Received July 27, 2021, accepted July 31, 2021, date of publication August 5, 2021, date of current version August 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3102859

Machine Learning for Medium Access Control Protocol Recognition in Communications Networks

MARGARET M. ROONEY AND MARK K. HINDERS
Department of Applied Science, William & Mary, Williamsburg, VA 23187, USA

Corresponding author: Margaret M. Rooney (mmrooney@email.wm.edu)

This work was supported in part by the U.S. Naval Research Laboratory, and in part by the Virginia Space Grant Consortium.

ABSTRACT The ability to recognize the medium access control protocol employed by a network can facilitate the incorporation of a cognitive radio into an existing network by elucidating an integral aspect of network behavior. Since the way in which users access the electromagnetic spectrum is one of the most prominent distinctions between reservation based and contention based medium access control protocols, the first part of this work exploits the regular timing of transmissions from networks utilizing reservation based time-division multiple access (TDMA) protocols to differentiate between transmissions governed by TDMA and by contention based carrier sense multiple access (CSMA) protocols. Our approach leverages modular arithmetic to identify periodicity in transmission timings and an unsupervised k-means algorithm to generate distinct TDMA and CSMA clusters. Several supervised machine learning algorithms are explored to build a protocol classifier. We then present a method of distinguishing between transmissions from multi-channel frequency division multiple access (FDMA) based networks and single channel networks. This method uses an automated machine learning clustering algorithm to obtain an estimate of the actual center frequencies of channels utilized by a network. Such information can be used to determine whether the network is employing an FDMA protocol to access the electromagnetic spectrum.

INDEX TERMS Classification, clustering, machine learning, medium access control protocol, wireless communications.

I. INTRODUCTION
Increased use of the electromagnetic spectrum in recent years has led to the development of new technologies with the ability to assess spectrum usage and to adjust transmission parameters intelligently to take advantage of unused frequency bands. For adaptive nodes to utilize vacant spectrum channels efficiently without causing unintended interference to other users, it is beneficial to determine how other networks are accessing a particular channel. If such information is known, a user can, for example, tailor transmitted packets to fit into a particular time slot. The method by which nodes of a communications network share frequency channels is referred to as the medium access control (MAC) protocol.

The majority of the work concerning medium access control protocols for wireless communication networks centers around the development and design of efficient time-slotted, random access, and hybrid protocols. Lai *et al.* [1] designed medium access protocols for cognitive users to access the spectrum opportunistically in the absence of primary users. They focused on determining the probability that a channel is occupied during a given time slot. Yahya and Ben-Othman [2] discussed MAC protocols for wireless sensor networks, including their design and the various advantages and disadvantages associated with each. Este *et al.* [3] investigated the implementation of the support vector machines algorithm to identify traffic emanating from specific applications, while Soysal and Schmidt [4] performed internet traffic classification using flow traces. There exist numerous surveys providing detailed discussion of MAC protocol design for both wireless sensor networks and cognitive radio networks [5]–[10]. A number of authors have investigated the use of machine learning for improved MAC protocols, for primary user detection, and in cognitive radio networks [11]–[16].

Over the last few years, there have been several studies into employing machine learning and deep learning algorithms for

The associate editor coordinating the review of this manuscript and approving it for publication was Jiankang Zhang.

MAC protocol recognition. The publications of Hu *et al.* [17], [18] and Yang *et al.* [19] presented supervised machine learning approaches to MAC identification. In their works, the authors used received power and channel state features combined with a support vector machines model to distinguish between TDMA, CSMA, pure ALOHA, and slotted ALOHA protocols. These methods assume identical transmission power for the nodes in the networks under consideration, and assume that each packet transmission duration in TDMA and slotted ALOHA systems is exactly equal to the length of the time slot. Laghate *et al.* [20] applied a fourth-order cumulant-based modulation type algorithm to classify TDMA, CDMA, and OFDMA channel access methods. Li *et al.* [21] used long short-term memory learning models to again classify these four channel access methods, showing the potential attractions of deep learning for MAC protocol classification scenarios. Zhou *et al.* [22] and Zhang *et al.* [23] both explored the use of CNNs and SVMs to classify MAC protocols: Zhang *et al.* classified TDMA, CSMA, pure ALOHA, and slotted ALOHA protocols by converting raw transmission data into spectrograms for CNN inputs, and Zhou *et al.* created spectrograms from TDMA, slotted ALOHA, and Frequency Hopping signals to use as CNN inputs, with both groups performing manual feature selection to classify signals using SVMs. Although the accuracies of the CNN classifiers reached 99%, the maximum overall accuracy of the various SVMs did not exceed 85%. While these works make simplifying assumptions that all reservation-based packets are of identical length and fit exactly the given time slot, our work does not assume perfect packet reception, identical packet lengths, and transmission timings in order to account for propagation delays and fluctuating power levels. The accuracy levels achieved by our MAC recognition algorithms match or exceed those of comparable works in this area using classical machine learning and deep learning techniques, with our classifiers performing well even when the network traffic is sparse. In addition, none of these publications considers the identification of FDMA vs. single-channel networks.

In this work, we develop a new set of metrics to generate a machine learning feature set for MAC protocol identification with the intention of being applied in blind scenarios to learn more about unknown or uncooperative networks. We then present a two-stage machine learning approach to time-division multiple access (TDMA) and carrier sense multiple access (CSMA) MAC protocol recognition. First, an unsupervised k-means clustering algorithm is employed to partition the dataset into reservation-based protocol and contention-based protocol clusters. These groupings then become labels for the data, and a variety of supervised machine learning algorithms are explored to generate a TDMA/CSMA MAC protocol classifier. Next, we focus on identifying frequency-division multiple access (FDMA) based protocols. The algorithm we have developed uses k-means clustering and an intra-cluster variance based metric to automatically determine the inherent number of clusters in a one-dimensional dataset of transmission center frequencies. The work presented here recognizes the need to analyze potentially uncooperative networks with no prior knowledge of their operating parameters, and the development of our feature sets and algorithms was carried out with this constraint in mind.

