SIGNAL CLASSIFIERS USING SELF-ORGANIZING MAPS: PERFORMANCE AND ROBUSTNESS

Awais Khawar (University of Maryland, College Park, MD; awais@umd.edu);
T. Charles Clancy (University of Maryland, College Park, MD; tcc@umd.edu)

ABSTRACT

This paper explores the use of self-organizing maps as a mechanism for performing unsupervised learning for signal classification. Approaches using unsupervised learning have a key advantage over traditional approaches that utilize neural networks and support vector machines because they do not require a training phase. We develop signal classifiers using self-organizing maps and explore their robustness. Another concern with using unsupervised learning is the ability for an adversary to shape what is learned. In this paper we explore avenues for attack, and how they affect the performance of the signal classifier. This paper extends previous work on this topic by building a full signal classifier and quantitatively measuring its performance, and introducing two new types of attacks against classification engines.

1 INTRODUCTION

With the growth of wireless communication technologies, the competition for access to electromagnetic spectrum has increased. In order to use electromagnetic spectrum in an efficient manner, a new spectrum sharing technique known as Dynamic Spectrum Access (DSA) was proposed [1]. The field of spectrum sensing has grown significantly over the past five years, with the growth of cognitive radio technology. Spectrum sensing is required for DSA, spectral awareness, interoperability, and many other smart radio applications.

Numerous techniques have been developed to efficiently detect and classify signals in DSA environments. One class of techniques involves computing various simple statistical properties of a detected signal and using a classification engine to determine to which class the signal belongs [2]. Others focus on classification using cyclostationary signal features [3, 4]. Another is based on likelihood approach in which likelihood function of the received signal is computed and a likelihood ratio test is used for detection [5]. For DSA this could be distinguishing between primary and secondary users. For interoperability applications in disaster recovery scenarios, the classifier's goal could be to determine the type of radio communications device being used.

Previous work has shown the usefulness of machine learning to cognitive radio in signal classification. Unsupervised learning is very powerful in a sense that minimal preconfiguration is required and the radios can learn the properties of other devices in their evolving environment. However, an adversary can more easily manipulate the learning process, compromising security [6, 7].

In [8] we investigated the use of unsupervised learning in signal classifiers, and attacks against self organizing maps. We showed how an attacker can manipulate signals and misclassify the adversary user as primary user. This paper extends our work on attacks against signal classifier, exploring two new possible attacks against various types of classification algorithms and presenting simulated misclassification rates under numerous scenarios.

The rest of the paper is organized as follows. Section 2 introduces unsupervised signal classification using self organizing maps and decision boundary algorithms. Section 3 develops new attack against these signal classifiers. Section 4 simulates the attacks and describes their significance. Section 5 proposes mitigation techniques and concludes.

2 CLASSIFICATION USING SELF ORGANIZING MAPS

This section describes the use of self organizing maps in classifying input data. We build decision regions within the derived map to distinguish between different classes of trained data.

2.1 Self Organizing Maps

Self Organizing Maps (SOM) are a type of neural network where individual weights are modified to accommodate input data [9]. SOM are used to explore and visualize properties of data. SOMs can be used to classify signals [8]. This paper applies and analyzes two approaches for clustering SOM data: K-means and hierarchal clustering.

We present a series of signal values $x_n(t)$ to a signal classifier and the goal is to determine whether $x_n(t)$ is a primary user **P** or a secondary user **S**. The correct class is selected by minimizing the probability of error [8].

2.2 Decision Boundary Algorithms

After training a SOM to a series of input data, the output neurons need to be clustered into output classes **P** and **S**. Computing the cluster boundary involves drawing a decision boundary between output neurons, as shown in Figure 1, where the decision boundaries are automatically computed and drawn on the neuron density maps.

In order to decide where to automatically draw the boundary we need some sort of clustering algorithm based on the means of the input neurons in weight space. Here we use two types of clustering algorithms: K-means [10] and hierarchical clustering [11, 12]. We test various hierarchical clustering algorithms and find out that the weighted average distance technique is more robust among hierarchical clustering algorithms in this application (see Section 4.6).

