)$ satisfying (11), and build its own communication signal with these values as the features.

A. Threats to Feature Extractors

In systems that utilize supervised learning, the main avenue of attack is through the feature extractors. Imagine an ideal classifier that can perfectly infer the conditional probability distributions' distribution function $p_C(f) = P(f|C)$ for $C \in \mathcal{C}$ and compute likelihoods. Each conditional distribution is a joint probability distribution across the feature space \mathbb{R}^N .

Given continuous probability distributions, the probability of any exact feature vector is zero. However, we can consider ϵ -neighborhoods around feature vectors as follows. To be misclassified a value $\hat{x}(t)$ must be found satisfying the following:

$$P(P) \int_{F(\hat{x}(t))-\epsilon}^{F(\hat{x}(t))+\epsilon} p_P(f) df > P(S) \int_{F(\hat{x}(t))-\epsilon}^{F(\hat{x}(t))+\epsilon} p_S(f) df \quad (12)$$

In the limit as $\epsilon \rightarrow 0$, this becomes

$$P(P)p_P(F(\hat{x}(t))) > P(S)p_S(F(\hat{x}(t))) \quad (13)$$

Rather than solve over $x(t)$, we can first compute the feasible set $\hat{F} \subseteq \mathbb{R}^N$ satisfying

$$P(P)p_P(f) > P(S)p_S(f) \quad \forall f \in \hat{F} \quad (14)$$

Assuming an adversary is a software-defined radio and can select between many different waveforms from the set \mathcal{W} , it wishes to select optimal waveform $\hat{w}(t) \in \mathcal{W}$ such that

$$\hat{w}(t) = \arg \max_{w(t) \in \mathcal{W}, F(w(t)) \in \hat{F}} \text{Rate}(w(t)|\text{Channel}) \quad (15)$$

In other words the malicious radio selects a waveform from the feasible set that will be misclassified that maximizes his own communications rate, given the current channel conditions.

The types of waveforms within the feasible set are highly dependent on the individual feature extraction algorithms. For example, if the only feature is a measure of occupied bandwidth, all an adversary need do is mimic the bandwidth of the primary user. If there are features representing spectral or cyclostationary properties of the signal, these too may need to be mimicked in order to succeed.

Also, the feasible set will be dependent on the type of classifier used. No classifier can precisely represent the necessary *a posteriori* probability distributions. Consequently, it makes approximations, interpolates over areas with insufficient training data, and overly reduces the

dimensionality of the problem to make the computational requirements tractable.

Thus individual classifiers may be susceptible to specific attacks. For example, a neural network interpolates the classification function given training data. Areas of the feature space for which no training data was provided could yield strong likelihoods for either P or S , depending on the weights. Section IV-C explores these ideas through simulation.

B. Threats to Classifiers

Systems that use unsupervised learning, or decision-feedback learning, are constantly updating their classifiers as new data arrives. By carefully sequencing transmitted signals, an adversary can manipulate the output of the classifier long-term.

For example, imagine a K -means clustering algorithm searching for $K = 2$ classes. As new data arrives, the cluster means can shift. Figure 1(a) shows a scenario where two features (f_1, f_2) are used to distinguish between two classes. Clusters form for primary users, secondary users, and malicious users. The malicious users are initially identified as secondary users. The malicious users then create waveforms whose feature vectors fill in the space between the primary user cluster and the malicious user cluster. These waveforms need not convey data; we term them *chaff points* since their sole purpose is to manipulate the classifier. After the chaff points are processed by the classifier, the cluster means shift, and the decision boundaries are redrawn. After the change, the malicious users are now misclassified as primary users.

Depending on how long the system maintains training data, this attack could result in a permanent shift in classification behavior. If data is aged off, the attacker may need to re-execute the attack.

For self-organizing maps, the manipulation process is similar. The efficacy depends on the evolutionary rate parameter η . As η converges to zero, the map becomes increasingly difficult to change. Thus it is easier for an attacker to manipulate the map boundaries earlier on in the learning cycle.

The exact usage of η will depend on the implementation. If a spectrum-sensing radio is trying to deal with a highly dynamic environment where primary and secondary users evolve, η cannot converge to zero with time. These types of implementations are exploitable at any time.