The metrics we develop to generate a machine learning feature set for MAC protocol identification result in accuracies of over 90% when used as inputs to SVM, k-NN and Naïve Bayes classifiers. These metrics have been created with the intention of being calculated and evaluated using ML algorithms in blind scenarios to learn more about unknown or uncooperative networks, or to provide a cognitive radio with a better understanding of channel usage in a particular environment. Inferences made from MAC protocol knowledge can allow adaptive nodes to make more efficient use of the available spectrum, or can aid in making informed decisions about effectively deploying jamming signals during electronic attack scenarios.

II. METHOD

We have developed a TDMA/CSMA protocol recognition algorithm that integrates both unsupervised and supervised machine learning techniques. Since the data used for our algorithm development and evaluation were unlabelled in an effort to emulate a blind scenario, we began by employing an unsupervised clustering method to partition the preprocessed data into two groups: reservation based and random access protocols. These clusters became class labels, and a classifier was trained to identify new inputs as TDMA or CSMA governed transmissions. The initial steps of our FDMA-based protocol identification algorithm involve the use of an unsupervised machine learning algorithm to cluster noisy sets of center frequencies.

A. TDMA/CSMA CLASSIFICATION

1) DATASET

The dataset used to develop, evaluate, and refine the algorithm was composed of traces collected from a testbed of universal software radio peripherals (USRPs) and of traces generated by the extendable mobile ad-hoc network emulator (EMANE) software, which allows for real-time modelling of mobile network systems [24]. Each trace contains features such as transmit time (microseconds), packet length (bytes), and center frequency (Hz) for all transmission events during a period of between two and ten minutes. The number of nodes in the networks ranged from four to ten, and nodes were arranged in different topologies over a range of distances producing varying amounts of pathloss to prevent the cultivation of an overly simplified data set by assuming perfect detection of all packets. The congestion levels of the networks were also varied to produce changing traffic loads, such that the average number of transmissions per minute fell between 300 and 25,000 events.

Each trace contains a record of the transmit time of every packet sent in the network, so the transmit time feature for trace T can be written as $T = t_1, t_2, \ldots, t_n$ for a trace containing n transmission events. This feature was used to calculate the differences between consecutive transmissions

$$T_d = \begin{bmatrix} t_2 \\ \vdots \\ t_n \end{bmatrix} - \begin{bmatrix} t_1 \\ \vdots \\ t_{n-1} \end{bmatrix} = \begin{bmatrix} td_1 \\ \vdots \\ td_{n-1} \end{bmatrix} \tag{1}$$

which became the basis of the feature vector generation stage. Since we intend for this approach to be used in blind scenarios, only the inter-packet arrival time is necessary for implementation of the recognition algorithm. In such a situation, information about transmission timings may be obtained by a node through a method such as pulse detection, from which time differences between the commencements of transmission events may be calculated.

In both cognitive radio and military applications, the user is interested in a particular channel or frequency band, and so will focus on detecting transmission events in that area of the spectrum. Although the user is primarily concerned either with accessing or observing a particular frequency band, it might not possess knowledge of the exact set of center frequencies being used by a network due to carrier frequency offset and measurement errors. Therefore, the node would be able to obtain the times of transmission events without knowing the precise center frequencies being used in the monitored band.

2) FEATURE VECTOR GENERATION

In number theory, modular arithmetic is defined by a modulus $N > 1$ and all integers $r \in [0, N-1]$ such that any integer taken modulo N is congruent to some $r \in [0, N-1]$. The congruence class of an integer k modulo N can be determined by writing k as

$$k = m * N + r, \tag{2}$$

where $m, r \in \mathbb{Z}$ and $0 < r < N$. Then,

$$r \equiv k(mod\ N), \tag{3}$$

and so k is in the same congruence class as r.

The goal of this work is to exploit the regular timing of TDMA transmissions to facilitate differentiation between TDMA and CSMA protocols. Ideally, the differences between transmission times of TDMA emissions should be integer multiples of the predetermined time slot length, τ. Thus for any TDMA transmission time difference td_i and modulus τ,

$$td_i\ mod\ \tau \equiv 0. \tag{4}$$

In reality, a variety of factors prevents TDMA transmission time differences from being exact multiples of the time slot duration. To account for effects such as noise and propagation delay, the modulo value of a transmit time difference td_i and the time slot length τ is normalized with respect to τ so that near-integer multiples of τ are treated as integer values. For example, suppose the time slot length $\tau = 8000\mu s$, and a time

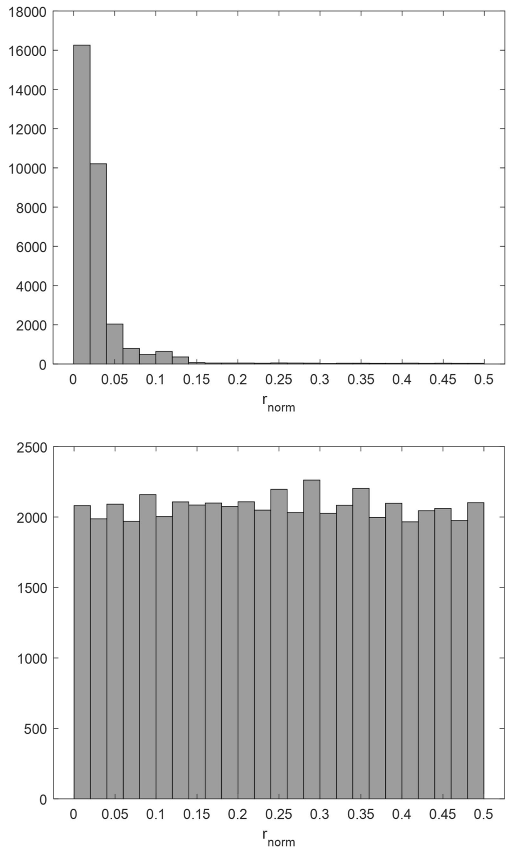

FIGURE 1. **The histograms provide a visual comparison of the differences between the r_{norm} values calculated for TDMA (top) and CSMA (bottom) protocols. Each histogram has a bin size of 0.02 and is plotted over the interval from 0 to 0.5.**

difference between consecutive transmissions is recorded as $56150\mu s$. Then, $56150\ mod\ 8000 \equiv 150$, and so the normalized modulo value is $150/8000 \approx 0.019$. Since 0.019 is close to zero, this particular transmit time difference is recognized as a multiple of the time slot duration. Therefore, for

$$td_i\ mod\ \tau \equiv r, \tag{5}$$

the normalized transmit time difference modulo value, r_{norm}, is calculated as

$$r_{norm} = \begin{cases} \dfrac{r}{\tau} & r < 0.5\tau \\ \dfrac{\tau - r}{\tau} & r \geq 0.5\tau \end{cases} \tag{6}$$

