Circumventing the Defense against Modulation Classification Attacks

Naureen Hoque
naureen.hoque@mail.rit.edu
Rochester Institute of Technology
Rochester, NY, USA

Hanif Rahbari
hanif.rahbari@rit.edu
Rochester Institute of Technology
Rochester, NY, USA

ABSTRACT

Modulation classification (MC) has a wide range of applications in spectrum sharing, management, and enforcement and can also be used by an adversary to launch traffic analysis or selective jamming. While recent modulation obfuscation techniques show promising results in mitigating MC attacks, in this paper we develop a novel convolution neural network (CNN)-based model to attack those defenses and successfully identify the true modulation scheme. Our extensive simulation and over-the-air experiments using show that our classification technique achieves around $85 - 99\%$ accuracy for SNR levels 0 dB and above. Furthermore, our results demonstrate that the proposed model can effectively differentiate between obfuscated and non-obfuscated symbols, even when a transmitter switches between them as a new defense mechanism, achieving an accuracy of 95%.

CCS CONCEPTS

• **Security and privacy** → **Mobile and wireless security**.

KEYWORDS

Modulation classification, deep learning.

ACM Reference Format:
Naureen Hoque and Hanif Rahbari. 2023. Circumventing the Defense against Modulation Classification Attacks. In *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '23), May 29-June 1, 2023, Guildford, United Kingdom.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3558482.3590197

1 INTRODUCTION

In modern wireless communications, digital modulation is used to map bits into analog symbols at the transmitter, with the number of bits per symbol (i.e., data rate) being adapted based on the transmitter's estimate of the channel condition at the receiver. In cases where the receiver is unable to obtain this information directly from the frame header, modulation classification (MC) is used to identify a received signal's modulation scheme, e.g., by clustering the received (noisy) symbols on the constellation map[1] to find out the alphabet size and how many bits each symbol represents (the modulation order). MC was initially developed for military applications, such as electronic warfare systems that identify enemy signals and their capabilities, including those from radars, improvised explosive devices (IEDs), unmanned aerial vehicles (UAVs), and other sources [1–4]. This information can further trigger countermeasures such as jamming to disrupt enemy communications. MC now has a wide range of civilian applications, too. For example, in spectrum surveillance, a blind monitor (a receiver without prior knowledge of the modulation scheme) in emerging spectrum sharing and dynamic spectrum access systems utilizes MC to determine if radio regulations are being followed by transmitters [4–9]. Link adaptation in vehicular networks, situational awareness for interference detection and mitigation in satellite communications, and the Internet of Things also benefit from efficiently using MC [10–12].

Although MC was designed for legitimate uses, it can also be used by an adversary to perform the actions above for illegitimate uses or to launch traffic analysis attacks (e.g., [13]) as the modulation scheme and coding rate combined can reveal the date rate and the payload size (in bytes). An attacker can also use this information to launch a selective jamming attack, e.g., resulting in an efficient denial of service in which a transmitter's data rate can be degraded from 54 Mbps to 1 Mbps, as shown in [14]. Additionally, an adversary can fingerprint a transmitter [13, 15], breach the privacy of the user by classifying their activities [16], and more.

To prevent one from performing MC, existing modulation obfuscation techniques aim to conceal the true order of modulation used at the transmitter without hampering the quality of communication with its intended receiver [17, 18]. In these techniques, the symbols are always selected from the alphabet (constellation map) of the highest-order modulation scheme supported by the system leveraging coded-modulation techniques; effectively hiding the payload's true modulation order. Such obfuscated symbols are designed to exhibit no distinguishable statistical features [17]. Hence, one cannot employ statistical learning approaches to classify them, as shown in [19]. Their underlying coded-modulation technique is further randomized using a shared secret to disguise the correlation between two successive symbols. However, we investigate whether an attacker can study a *long* sequence of such obfuscated symbols to train a classifier to identify the underlying modulation order. Therefore, in this paper, we aim to answer the following question: *is it possible to classify the underlying modulation order of the obfuscated signals without breaking the secret key?* If one can evade the current obfuscation defenses, then all of the critical applications above is going to be vulnerable to a new generation of MC attacks.

[1] A constellation map is a scatter diagram in the complex plane used to geometrically represent a set of modulated symbols (each defined by its phase and amplitude).