3 THREATS TO SIGNAL CLASSIFIERS

In both supervised and unsupervised machine learning environments, an adversarial user can manipulate the feature extractors and classifier engine to affect their output and thus results in the misclassification of the intended signals [7]. Consider three signal classes: primary users, secondary users, and adversarial users. In addition to transmitting signals contain data, adversarial users wish to transmit signals that will influence the decision boundaries, causing them to be classified as primary users by other secondary users.

Connection attacks involve transmitting manipulated signals to add *chaff points* to the classifier map that are collinear with the means of the primary and secondary user clusters. This confuses the signal classifier and causes adversarial users to be clustered with primary users [8]. In this section we extend previous work and develop two new kinds of attacks: *point cluster* and *random noise*.

In the *point cluster* attack, an attacker adds many chaff points all in one place. The idea is to confuse the signal classifier into clustering all users into a single class, with the second class being the chaff points. This approach is successful, since in many classification algorithms, e.g. K-means, we seek to find the mean position of all the samples in the class. The presence of many chaff points at a particular point draws the mean of adjacent classes closer to itself. In practice if we produce many signals with the same statistical properties we can force the signal classifier to term the chaff cluster a single independent class and all other classes to be a second class. Thus an adversary producing large amount of signals can cause secondary users to believe that the adversarial users are primary users, giving them unrivaled access to the spectrum.

In the *random noise* attack, an attacker adds chaff points randomly all over the map space. The idea is to confuse the signal classifier so that it draws the decision boundaries randomly. The randomness in the decision boundaries is in the sense that some times it classifies adversarial user to be the primary user and sometimes it draws boundaries accurately. Through simulation we show that this type of attack is also strong enough to confuse the signal classifier and thus when effective draws inaccurate decision boundaries.

4 SIMULATION

We use the MATLAB Neural Network Toolbox to show the efficacy of decision boundary movement in self organizing maps. We create a network with 3-dimensional weight space and 2-dimensional map space, containing 100 neurons in grid topology. The weight space spans values $[0, 10]$ in each dimension. Input data samples are taken from two sets of three Gaussian distributions. The means of the two Gaussian distributions are chosen in such a way so that in one set we have adversaries adjacent to secondary users other and in the other all three classes are equidistant.

Adjacent Adversaries:

$$\mu_1 = (3, 3, 3), \quad \mu_2 = (7, 7, 7), \quad \mu_3 = (5, 8, 7)$$

Equilateral Adversaries:

$$\mu_1 = (3, 3, 3), \quad \mu_2 = (4, 7, 5), \quad \mu_3 = (5, 3, 7)$$

where μ_1, μ_2 and μ_3 correspond to the means of primary, secondary and adversarial users respectively. The variance for all classes is 0.25. The network is first trained to input samples, in our case we take training data of 600 samples, and then we run the classification algorithm and determine primary and secondary users through the decision boundary algorithm.

The means and variances in the *Adjacent Adversaries* case are consistent with real signal features, namely the standard deviation of time-domain signal, standard deviation of time-averaged time-domain signal, and the standard deviation of the

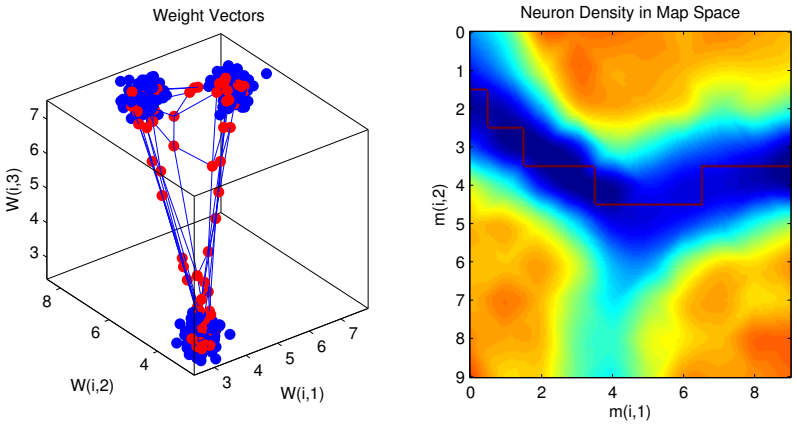

Figure 1: Weight vectors and neuron densities for adjacent adversaries without an attack present