Then for any td, $r_{norm} \in [0, 0.5]$. If the majority of normalized values of transmit time differences modulo τ are approximately zero, this indicates that transmissions frequently occurred at regular intervals, so they likely adhere to a TDMA protocol. If the normalized values of transmit time differences modulo τ are dispersed fairly evenly throughout the interval $[0, 0.5]$, this then indicates that transmissions occurred at random intervals, a characteristic of CSMA protocols. Fig. 1 provides a visualization of the differences in the r_{norm} value distributions between TDMA and CSMA network transmissions.

Since we assume no prior knowledge of the precise operating parameters of a network, only the ability to detect packet timings via methods such as pulse detection, the feature vector generated for each trace contains two calculated elements: the mean of the normalized transmit time difference modulo values, and the variance of the normalized transmit time difference modulo values. Thus the feature vector for trace T_i containing n transmission events is $[\mu_i, \sigma_i^2]$, where the mean of the normalized transmit time difference modulo values is defined as

$$\mu_i = \frac{1}{n-1} \sum_{j=1}^{n-1} r_{norm,j} \tag{7}$$

and the variance of the normalized transmit time difference modulo values is defined as

$$\sigma_i^2 = \frac{1}{n-1} \sum_{j=1}^{n-1} (r_{norm,j} - \mu_i)^2. \tag{8}$$

3) TIME SLOT LENGTH ESTIMATION

TDMA is a medium access control protocol that allows multiple users to transmit on a single channel without collisions. This is accomplished through segmenting time into a series of repeating frames that are further divided into individual time slots. Each time slot is assigned to a network user so that only one user may transmit at any given time. Time slots may be re-assigned to accommodate new users entering the network. In general, due to the slotted structure of a TDMA network, some portion of the transmissions will inevitably occur in consecutive time slots regardless of the amount of network traffic. The percentage of total transmissions that occur in consecutive time slots will be high for congested networks, where vacant time slots are rare, but will be low for uncongested networks where few consecutive time slots are used from one frame to the next.

CSMA is a medium access control protocol in which users access the spectrum randomly and opportunistically. To avoid collisions, users transmit only when the channel seems vacant. If another transmission is in progress, the user waits until the ongoing transmission is complete before using the channel. For CSMA traces, the choice of a potential time slot duration to use as the modulus has little effect on the outcome of the modular arithmetic-heavy feature generation, since the randomness of transmission times will ensure that *mod* values are fairly equally spread throughout the interval [0, 0.5] regardless of the modulus used. Therefore, it was only necessary to focus on establishing a method of estimating the time slot duration which produced globally good results for TDMA traces of varying congestion levels.

Initially, the time slot length for a specific trace was estimated as the minimum transmission time difference. However, this did not consistently result in an accurate estimation of the time slot length since in some TDMA traces, the minimum transmission time difference was much less than the time slot length due to noise. Therefore, the time slot duration used as the modulus in the feature vector generation was estimated individually for each trace by averaging a small percentage of the shortest transmission time durations. Several values between 5% and 10% of the shortest transmission time durations were tested, with 6% repeatedly producing the best approximation of the time slot length for the entire spectrum of network traffic levels. Using such a small percentage of the shortest transmission time durations to estimate the time slot length worked equally well for both highly congested networks and severely uncongested networks, where often less than 20% of transmissions occurred in consecutive time slots. In most cases, the estimated time slot length was within 2% of the actual time slot duration for TDMA traces.

4) MACHINE LEARNING FOR CLUSTERING AND CLASSIFICATION

After generating feature vectors for each of the roughly 160 TDMA and 160 CSMA traces, the unlabelled dataset of over 300 feature vectors was fed into a k-means clustering algorithm that partitioned the dataset into two distinct clusters, each containing approximately half the entire dataset. One cluster, centered near the origin, was composed of traces with low means and low variances. The second cluster was composed mainly of traces with a mean value of about 0.25 and a variance of about 0.0200.

The cluster indices generated by the k-means clustering algorithm were then used to label the entire dataset, which was then split into training and test datasets. Since the k-means clusters accurately divided the dataset into TDMA and CSMA clusters, with the exception of only three data points, these cluster indices were well-suited to being used as supervised data labels. Fig. 2 shows a plot of the two clusters obtained as a result of running the k-means clustering algorithm, while Fig. 3 shows a plot of the true TDMA and CSMA groupings. Various training/test data splits were imposed to assess the performance of each type of classification model. The accuracy of each classifier was calculated as

$$Acc = \frac{TC - MC}{TC} \tag{9}$$

where TC is the total number of cases and MC is the number of misclassified cases.

Each of the classifiers performed well for the varying training/test data splits, with all accuracies exceeding 90%.

Initially, the training/test data split was set as 70/30. Models for k-NN, Naïve Bayes, and SVM classifiers were trained on 227 data points and evaluated on 99 data points. Using a 70/30 training/test data split, each of the five classifiers accurately predicted the class membership of the majority of the test data, with only a handful of misclassified cases. For classification models trained on a 50/50 training/test data split, each subset contained 163 data points. Models generated for a 30/70 training/test data split were trained on 99 data points and tested on 227 data points. Table 1 contains the accuracies and misclassified cases for all classification models and training/test data splits.