IV. EXPERIMENTAL RESULTS

For our simulations and results, we focus on a Dynamic Spectrum Access (DSA) application that uses signal classification techniques to distinguish between primary and secondary users. The FCC has recently

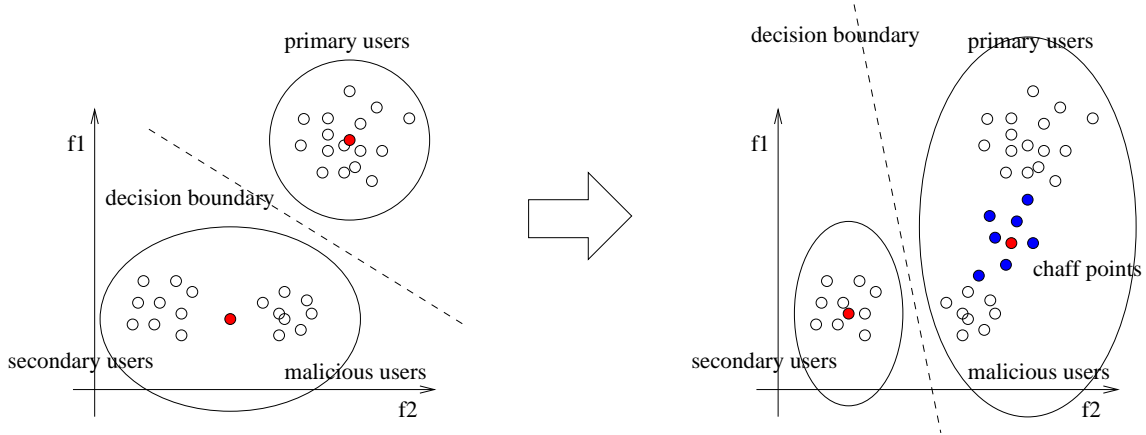


Fig. 1. (a) original cluster means and decision boundary; (b) after attacker adds chaff points, means shift and boundaries are recomputed, causing original attacker waveforms to be reclassified as primary signals

approved a measure that would make the unused space between Digital Television (DTV) channels, or the *white space* spectrum, available for next-generation wireless broadband techniques such as IEEE 802.22 [10]. The IEEE 802.22 is a standard for a wireless regional area network (WRAN) that is able to utilize the white space in a non-interfering manner [11]. In order for the standard to utilize the white space, the spectrum must be sensed and the primary users, the DTV signals and wireless microphone signals, must be accurately classified as such. Secondary users are other 802.22 signals with varying bandwidths and modulation types. Our signal classification experiments focus on the DTV scenario characteristics to simulate a signal classification environment that may be of practical use in the near future.

A. Signal Classification Features

Surveying the many different signal classification models that have been published over the years reveals that features used by classifiers come in many different flavors. On a waveform level, signal features are either temporal-based or spectral-based.

a) Temporal-based Features: Temporal-based features are derived from the complex envelope of the signal as shown in Eq. 16.

$$s(t) = A(t)g(t)\cos(2\pi f(t)t + \vartheta(t)) \quad (16)$$

where $A(t)$ is the amplitude, $g(t)$ is the symbol pulse, $f(t)$ is the frequency, and $\vartheta(t)$ is the phase. From the time domain signal envelope many temporal features can be derived. Typical feature sets that have been used in existing work include the standard deviation of the amplitude and phases, the standard deviation of the change in phase, and the standard deviation of the absolute value in the change in phase [3]. Most of the temporal

features can be generalized as statistical moments with different orders, of the phase and amplitude of the time domain signal. A major drawback of low-order time-domain features is in low noise environments (< 10 dB), the time-domain features are more susceptible to noise and distortion. As we will show this leaves the classifier open to exploitation.

b) Spectral-based Features: Typical spectral statistical features can be extracted from the power spectral density (PSD) of the signal or through analysis of the cyclostationary features of the signal [12]. Using the Fast Fourier Transform (FFT) to transform the time-domain signal into the frequency domain, the Power Spectral Density (PSD) of the signal can be estimated. From the PSD, first-order spectral features such as the bandwidth, received power, and the roll-off factor can be estimated. The first order features are easily emulated with a flexible RF front end, and typically are not used as the only signal classifier features. Cyclostationary signal features are less likely to be emulated and perform much better in low noise environments, but real-time analysis requires much more processing than time-domain features or first-order spectral-based features. This can be a burden on mobile devices that have limited processing resources. The high requirement for processing is primarily due to the calculation of the spectral coherence function [13], [14]. Recent work is attempting to lower this processing hurdle by developing more powerful, smaller, and power efficient processors [15].