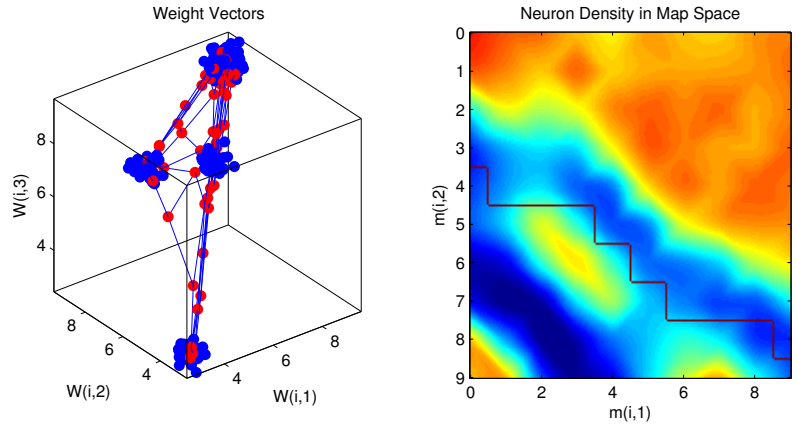

Figure 2: Weight vectors and neuron densities for adjacent adversaries under the point cluster attack

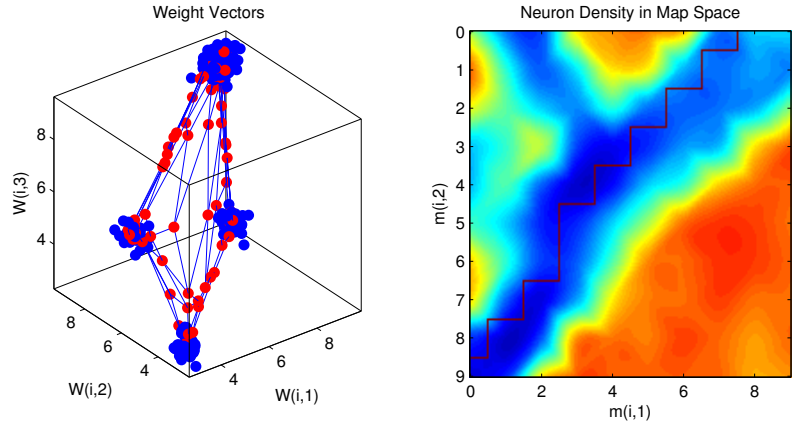

Figure 3: Weight vectors and neuron densities for equilateral adversaries under the point cluster attack

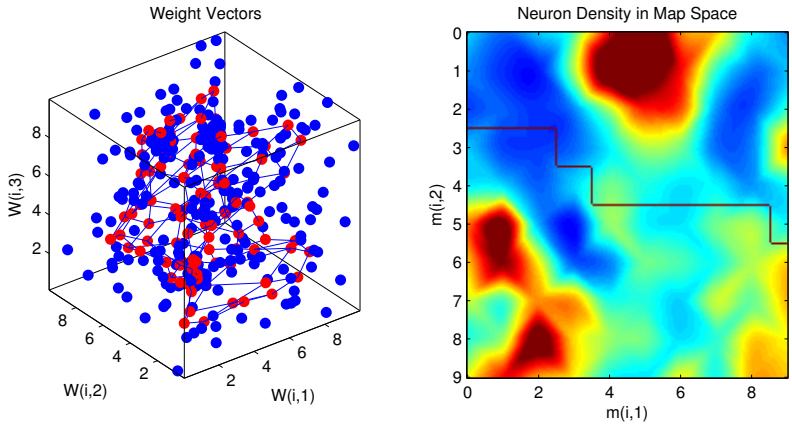

Figure 4: Weight vectors and neuron densities for adjacent adversaries under the random noise attack

derivative of the signal [8]. We also explore the *Equilateral Adversaries* case which is more realistic when adversarial and secondary users are dissimilar in feature-space.

We explain the significance of results in the presence of no attack, connection attack, point cluster attack, and random noise attack. We also demonstrate the relationship between the strength of the attacks on signal classifier and the number of chaff signals.

4.1 No Attack

Figure 1 gives a general overview of how K-means behaves in the case of adjacent adversaries with no attack. Note that when we say "no attack" we do not mean there are no adversarial users in the network; we mean that these adversarial users are not also injecting chaff points to manipulate the decision boundaries.