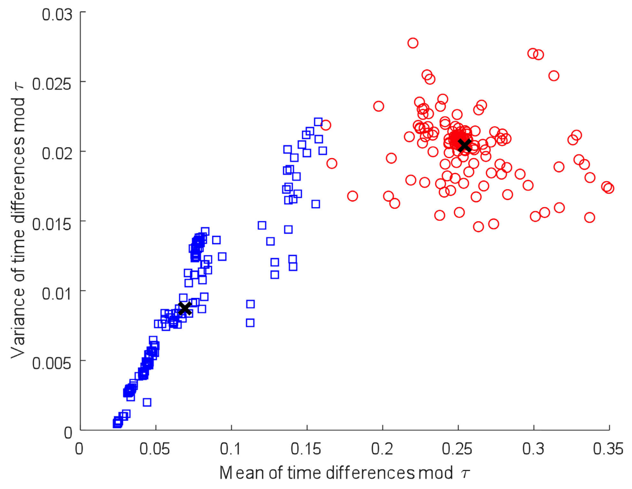

FIGURE 2. Plot of the two clusters obtained as a result of running the k-means clustering algorithm on a dataset composed of 325 data points. The centroid of the cluster near the origin is located at (0.0692, 0.0087), and that of the cluster in the top right of the plot is located at (0.2542, 0.0204).

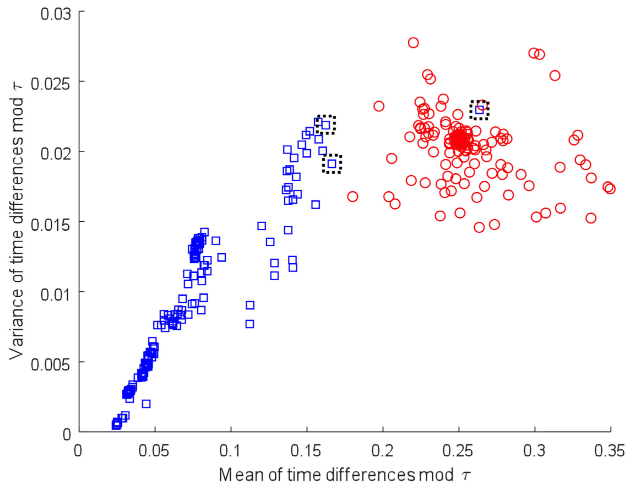

FIGURE 3. Plot of true TDMA and CSMA trace groupings. TDMA traces are represented by blue squares, and CSMA traces are represented by red circles. A comparison between the ground truth clusters and the k-means clusters (Fig. 2) shows that only three of the 325 data points were TDMA traces misclassified as CSMA traces. These misclassified points are outlined by dashed boxes.

B. FDMA-BASED PROTOCOL RECOGNITION

In theory, recognition of an FDMA-based protocol is seemingly straightforward: an FDMA protocol may reasonably be assumed if the number of different center frequencies recorded for a set of network transmissions exceeds one. In practice however, this becomes more complicated. Any collection of packet transmissions will contain noise, so simply tabulating and counting the number of recorded center frequencies used by a network provides a significant misrepresentation of the actual number of channels being used by network nodes to transmit messages. The goal of this algorithm is to run an automated clustering algorithm on a set of center frequencies extracted from a noisy network trace in order to obtain an estimate of the set of center frequencies actually utilized by the network. Such information can then be used to determine whether or not the network is employing an FDMA protocol to access the electromagnetic spectrum.

TABLE 1. Classifier accuracies for various Train/Test splits. All accuracies exceeded 90%.

Train/test split	Classifier	% Acc.	Misclass. cases
70/30	k-NN	97.9	2
	Naïve Bayes	97.9	2
	SVM (linear kernel)	94.9	5
	SVM (cubic kernel)	98.9	1
	SVM (rbf kernel)	98.9	1
50/50	k-NN	98.8	2
	Naïve Bayes	98.8	2
	SVM (linear kernel)	98.2	3
	SVM (cubic kernel)	99.4	1
	SVM (rbf kernel)	98.2	3
30/70	k-NN	98.7	3
	Naïve Bayes	97.8	5
	SVM (linear kernel)	92.5	17
	SVM (cubic kernel)	99.5	1
	SVM (rbf kernel)	97.8	5

1) DATA COLLECTION

The datasets used to create and refine the FDMA recognition algorithm were generated by software defined radio testbeds and the EMANE network simulation software. The software defined radio testbed was composed of USRPs transmitting data generated by video streaming applications. Testbed datasets included transmissions from FDMA, TDMA-FDMA, and CSMA-FDMA networks with between two and eight nodes arranged in various topologies transmitting over the 2.0-2.1 GHz frequency band. The EMANE networks consisted of between two and ten nodes transmitting over the 2.4 - 5.0 GHz frequency band, and employed either a CSMA-FDMA or TDMA-FDMA protocol. Traces of packet transmissions were collected for each network. Included in the fifteen recorded trace features are transmit time in microseconds, packet length in bytes, source and target node identification, bandwidth in MHz, and center frequency in GHz. Therefore, a collection of n transmissions for some network N is recorded in trace T_N as follows:

$$T_N = \begin{bmatrix} TxTime_1 \; packetLength_1 \cdots \; centerFreq_1 \; \cdots \\ \vdots \\ TxTime_n \; packetLength_n \cdots \; centerFreq_n \; \cdots \end{bmatrix}$$

2) CLUSTERING ALGORITHM

A k-means clustering algorithm was run on the set of center frequencies recorded for all transmissions of the network to determine the number of channels. In a dataset exhibiting a relatively low amount of noise, the optimal number of clusters to generate should be equivalent to the number of channels occupied by a network. The algorithm presented here initially partitions the data into two clusters, and in each subsequent implementation of k-means the number of clusters increases by one until the number of channels can be determined confidently. This optimal number of clusters is found by identifying the minimum number of clusters that explains the majority of variance within the dataset.

In every iteration of the k-means algorithm, the variance of each cluster c_i with $i \in [1, k]$, denoted σ_i^2, is calculated as

$$\sigma_i^2 = \frac{1}{m-1} \sum_{j=1}^{m} |x_j - \mu_i| \tag{10}$$

where m is the number of datapoints x in cluster i, and μ is the mean of cluster i. These variances are summed to give the total value of the intra-cluster variances for a set of clusters,

$$\sigma_{tot,k}^2 = \sum_{i=1}^{k} \sigma_i^2. \tag{11}$$

This sum is normalized through division by the variance of the entire set of center frequencies σ_{all}^2 and is then subtracted from one so that the resulting value var_k corresponds to the amount of the total variance within the data that can be explained by segmenting the data into k clusters, such that

$$var_k = 1 - \frac{\sigma_{tot,k}^2}{\sigma_{all}^2}. \tag{12}$$

The var_k values for each k are then recorded in a k-by-2 matrix. The first two columns of Table 2 provide an example of such a matrix.