WiSec '23, May 29-June 1, 2023, Guildford, United Kingdom

ACM ISBN 978-1-4503-9859-6/23/05…$15.00
https://doi.org/10.1145/3558482.3590197

To this end, we show that a convolutional neural network (CNN) model can convolve across *long* input sequences and quickly filter out redundant and low-correlated features can subsequently reveal the rate-dependent correlation among the obfuscated symbols. This in turn will disclose the underlying modulation scheme with a significantly higher accuracy than two alternatives known for learning long-term dependencies in sequences. To the best of our knowledge, there is no prior work on attacking signals protected by modulation obfuscation, except one that states but does not analytically quantify that existing classifiers would not perform well if symbols are obfuscated [20]. Our work is the first to empirically break such defenses.

Despite the availability of public datasets of over-the-air received signals for MC, e.g., RadioML [21], they do not contain signals obfuscated before transmission. Hence, they are not applicable to our study. Therefore, we first create a first-of-its-kind dataset of obfuscated symbols to evaluate the performance of various learning algorithms. Using this dataset, we observe that the long short-term memory (LSTM) and transformer network algorithms, known to be capable of learning long-term dependencies in sequences, are unable to identify the underlying pattern efficiently. Based on extensive performance and complexity analysis, our CNN-based model can circumvent the obfuscation techniques with significantly higher accuracy and less complexity than LSTM and transformer models. In contrast to those alternatives, our powerful yet simple model uses *a single one-dimensional convolutional layer* to convolve across input sequences and filter out redundant and low-correlated features, and several special layers that do not contain any neurons themselves, which contribute to its lower computational complexity, faster processing, and higher accuracy.

Contributions—We show that if an adversary is given enough time to collect long sequences of modulation-obfuscated signals (e.g., 2000 symbols) and find an appropriate classifier to train it (a white box model), it will gradually be able to outwit the obfuscation and launch a MC attack. Specifically, our main contributions are:

- We successfully circumvent the current modulation obfuscation defense techniques designed to hide the modulation scheme by developing a novel, powerful, and simple CNN-based classifier. We further show and discuss that among different deep learning (DL) algorithms, including LSTM, our CNN-based model with the lowest complexity is the most suitable classifier to effectively launch the attack.
- We generate a new dataset[2] and evaluate our attack through extensive simulation under varying levels of signal-to-noise ratio (SNR) and complementary USRP[3]-based experiments. The results show that our attack can dodge the existing defenses with 99.7% accuracy. Additionally, our model achieves 85% accuracy under very noisy channels (i.e., 0 dB SNR).
- We further show that if a transmitter chooses to switch between obfuscated and non-obfuscated signals randomly as an improved defense mechanism, our attack can differentiate between these two types of signals and further can identify the underlying modulation order with 95% accuracy.

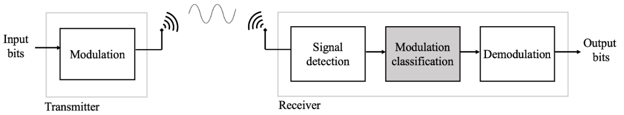


Figure 1: Digital communication when the receiver is blind.

Paper Outline— We explain modulation classification and the existing defense against MC attacks in Section 2 and the threat model in Section 3. Our attack against modulation obfuscation-based defense and the attack performance are in provided Sections 4 and 5, respectively. We discuss possible new defenses in Section 6 before concluding our paper in Section 7.

2 MODULATION OBFUSCATION

In a digital communication system where the receiver is blind (e.g., the headers are encrypted, not decodable, or the communication protocol is unknown), that receiver applies MC to identify the modulation scheme before the demodulation process–see Figure 1. A statistical learning algorithm can easily identify a received signal's modulation class (alphabet) by grouping or clustering the received modulated symbols. As a result, the series of received symbols is a vastly used feature in MC, e.g., using constellation diagrams, distances between symbols, in-phase and quadrature (IQ) values, amplitude and/or phase of symbols [1, 5–7, 20, 22]. Specifically, the existing MC studies use *short* series of non-obfuscated symbols (128 symbols, rarely 512 or more) and achieve high scores using different machine learning (ML) and DL algorithms. For example, Hanna *et al.* [1] used an advanced recurrent neural network and Perenda *et al.* applied spatial transformers to identify each modulation scheme by learning the correlations of the IQ samples [6]. These studies were only focused on classifying the modulated symbols without considering any obfuscated ones. In fact, it is stated in [20] (although not quantified through any analysis) that their models would not perform well under modulation obfuscation.