K-means is able to accurately draw boundaries for the primary and secondary users under no attack. Table 1 shows that the K-means signal classifier is highly successful in drawing accurate decision boundaries under no attack with adjacent adversaries but with equilateral adversaries the performance of K-means degrades and it classifies adversary users to be primary user 35% of the time, as shown in Table 2. This is to be expected, since there are 3 possible partitions of the set selected with uniform probability, so roughly one-third of the time the partition will be selected that causes adversarial users to be misclassified as primary.

Even with no attack present, the misclassification rate is high for K-means under the equilateral adversaries case. The weighted hierarchical clustering algorithm performs much like K-means with adjacent adversaries (see Table 3), but with equilateral adversaries it has a higher misclassification rate as compared to K-means under similar conditions (see Table 4).

4.2 Connection Attack

This section evaluates the performance of the connection attack with K-means and weighted clustering under adjacent and equilateral adversaries. Tables 1 and 3 show that connection attack has no impact on clustering algorithms with adjacent adversaries, however, tables 2 and 4 show that with equilateral adversaries the connection attack has a higher misclassification rate. The connection attack is more powerful if we use weighted clustering, misclassifying adversaries as primary 75% of the time but with K-means the misclassification rate is 36%.

To mitigate attacks, it makes sense to use weighted clustering which is less prone to signal classifier attacks.

4.3 Point Cluster Attack

The point cluster attack in the adjacent adversaries case is shown in Figure 2. Notice the location of the cluster of chaff points. The chaff points are added with mean (9,9,9) relative to adjacent signal classes with means (5,8,7) and (7,7,7). The classification algorithm lumps these three classes into a single class because of their close proximity, whereas the signal class with mean (3,3,3) is classified as the other class. The neuron density map has dense region showing the presence of many chaff points.

Figure 3 shows the point cluster attack in the

equilateral adversaries case. The chaff means are (9,9,9), and the other three signals have means (3,3,3), (4,7,5), and (5,3,7). Notice the chaff signals are far from the other signals in the weight vector map as compared to the adjacent adversaries case. Tables 1 and 3 suggest that the weighted clustering algorithm performs better under the point cluster attack with adjacent adversaries as the error rate it produces is much lower than K-means. Weighted clustering has better decision performance over K-means under this attack. Also, tables 2 and 4 show that the weighted algorithm slightly outperforms the K-means with equilateral adversaries. It is interesting to note that K-means completely misclassifies users, failing to draw decision boundaries accurately whereas the weighted is somewhat successful, though still has a 98% error rate.

4.4 Random Noise Attack

In the random noise attack, chaff points are randomly distributed, uniformly, over the probability space. When the self-organizing map is then trained, the neurons have no specific data clusters onto which to converge. Due to the random nature of these signals the signal classifier finds it difficult to draw accurate decision boundaries, for example, it is difficult for K-means signal classifier to locate the mean of the sample clusters when there is randomly distributed noise all over the space of signals. Consequently it randomly draws the decision boundaries.

Figures 4 and 5 show the weight vectors and the neuron density map when a random noise attack is performed on the adjacent and equilateral adversaries cases, respectively. Note that the chaff signals are randomly distributed over the weight space. After the training phase, they are aligned more close to the three classes depending upon the orientation of their weight vectors. The neuron density map shows the two decision regions. The darker and more dense areas correspond to our signal classes whereas the less dense areas are the result of the random chaff signals grouped arbitrarily. Every time the decision boundary algorithm runs on this attack, it tries to find the best boundaries, taking in account the random chaff signals present around original classes and splits the decision region into primary and secondary user.

Table 1 shows that the random noise attack has no effect on K-means with adjacent adversaries but if we use weighted algorithm then we get poor decision regions as shown by table 3. Table 2 shows that in the case of equilateral adversaries this attack has minimal impact over no attack for K-means clustering, but increases misclassification over no attack in the equilateral adversaries case. This shows that K-means is a good algorithm for drawing decision boundary for primary and secondary users as it is better able to withstand the random noise attack.

4.5 Effect of Chaff Points

It is interesting to note the behavior of signal classifiers under the point cluster attack and random noise attack with adjacent and equilateral adversaries by changing the number of chaff signals. The intensity of attack increases with the increase of chaff points. This makes sense since by increasing the number of chaff points we can force the clustering algorithms to draw inaccurate decision boundaries and thus increasing the probability of misclassification.