Plotting the results produces a curve from which the optimal number of clusters can be identified by locating the point at which the slopes of successive segments begin to approximate zero. Fig. 4 shows an example of a curve created using data collected from an 11-node FDMA-based network. The points on the plot are of the form (k, var_k), where k corresponds to the number of clusters and var_k is calculated using (12). The slopes of the line segments are labelled according to the endpoints of the line, so that the slope of the segment connecting points $(k-1, var_{k-1})$ and (k, var_k) is denoted $L_{k-1,k}$. Then for each $i \in [2, k]$, the slope ratio

$$r_i = \frac{L_{i-1,i}}{L_{i,i+1}} \tag{13}$$

is computed. Table 2 provides an example of slope ratios calculated for a plot of the (k, var_k) values computed for an 11-node TDMA-FDMA network.

The largest slope ratio, $r_{max} = max(r_i),\ i \in [2, k]$, nearly always corresponds to the point where subsequent segments have a slope of approximately zero. When r_{max} exceeds a user-defined threshold, the algorithm stops increasing the number of clusters on which to run the k-means algorithm, and the optimal number of channels is identified as the number of clusters corresponding to the maximum slope ratio r_{max}. Oftentimes, since the maximum slope ratio is significantly greater than each of the other slope ratios, the choice

TABLE 2. Example var_k and r_k values calculated for an 11-node TDMA-FDMA network. The optimal number of clusters corresponds to the point at which the slope ratio r_k far exceeds that calculated for other possible numbers of clusters. Here, this point is identified confidently at $k = 11$.

k	var_k	r_k
1	0	—
2	0.508	2.92
3	0.682	1.91
4	0.773	2.22
5	0.815	0.69
6	0.874	2.50
7	0.897	0.95
8	0.922	0.97
9	0.948	1.07
10	0.971	0.95
11	0.996	168.28

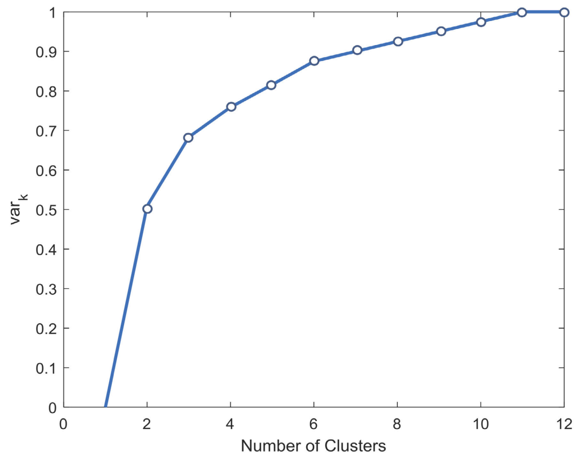

FIGURE 4. Example of a plot of the total variance within the data that can be explained by segmenting the data into k clusters. The optimal number of clusters is chosen by identifying the point on the curve at which slopes of succeeding segments are approximately zero. The curve above was generated using the trace from an 11-node TDMA-FDMA network, where channels transmitted packets on center frequencies between 2.457–2.458 GHz and were separated by 0.1 MHz. As evident from the plot, the curve plateaus starting at $k = 11$.

of the threshold is flexible. If the maximum ratio remains below the threshold after some set number of iterations of the k-means algorithm, another user-defined parameter, the optimal number of clusters is assumed to be one.

After having determined the number of channels the network is likely using to transmit information, the k-means algorithm is run a final time on the collection of center frequency data, with the k-value set equal to the estimated number of channels. This last iteration of the k-means algorithm is then used to create a new list of center frequencies for the network trace by replacing the cluster index of each transmission event with the associated cluster centroid. The new set of center frequencies is appended to the existing trace, and can be used along with the set of transmission source IDs to learn which nodes are communicating on each center frequency.

The final step of the algorithm involves creating a table which contains the identification numbers of nodes that are transmitting on each frequency. It makes a count of the number of nodes transmitting on each channel, calculates the mean of the counts, and outputs the average number of nodes using any channel in the network. If the average number of nodes per channel falls below 1.5 and the number of channels in the network exceeds one, the network protocol is classified as FDMA.

III. DISCUSSION OF RESULTS

The algorithm presented in this paper was trained and tested on data from single-channel CSMA and TDMA networks, as well as TDMA-FDMA and CSMA-FDMA protocols. Our results show that the algorithm we have developed was able to determine accurately whether the network was adhering to a single channel CSMA or TDMA protocol, or whether it was using a hybrid protocol to access the spectrum.

The modular arithmetic method used to generate machine learning features from network traffic was developed under the assumption that only packet transmission times could be reliably detected. Since no additional features, such as number of nodes, node location, and received power are required as inputs to the algorithm, this method provides a way to recognize blindly the MAC protocol of a network. Repeated iterations of the k-means algorithm partitioned the unlabelled dataset into two distinct protocol clusters. The cluster concentrated near the origin contained traces with features characteristic of TDMA, while the second cluster contained traces with higher mean and variance values, characteristic of CSMA. Therefore, each cluster could be labelled confidently as a grouping of TDMA traces or a grouping of CSMA traces. When compared to ground truth, only three out of all 325 traces were incorporated into the wrong cluster.

All classifiers trained on the feature vectors created using the modular arithmetic method described in Section II-A2 achieved an accuracy of over 90%. Only data points midway between the cluster concentrations in the feature space were misclassified.

Although each of the classifiers accurately predicted the correct class of nearly every test data point, two of the three classification algorithms, SVM and k-NN, produce only a hard decision on class membership rather than calculating the probability that a data point belongs to a certain class. Therefore, a distinct advantage of the Naïve Bayes classifier is the option of a probabilistic output, which provides some idea of the certainty with which a class is assigned to each input.

For the few misclassified cases of the Naïve Bayes model for each training/test data split, the predicted probabilities, along with the coordinates of the data points themselves, are listed in Table 3. Plots of the misclassified points together with the set of training data, provided in Figs. 5-7, show that all incorrectly classified traces were situated between the CSMA and TDMA clusters.