Signal classification research and literature about determining the signal features to use have been around for many years. Only recently have they been combined with Software-Defined Radios in order to enable DSA applications. Table I gives a brief summary of many of the commonly used signal classification features and the domain that they exist.

TABLE I
EXAMPLE SET OF SIGNAL CLASSIFICATION FEATURES

Signal Feature	Feature Type
Received Signal Power	Spectral
Baseband Bandwidth	Spectral
Roll-off Factor	Spectral
Alpha Profile	Spectral
Standard Deviation of Amplitude	Temporal
Standard Deviation of Phase	Temporal
Standard Deviation of Envelope Amplitude	Temporal
Standard Deviation of Change in Phase	Temporal
Standard Deviation of Absolute Value of the Change in Phase	Temporal
Nth order Moment/Cumulant of Amplitude	Temporal
Nth order Moment/Cumulant of Phase	Temporal

B. MATLAB DTV Signal Classifier

In a common DTV scenario, where the next-generation network devices have DSA capabilities, the primary users are the DTV signals. Two well defined signal characteristics, as defined by the Advanced Television Systems Committee (ATSC), are the signal bandwidth and modulation. A standard DTV signal uses 6 MHz of bandwidth and local broadcasters are to use 8-level vestigial sideband modulation (8VSB for transmission). The DTV 8VSB modulated signal carries a symbol rate of 10.76 symbols/second and uses Vestigial Sideband Modulation (VSB) that is a derivative of Amplitude Modulation (AM) with a portion of the lower sideband cutoff. A valid assumption for secondary signals is that the IEEE 802.22 standard will be used within the DTV whitespaces as the next-generation WiFi network. Signal features for 802.22 signals have varying bandwidths in order to fit into whitespaces and will make use of the same adaptive modulation and coding techniques as many of the current commercial wireless technologies.

The 8VSB modulation that DTV signals use is not used by 802.22 signals. This allows us to focus on existing signal classification techniques that use modulation classification to distinguish between the signals. For our simulations we use signal classification features that are commonly seen in literature for SDRs. Typically there is a tradeoff between complex, yet robust signal features and simplistic features that are easy to practically implement. Unfortunately, in most cases the simplistic features are used. We demonstrate how using lower order signal features creates significant security issues within the classifier.

In the following MATLAB simulations we use the standard deviation of the time-domain amplitude and the standard deviation of the absolute value of the phase change as the signal features. These features have been used widely throughout modulation classification literature as a low complexity method for classifying both digital and analog modulation schemes. Regarding

the signal construction, we filter the signals using a root-raised cosine filter. The DTV signal is filtered using a roll-off of .11 as specified in the ATSC specifications, while the other digital modulations use a more relaxed roll-off of 0.25. We use 200 samples for each signal classification and simulate the channel using various signal-to-noise ratios.

C. Supervised Learning

Supervised learning is primarily done offline. A set of known inputs and their corresponding outputs or input to the specific classification technique and these are used to train the algorithm to perform in a useful manner. In an extremely static and unchanging environment, with little to no environmental or signal uncertainty, supervised learning produces a classifier that is of very high quality. However, in wireless environments with significant amount of signal uncertainty or moderate to high levels of noise, supervised learning is open to being exploited. This is because the classifier has been defined to operate in a specific manner in an environment that it may not have been trained for. It is in these untrained states that a malicious user can take advantage of the classifier.