In Table 5 we demonstrate that the strength of the attack increases regardless of its type as we increase the number of chaff signals. In Table 5 the numbers 0, 200, 400, and 600 respectively correspond to no chaff points, one-third chaff points, two-third chaff points, and an equal number chaff and data points (i.e. 600 data points, 200 from each class, were used in all scenarios). Observe that the strongest attack comes out to be the point cluster attack, as it has high probability of misclassifying adversary users as primary user as compared to the connection and random noise attack.

4.6 Performance of Signal Classifiers

In an unsupervised learning environment, the signal classifiers are very sensitive to an attack because they update themselves when new data arrives so an adversary can manipulate the output of the classifier in the long run [7]. We used SOM as our signal classifier and K-means and hierarchical clustering algorithms to develop decision boundaries.

For K-means we kept $K = 2$ for our two classes of primary and secondary users. In the hierarchal clustering algorithm we tried classification on the basis of weighted, average, complete, single, and ward algorithms (as defined by the MATLAB `cluster` function). *Weighted* uses the weighted average distance, *average* uses the unweighted average distance, *complete* uses the furthest or largest distance, *single* uses the shortest distance, and *ward* uses the inner squared distance among the weight vectors in the two clusters. Note that we are taking into account only the Euclidean distance and the hierarchical tree structure is monotonic. We evaluated the performance of these algorithms under no attack and found no significant misclassification rates.

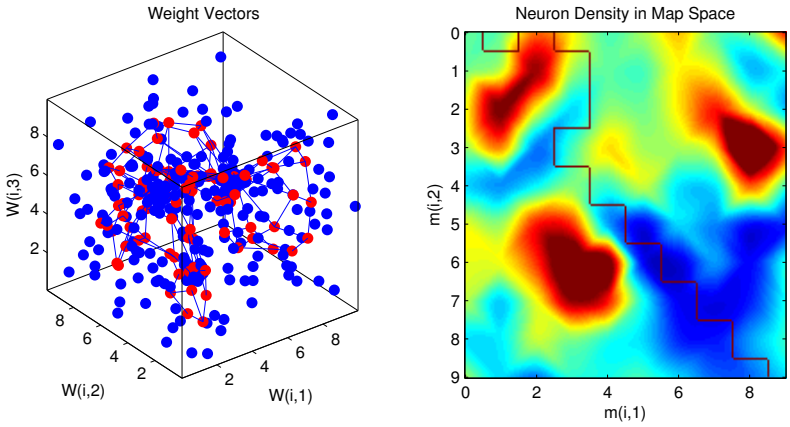

Figure 5: Weight vectors and neuron densities for equilateral adversaries under the random noise attack

Table 6 evaluates the performance with equilateral adversaries, keeping in view the average performance of these algorithms under this situation we find that K-means and *weighted* perform better and are less prone to misclassification on the average as compared to other classification algorithms used. We use these two classification algorithms, K-means and weighted, in order to evaluate their performance under no attack, connection attack, point cluster attack and random noise attack in the previous sections.

4.7 Attack Practicality

Manipulating decision boundaries through transmission of chaff signals can be complex. An adversary needs to generate signals with the proper features in order to inject points into the classifier map, and they need to do it at enough different center frequencies to add enough chaff points to shift decision boundaries. A complete study of the required transmission power to accomplish such an attack has yet to be completed.

As shown in [8], the connection attack can be achieved through transmitting random linear combinations of signals to join their clusters in feature space, assuming linear feature extraction algorithms. The point cluster attack is relatively simple to execute because one only needs to generate a single chaff signal with a sufficiently distant mean to perform the attack. The random noise attack may be the most difficult to achieve, unless a constructive mechanism can be developed to generate signals with arbitrarily random features. Execution of such an attack is highly dependent on knowing the feature extraction algorithms used by other secondary users.