TABLE 3. Class membership probabilities, actual class, and feature values for all data points misclassified by the Naïve Bayes classifiers for various Train/Test splits. All incorrectly classified traces were situated between the TDMA and CSMA clusters in the feature space.

Train/Test Split	P(TDMA)	P(CSMA)	Actual Class	μ Feature	σ^2 Feature
30/70	0.11	0.89	TDMA	0.1603	0.0200
	0.24	0.76	TDMA	0.1518	0.0214
	0.48	0.52	TDMA	0.1466	0.0205
	0.11	0.89	TDMA	0.1494	0.0221
	0.32	0.68	TDMA	0.1494	0.0212
50/50	0.69	0.31	CSMA	0.1623	0.0219
	0.72	0.28	CSMA	0.1664	0.0191
70/30	0.41	0.59	TDMA	0.1573	0.0221
	0.39	0.61	TDMA	0.1603	0.0200

For two of the five points misclassified by the Naïve Bayes model trained on 30% of the full dataset, *P(TDMA)* and *P(CSMA)* were roughly equal. In each of the other three cases, the probability leaned in favor of CSMA. However, since all misclassified cases were nearer the concentration of CSMA training data than the concentration of TDMA training data, it was unsurprising that they were all incorrectly classified as CSMA traces.

The Naïve Bayes classifier trained on 50% of the entire dataset misclassified only two of the test data points. Again, the plot of the misclassified points with the training dataset confirms that both points were near the boundary between the two classes in the feature space, and were CSMA traces incorrectly identified as TDMA traces. Since all training data points surrounding the two incorrectly labelled points were TDMA, these misclassifications are not surprising.

The two points misclassified by the Naïve Bayes model trained on 70% of the complete dataset fell on the border between the TDMA and CSMA classes in the feature space, and the output probabilities for each class are roughly equal. Since the data points are midway between the clusters and the CSMA class probability is only marginally higher than the TDMA class probability for each, these misclassifications are not unreasonable.

The FDMA recognition algorithm was generated and refined using data from a variety of different protocol scenarios, including TDMA, CSMA, FDMA, TDMA-FDMA, and CSMA-FDMA networks. All networks utilized between one and eleven channels, with each channel supporting either a single node or the entirety of the network's nodes. An example of the var_k values for each k in a network using 11 channels between 2.457 - 2.458 GHz is given in Table 2. These values represent the amount of variance within data that can be explained by k clusters for an 11-node TDMA-FDMA network. As evidenced by the results in Table 2, over 99% of the variance in the dataset for this particular network can be explained by segmenting the set of center frequencies into 11 clusters. The algorithm accurately estimated the center frequencies of all channels, with differences from actual center frequencies on the order of several hundred kHz. Similar

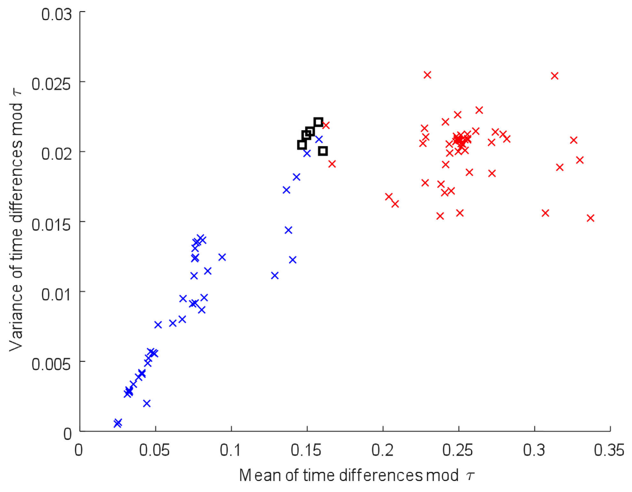

FIGURE 5. Plot of training data and five cases misclassified by the Naïve Bayes model trained on 30% of the entire dataset. The TDMA data points are represented by blue crosses, CSMA data points are represented by red crosses, and the misclassified cases are represented by black squares. Here, each of the misclassified cases was a TDMA data point incorrectly predicted to be a CSMA data point. The model-generated likelihood of the class probabilities for each mislabelled point are recorded in Table 3.

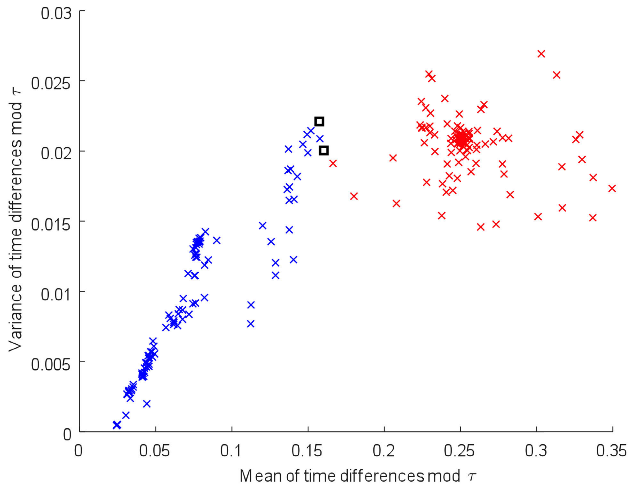

FIGURE 7. Plot of training data and two cases misclassified by the Naïve Bayes model trained on 70% of the entire dataset. The TDMA data points are represented by blue crosses, CSMA data points are represented by red crosses, and the misclassified point is represented by a black square. Here, the single misclassified point was a TDMA data point incorrectly predicted to be a CSMA data point. The model-generated likelihood of the class probability for the mislabelled point is recorded in Table 3.