In order to illustrate, we simulate simple K-nearest neighbor algorithm. In this simulation the K-NN algorithm is trained with 40 signals. Each signal is input to the classifier and it is explicitly told what type of signal the input represents. Figure 2 shows a representation of the classification landscape of a K-NN MATLAB simulation using data trained in an AWGN channel with an SNR of 20 dB. We illustrate the ease of misclassification problems in Figure 3. Here, the same training set is used, however the unknown signal is received at a much lower SNR of 5 dB. In this case a QPSK signal is misclassified as a DTV signal due to the lower SNR. A malicious user does not need to overwhelm the receiver with noise, but only needs to increase the noise floor enough for the signal classifier to become unreliable.

This technique is extremely trivial, however the affect of a maliciously raised noise floor is often overlooked in signal classification literature. Section IV-D discusses many ways that this malicious attack can be avoided. Many mitigation techniques trade a more complex detection algorithm for a more robust classifier. However, if the more resources for a complex signal classifier are not available, then performing online training during operation may help offset the affects of a malicious user if the attack can not be prevented. However, these unsupervised learning techniques must be designed carefully so as not to open up even more exploitable holes in the system.

[Demo neural network untrained areas]

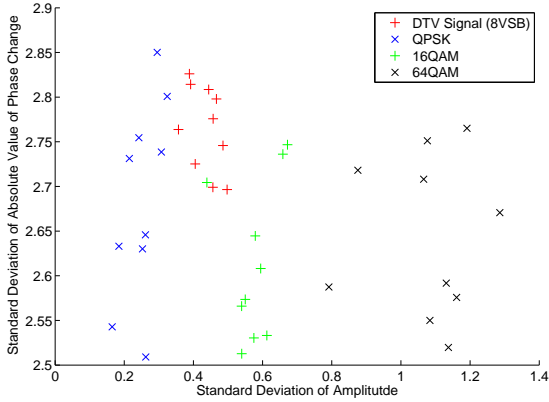


Fig. 2. Example of a trained K-NN modulation classifier using trained data with SNR of 20 dB.

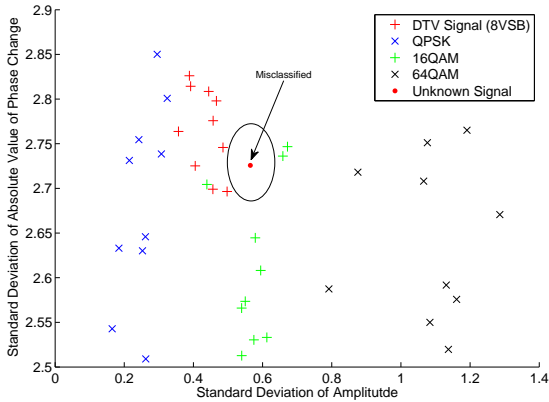


Fig. 3. Example of a trained K-NN modulation classifier using trained data with SNR of 20 dB. The unknown signal received at the signal classifier with a SNR of 5 dB is misclassified

D. Unsupervised Learning

As described earlier in Section II-E, the two primary types of unsupervised learning include K -means clustering and self-organizing maps. Our simulations will focus on the K -means clustering algorithms in order to demonstrate how online learning in general can be exploited over time. Unsupervised learning techniques are not required to hold tight to a specific set of input and output constraints but can adapt the classification to the changing channel conditions. For this reason, unsupervised techniques typically perform better in noisy environments relative to supervised techniques that were not trained for noisy environments. However, a completely unsupervised learning technique is at the complete mercy of the signal feature set. If a malicious user can manipulate the signal features, even in the slightest manner, then the technique may become vulnerable to signal classification attacks.

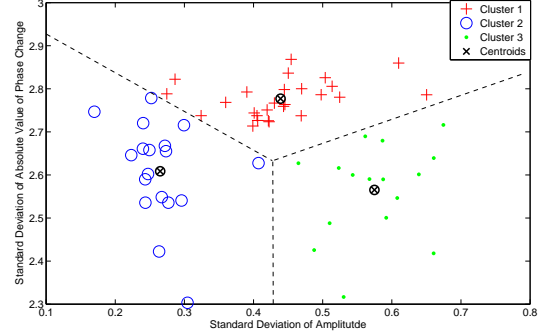


Fig. 4. Example of a K -means clustering algorithm and the decision boundaries after 60 signals have been observed.