Table 1: Error types and rates for different attack types using K-means clustering with adjacent adversaries

Error Type	None	Connect	Cluster	Noise
Pri→Sec	0	0	0	0
Sec→Pri	0	0	0.35	0
Adv→Pri	0	0	0.38	0

Table 2: Error types and rates for different attack types using K-means clustering with equilateral adversaries

Error Type	None	Connect	Cluster	Noise
Pri→Sec	0	0	0	0
Sec→Pri	0.38	0.33	1	0.36
Adv→Pri	0.35	0.36	1	0.58

Table 3: Error types and rates for different attack types using hierarchical clustering with adjacent adversaries

Error Type	None	Connect	Cluster	Noise
Pri→Sec	0	0	0	0
Sec→Pri	0	0	0.10	0.38
Adv→Pri	0	0	0.11	0.39

Table 4: Error types and rates for different attack types using hierarchical clustering with equilateral adversaries

Error Type	None	Connect	Cluster	Noise
Pri→Sec	0	0	0	0
Sec→Pri	0.30	0.16	0.85	0.40
Adv→Pri	0.42	0.75	0.98	0.55

Table 5: Adv→Pri error rates for different attack densities using hierarchical clustering with equilateral adversaries

Attack Type	0	200	400	600
Point Cluster	0	0.91	0.96	0.98
Connectivity	0.44	0.52	0.58	0.81
Random Noise	0.39	0.46	0.50	0.54

Table 6: Performance of Classification Algorithms under No Attack with equilateral adversaries

Algorithm	Pri→Sec	Sec→Pri	Adv→Pri
K-means	0	0.56	0.30
Weighted	0	0.54	0.44
Average	0	0.84	0.16
Complete	0	0.90	0.10
Single	0	0.68	0.56
Ward	0	0.70	0.30

5 MITIGATION AND CONCLUSION

In this paper our focus was to explore various types of attacks against signal classifiers. We used SOM as our signal classifier and K-means and weighted hierarchical clustering algorithm as our decision boundary algorithms. Simulations demonstrated the effectiveness of connection, point cluster, and random noise attacks against signal classifiers. This demonstrated the fact that unsupervised machine learning environment can be easily manipulated by an adversarial user, since if radios that can adaptively learn from their environment it can also be taught by the environment.

In order to expedite simulation and analysis, the classifiers used cluster means representative of those used in found in previous work [8, 7], without simulating the signals themselves. Nonetheless these results show manipulation of classification algorithms is relatively simple to perform if an adversary can inject chaff points into the classifier.

Further study can use other digital modulation schemes and features well suited for distinguishing them. The use of signal classifiers and clustering algorithms for DSA has opened new frontiers of research in this area. There is a strong need to explore other types of signal classifiers and decision boundary algorithms which are less prone to attacks, more robust and efficient for DSA.

There is no simple choice of decision boundary algorithm to mitigate security threats. While K-means is more robust against the random noise attack, hierarchical clustering is robust against point cluster attacks. A proposed mitigation technique against these attacks is to use known means of primary and secondary users and use them to disambiguate class identities and keep on adding clusters until primary and secondary users are in separate classes. The expected result is that we can reduce the probability of misclassification. The simulation and evaluation of this idea is left for future study.

References

[1] I. Akyildiz, W. Lee, M. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey," *Computer Networks: The International Journal of Computer and Telecommunication Networking*, vol. 50, pp. 2127–2159, 2006.

[2] 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.

[3] 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.

[4] 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.

[5] O. A. Dobre and F. Hameed, "Likelihood-based algorithms for linear digital modulation classification in fading channels," in *IEEE CCECE*, 2006.

[6] T. Clancy and N. Goergen, "Security in cognitive radio networks: Threats and mitigation," in *International Conference on Cognitive Radio Oriented Wireless Networks and Communications (Crowncom'08)*, May 2008.

[7] T. Newman and T. Clancy, "Security threats to cognitive radio signal classifiers," in *Virginia Tech Wireless Personal Communications Symposium*, June 2009.

[8] T. Clancy and A. Khawar, "Security threats to signal classifiers using self-organizing maps," in *International Conference on Cognitive Radio Oriented Wireless Networks and Communications (Crowncom'09)*, June 2009.

[9] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, pp. 1–6, November 1998.

[10] J. Hartigan and M. Wong, "A K-means clustering algorithm," *Applied Statistics*, vol. 28, pp. 100–108, 1979.

[11] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[12] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput. Surv.*, vol. 31, pp. 264–323, September 1999.