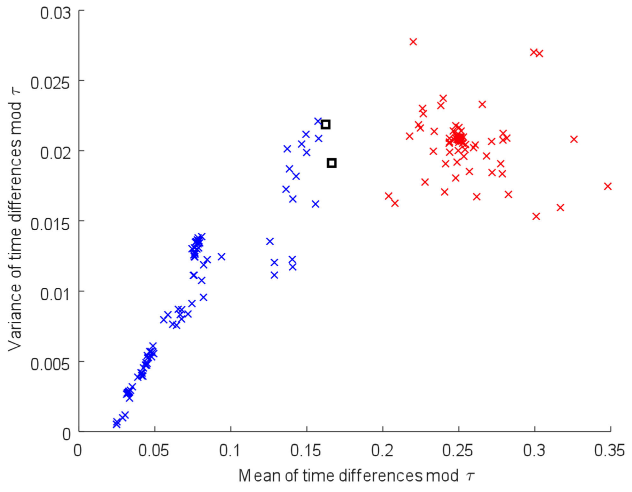

FIGURE 6. Plot of training data and two cases misclassified by the Naïve Bayes model trained on 50% of the entire dataset. The TDMA data points are represented by blue crosses, CSMA data points are represented by red crosses, and the misclassified cases are represented by black squares. Here, each of the misclassified cases was a CSMA data point incorrectly predicted to be a TDMA data point. The model-generated likelihood of the class probabilities for each mislabelled point are recorded in Table 3.

accuracies were obtained for the traces from all FDMA-based networks in the dataset.

Since the main component of the frequency clustering algorithm uses an unsupervised machine learning technique, no training data were required to generate the algorithm. Therefore, all datasets were used to evaluate the accuracy of the channel estimation. The frequency clustering algorithm was tested on 60 sets of network traces, 30 of which were from multi-channel FDMA networks and 30 of which were collected from single channel non-FDMA networks. Across all network traces, the algorithm was tasked with identifying a total of 170 center frequencies from noisy traces. This evaluation was repeated around a dozen times for differing levels of added noise. The reasonable range of noise to introduce to the center frequencies of the simulated data was determined through a Monte Carlo simulation, which indicated that detection error generally does not exceed 0.002% for a 1 MHz signal.

To test for accuracy, the algorithm was run on the entire set of traces eleven times, each time introducing a different amount of detection error into the set of center frequencies. The amount of noise added to the center frequencies from the traces ranged from 0.0 to 0.1% of the channel bandwidth, which corresponded to ± 5 kHz. The accuracy was calculated as the fraction of the 170 center frequencies that were correctly identified. The majority of the misclassifications were single channel networks estimated to be transmitting on two nearly identical center frequencies. Details of the accuracies and misclassifications for all amounts of detection error are contained in Table 4. The algorithm consistently identified over 95% of the center frequencies, with the accuracy improving as the amount of detection error decreased.

As expected, the number of incorrectly estimated center frequencies decreases as the amount of detection error decreases, and all but a few are within the immediate neighborhood of the actual value. The misidentified center frequencies that were not within a small radius of an actual center frequency but were equidistant from two ground truth data points were all from networks where the separation between channels was 1.6 kHz, the smallest channel separation throughout the entire dataset. In these cases, the algorithm generally underestimated about one quarter of the channel frequencies, oftentimes combining data that should have been grouped into two distinct clusters into a single

TABLE 4. Accuracy of the frequency clustering algorithm for varying amounts of detection error. The algorithm consistently identified over 95% of the center frequencies, with the accuracy improving as the amount of detection error decreased.

Detection Error (%)	Accuracy (%)	Misclassified non-FDMA	Misclassified FDMA
0	100.0	0	0
0.01	99.4	1	0
0.02	98.8	2	0
0.03	98.8	2	0
0.04	97.7	0	1
0.05	99.4	1	0
0.06	100.0	0	0
0.07	98.8	2	0
0.08	99.4	1	0
0.09	96.0	1	1
0.10	96.6	1	1

cluster with a centroid near the average of the actual center frequencies. However, although the center frequencies were not all identified correctly for such networks, the algorithm was still able to recognize correctly that the network was FDMA-based.

In order to gauge the efficiency of the algorithm, the run times for traces of various lengths and protocols were recorded and examined. All algorithms were run locally on a PC with an i5-3320M processor running at 2.60 GHz. The total run time for a trace composed of approximately 800 transmission events collected from an 11-channel network averaged around two seconds, as did the run time for the trace from a single-channel network with around 750 recorded transmission events. However, although the run time for the 95,000 event trace log from a three-node FDMA network remained at just over two seconds, the run time for a single-channel network averaged closer to two minutes for a trace with about 50,000 transmission events. Therefore, the algorithm works much more efficiently for multi-channel networks. It seems likely that the method used to identify non-FDMA networks, namely, re-running k-means until the user-defined maximum number of clusters to consider has been reached, causes the significantly longer run time for single-channel networks.

IV. CONCLUSION AND FUTURE WORK

This work presents a MAC protocol recognition algorithm which exploits the regular timing of transmissions from networks utilizing a reservation based TDMA protocol to differentiate between transmissions governed by TDMA and by CSMA protocols. The dataset used to generate feature vectors, train machine learning classifiers, and evaluate the results was collected from a testbed of USRPs and from network modelling software. We used a modular arithmetic-based method to extract features from a record of the transmission times of packets in a network. We then developed a two-stage machine learning approach to MAC protocol recognition, which first involved using an unsupervised k-means clustering algorithm to partition the dataset into reservation-based protocol and contention-based protocol clusters. After clustering, we used these groupings to label the data and then explored a variety of supervised machine learning algorithms to generate a MAC protocol classifier. All classifiers trained on the feature vectors created using the modular arithmetic method achieved an accuracy of over 90%. Our MAC protocol recognition algorithm performed well on both highly congested and uncongested networks.

The channel clustering algorithm was developed for the purpose of recognizing whether or not a network is employing a frequency division multiple access based medium access control protocol. It uses an unsupervised k-means clustering algorithm on one dimensional noisy center frequency data to estimate closely the actual center frequencies which a network is using to transmit packets, and then uses that list of frequencies to determine the number of nodes transmitting on each channel. The data sets used to evaluate the accuracy of the algorithm were collected from both a USRP testbed and from EMANE network simulator software, and included various types of FDMA and non-FDMA based networks. The algorithm accurately identified the center frequencies of the transmission channels, with accuracy consistently exceeding 95% for differing levels of detection error added to the traces, and was successful in distinguishing between FDMA and non-FDMA based networks.

Future work will include extending the MAC recognition algorithms to accommodate a broader range of protocols, particularly those that will be utilized by emerging networks, and finding a more efficient way to recognize networks transmitting on only one center frequency.