Our simulations are again based on the DTV scenario, using the same signal features as before. The K -means clustering algorithm will input the values of the features and begin to form clusters representing the different modulation types. Figure 4 shows the initial clustering of 3 different types of signals, an 8VSB DTV signal (cluster 1 in Figure 4), a QPSK modulated signal (cluster 2 in Figure 4), and a 16QAM modulated signal (cluster 3 in Figure 4). These results were gathered after 60 generated signals, 20 of each modulation type. The dashed lines denote the current decision boundaries, while the large X mark denotes the centroid of the cluster.

At this point, a malicious user may influence the decision boundaries in multiple ways. Depending on the specific modulation required for communication, the malicious user can force the signal classifier to begin to input a large number of signals for a specific modulation. This will bias the classifier and shift the mean of the clusters. For example, if the malicious user desired to communicate using 16QAM modulation, then creating large numbers of signals using 16QAM, relative to the number of DTV signals, would cause the decision boundaries to shift, increasing the number of misclassifications of DTV signals while also increasing the number of 16QAM signals that get classified as DTV signals or primary users. The quantity of extra signals required depends on the speed at which the malicious user desires to shift the decision boundary. It is important to note that the content of the signals used to bias the classifier is irrelevant, they need not carry any data. The only requirement is that the signal features match that of 16QAM signals.

Figure 5 shows the result of increasing the number of observed 16QAM signals from 20 to 100, while keeping the DTV observed signals constant at 20. The dashed lines again show the current decision boundaries, while the solid lines show the previous decision boundaries. Not only have many 16QAM signals been classified

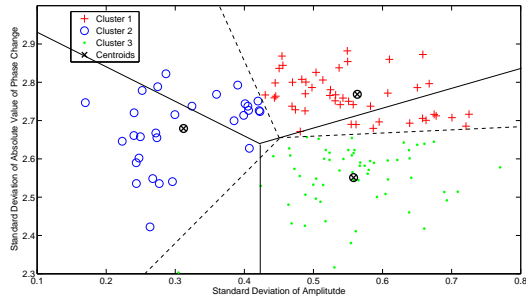


Fig. 5. Example of a K -means clustering algorithm after an influx of 16QAM signals have been observed.

as primary users, but many primary users have now been classified as QPSK signals or secondary users. Once the boundary has been shifted sufficiently, the malicious user can begin to communicate using 16QAM and have the signal classifier believe a primary user is communicating. In a next-generation 802.22 network, this would most likely cause the signal classifier wireless device to evacuate the channel allowing the malicious user full access to the bandwidth.

V. MITIGATING ATTACKS

Our scenarios demonstrate the trivial nature in fooling signal classification systems. One reason that these algorithms are able to be manipulated is that they are designed to classify signals in environments with specific noise, not environments where malicious users are present. With the advent of software-defined and cognitive radios and the future importance signal classification techniques will play in advanced wireless communication, specifically in the TV spectrum, the paradigm of signal classification algorithms need to take into account malicious signals.

The first approach in developing signal classifiers that are robust to malicious users is the of signal feature combinations that are difficult to emulate. While this method seems trivial, using more than simple first-order features makes the emulation of such features much more complex and more prone to error that may be detected by the signal classifier. The drawback is higher-order signal features require more complex signal analysis and are much more computationally complex to implement. This results in the large number of lower complexity, quick and easy classification techniques that have been published. Developing and using more robust signal features applies to both supervised and unsupervised learning approaches.

As was seen in Section IV-D, unsupervised learning approaches can have disastrous outcomes if a malicious user is able to manipulate the learning process. For this reason, a completely unsupervised signal classification

learning approach should never be used. In these cases, the long-term memory of the classifier can be poisoned by malicious users and may possibly never recover completely unless a significant amount of time passes with no malicious input. Using a combination of offline training and online learning has the benefit of static training cases that can not be undone, yet still have the benefit of adapting to an uncertain environment if necessary. However, with offline training cases in place, the online training is essentially “anchored” to a valid set of training cases, thus somewhat diminishing the advantages of unsupervised learning techniques. In approaches such as the K -means technique, a hybrid technique such as this makes it tougher for malicious users to cause the algorithm to drift off into a misclassification state, and will speedup the algorithm recovery when the malicious signal is absent.