ACKNOWLEDGMENT

The authors would like to thank Bryan Nousain, Dr. Crystal Acosta, and Dr. Daniel Salmond for many helpful discussions.

REFERENCES

[1] L. Lai, H. El Gamal, H. Jiang, and H. V. Poor, "Cognitive medium access: Exploration, exploitation, and competition," *IEEE Trans. Mobile Comput.*, vol. 10, no. 2, pp. 239–253, Feb. 2011.

[2] B. Yahya and J. Ben-Othman, "Towards a classification of energy aware MAC protocols for wireless sensor networks," *Wireless Commun. Mobile Comput.*, vol. 9, no. 12, pp. 1572–1607, Dec. 2009.

[3] A. Este, F. Gringoli, and L. Salgarelli, "Support vector machines for TCP traffic classification," *Comput. Netw.*, vol. 53, no. 14, pp. 2476–2490, 2009.

[4] M. Soysal and E. G. Schmidt, "Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison," *Perform. Eval.*, vol. 67, no. 6, pp. 451–467, 2010.

[5] J. Marinho and E. Monteiro, "Cognitive radio: Survey on communication protocols, spectrum decision issues, and future research directions," *Wireless Netw.*, vol. 18, no. 2, pp. 147–164, Feb. 2012.

[6] L. Gavrilovska, D. Denkovski, V. Rakovic, and M. Angjelicinoski, "Medium access control protocols in cognitive radio networks," in *Cognitive Radio and Networking for Heterogeneous Wireless Networks*. Cham, Switzerland: Springer, 2015, pp. 109–149.

[7] K. Kredo and P. Mohapatra, "Medium access control in wireless sensor networks," *Comput. Netw.*, vol. 51, no. 4, pp. 961–994, 2007.

[8] J. Xiang, Y. Zhang, and T. Skeie, "Medium access control protocols in cognitive radio networks," *Wireless Commun. Mobile Comput.*, vol. 10, no. 1, pp. 31–49, Jan. 2010.

[9] Y. Z. Zhao, C. Miao, M. Ma, J. B. Zhang, and C. Leung, "A survey and projection on medium access control protocols for wireless sensor networks," *ACM Comput. Surv.*, vol. 45, no. 1, pp. 1–37, Nov. 2012.

[10] C. Cormio and K. R. Chowdhury, "A survey on MAC protocols for cognitive radio networks," *Ad Hoc Netw.*, vol. 7, no. 7, pp. 1315–1329, Sep. 2009.

[11] M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1996–2018, 4th Quart., 2014.

[12] C. Clancy, J. Hecker, E. Stuntebeck, and T. O'Shea, "Applications of machine learning to cognitive radio networks," *IEEE Wireless Commun.*, vol. 14, no. 4, pp. 47–52, Aug. 2007.

[13] K. M. Thilina, K. W. Choi, N. Saquib, and E. Hossain, "Pattern classification techniques for cooperative spectrum sensing in cognitive radio networks: SVM and W-KNN approaches," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 1260–1265.

[14] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1136–1159, 3rd Quart., 2013.

[15] T. C. Clancy, A. Khawar, and T. R. Newman, "Robust signal classification using unsupervised learning," *IEEE Trans. Wireless Commun.*, vol. 10, no. 4, pp. 1289–1299, Apr. 2011.

[16] Z. Han, R. Zheng, and H. V. Poor, "Repeated auctions with Bayesian nonparametric learning for spectrum access in cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 3, pp. 890–900, Mar. 2011.

[17] S. Hu, Y.-D. Yao, and Z. Yang, "MAC protocol identification approach for implement smart cognitive radio," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 5608–5612.

[18] S. Hu, Y.-D. Yao, and Z. Yang, "MAC protocol identification using support vector machines for cognitive radio networks," *IEEE Wireless Commun.*, vol. 21, no. 1, pp. 52–60, Feb. 2014.

[19] Z. Yang, Y.-D. Yao, S. Chen, H. He, and D. Zheng, "MAC protocol classification in a cognitive radio network," in *Proc. 19th Annu. Wireless Opt. Commun. Conf. (WOCC)*, May 2010, pp. 1–5.

[20] M. Laghate, P. Urriza, and D. Cabric, "Channel access method classification for cognitive radio applications," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 70–73, Feb. 2018.

[21] H. Li, S. Peng, Z. Chen, and X. Qin, "MAC protocol recognition based on LSTM network in cognitive radio," *J. Signal Process.*, vol. 35, no. 5, pp. 837–842, 2019.

[22] Y. Zhou, S. Peng, and Y. Yao, "MAC protocol identification using convolutional neural networks," in *Proc. 29th Wireless Opt. Commun. Conf. (WOCC)*, May 2020, pp. 1–4.

[23] X. Zhang, W. Shen, J. Xu, Z. Liu, and G. Ding, "A MAC protocol identification approach based on convolutional neural network," in *Proc. Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2020, pp. 534–539.

[24] *Extendable Mobile Ad-Hoc Network Emulator (EMANE)*. Accessed: Nov. 2018. [Online]. Available: https://www.nrl.navy.mil/itd/ncs/products/emane

MARGARET M. ROONEY received the B.S. degree in mathematics from St. John's University, Queens, NY, USA, and the M.S. degree in applied science from William & Mary, Williamsburg, VA, USA, where she is currently pursuing the Ph.D. degree in applied science. Her research focuses on two areas: device identification and threat detection in densely populated wireless networks and the characterization of the interactions between high frequency radio waves and the environment. Her research interests include machine learning, cognitive radio, and 5G communications.

MARK K. HINDERS received the B.S., M.S., and Ph.D. degrees in aerospace and mechanical engineering from Boston University. He is currently a Professor of applied science with William & Mary. Before coming to Williamsburg, he was a Senior Scientist at Massachusetts Technological Laboratory, Inc., and also a Research Assistant Professor at Boston University. Before that, he was an Electromagnetics Research Engineer at the USAF Rome Laboratory located at Hanscom AFB, MA, USA. He is the founder of Williamsburg Machine Learning Algorithmics, LLC. He conducts research in medical imaging, intelligent robotics, security screening, remote sensing, and nondestructive evaluation. He and his students study the interaction of acoustic, ultrasonic, elastic, thermal, electromagnetic, and optical waves with materials, tissues, and structures.