VI. CONCLUSION

Thanks to the reconfigurable nature of software-defined radios, signal classification algorithms are again becoming a major topic of research. Specifically in the domain of dynamic spectrum access, signal classification will play a major role in the techniques required to access the spectrum. This new paradigm of using signal classification algorithms to determine access to a channel opens up a new area of security research related to dynamic spectrum access and signal classification. The focus shifts from complex classifiers with relaxed detection time constraints, to signal classification approaches that must meet hard constraints in order to minimize the interference causes to primary users. These initial requirements have dictated the use of simple and quick signal detection and classification methods that can be easily exploited. The reason for this is because we are using traditional signal classification methods not designed for use in a DSA environment.

We have demonstrated using a practical DTV scenario that traditional, well published signal classification approaches can be fooled into misclassification with little effort. Unsupervised and supervised classification techniques are both vulnerable to these malicious attacks and must be developed with malicious signals in mind. Our research has shown that a fully unsupervised approach can result in long-term damage to the classification system, and in turn must always be used in conjunction with a supervised technique. In addition to the hybrid classification approach, the ideal solution is to develop a classification system with robust and unique signal features that are difficult to emulate. Unfortunately, analyzing these features is typically too complex to do in real-time with commercial wireless device hardware.

VII. ACKNOWLEDGEMENTS

This research was funded by the Laboratory for Telecommunications Sciences, US Department of Defense, and the DNI Postdoctoral Fellowship. The opinions expressed in this document represent those of the authors, and should not be considered an official opinion or endorsement by the Department of Defense, Office of the Director of National Intelligence, or US Federal Government.

REFERENCES

- [1] D. Cabric, S. Mishra, and R. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in *IEEE Asilomar Conference on Signals, Systems, and Computers (Asilomar'04)*, pp. 772–776, November 2004.
- [2] A. Fehske, J. Gaeddert, and J. Reed, "A new approach to signal classification using signal correlation and neural networks," in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN'05)*, November 2005.
- [3] B. Le, T. Rondeau, D. Maldonado, and C. Bostian, "Modulation identification using neural networks for cognitive radios," in *SDR Forum Technical Conference (SDR'05)*, November 2005.
- [4] T. O'Shea, T. Clancy, and H. Ebeid, "Practical signal detection and classification in gnu radio," in *SDR Forum Technical Conference (SDR'07)*, November 2007.
- [5] A. Wagstaff and N. Merricks, "A subspace-based method for spectrum sensing," in *SDR Forum Technical Conference (SDR'07)*, November 2007.
- [6] R. Chen, J. Park, and J. Reed, "Defense against primary user emulation attacks in cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 25–37, January 2008.
- [7] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, pp. 559–572, 1901.
- [8] J. Hartigan and M. Wong, "A K-means clustering algorithm," *Applied Statistics*, vol. 28, pp. 100–108, 1979.
- [9] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, pp. 1–6, November 1998.
- [10] FCC, "FCC adopts rules for unlicensed use of television white spaces." http://hraunfoss.fcc.gov/edocs_public/attachmatch/DOC-286566A1.pdf.
- [11] IEEE 802.22, "IEEE 802.22 working group on wireless regional area networks." <http://www.ieee802.org/22/>.
- [12] S. Kay and J. Marple, S.L., "Spectrum analysis a modern perspective," *Proceedings of the IEEE*, vol. 69, pp. 1380–1419, Nov. 1981.
- [13] W. A. Gardner, *Introduction to Random Process with Applications to Signals and Systems*. MacMillan, 1996.
- [14] W. A. Gardner, *Statistical Spectral Analysis A Nonprobabilistic Theory*. Prentice Hall, 1988.
- [15] P. Sutton, K. Nolan, and L. Doyle, "Cyclostationary signatures for rendezvous in ofdm-based dynamic spectrum access networks," *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, pp. 220–231, April 2007.