Existing modulation obfuscation techniques, namely, Conceal and Boost Modulation (CBM) [18] and Friendly CryptoJam (FCJ) [17], *encode* the symbols of any modulation scheme into the highest-order modulation scheme available in the system. These techniques apply Trellis-coded modulation (TCM) to maintain the communication quality when the true modulation order, which depends on channel capacity, is less than the highest one. However, unlike CBM, FCJ maintains and improves transmission quality (e.g., bit error rate) using low-complexity TCMs with two or four states only, without requiring any additional power amplification or latency [17]. In the following, we explain how FCJ-based obfuscation works.

For a given ordered set of supported modulation schemes $\mathcal{M}_i$, where $i = 1, 2, \ldots, M$, the FCJ scheme first divides the points in the constellation map of $\mathcal{M}_M$ into a number of equal-size disjoint subsets (U) with maximum intra-symbol Euclidean distance. The number of sets is determined by the ratio of P_M to P_i, where P_i denotes the numbers of constellation points in $\mathcal{M}_i$. To account for additional demodulation errors resulting from using the denser constellation map of $\mathcal{M}_M$, TCM is used to generate correlated $\mathcal{M}_M$-modulated symbols to represent uncorrelated $\mathcal{M}_i$-modulated ones. TCM uses a finite state machine at the transmitter to map symbols from a given $\mathcal{M}_i$ into either set U_j or U_{j+1} (in the case of FCJ) based on the current state (which itself depends on the past input symbols),

[2] Our implementation code and dataset of (simulated and over-the-air) obfuscated symbols are available at https://github.com/hoquenaureen/attack-mo.
[3] Universal software radio peripheral, a type of software-defined radio designed by NI.

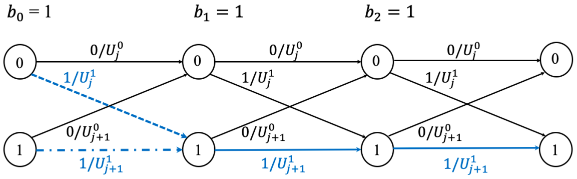


Figure 2: Example of a Trellis diagram of a 2-state TCM under FCJ. b/U_j^b denotes the transmission of a symbol determined by the bit b in U_j. Two possible paths for b_0 depending on the initial state (marked as dashed and dash-dotted lines).

and then moves to the next state based on the current symbol. FCJ varies $j \in \{0, \dots, P_M/P_i - 1\}$ *for each symbol* based on random bits $\mathcal{J}$ generated using a shared secret to prevent an adversary from discerning the size of $\mathcal{M}_i$ (all $\mathcal{M}_M$-modulated symbols must be used, with equal probability) and tracing the dependency among the coded symbol as that would leak $\mathcal{M}_i$ when sets U_j are not secret.

The number of disjoint sets decreases as i approaches M (e.g., to obfuscate binary phase-shift keying (BPSK) and 16-quadrature amplitude modulation (QAM) as 64-QAM, there are 32 and 4 disjoint sets, respectively). In addition, using TCM introduces multiple possible paths for the next symbols, albeit not infinite in number. Figure 2 illustrates a scenario where the first input bit b_0 when $\mathcal{M}_i$ = BPSK has two possible transition paths to its next symbol. Our conjecture is that there are distinct patterns of each obfuscated $\mathcal{M}_i$ depending on the number of possible paths, which itself depends on the number of TCM states and the number of disjoint sets. Although the successive symbols are shown in [17] to be statistically uncorrelated, we posit that observing long sequences may contribute to multiple occurrences of such discerning patterns. Therefore, if an adversary collects sufficiently large dataset of long sequences of obfuscated traffic, they may potentially be able to discern these patterns. We have demonstrated the correctness of this intuition in Section 6, specifically where we observe that the classification of BPSK is comparatively more challenging for a learning algorithm than 16-QAM modulation when both are obfuscated as 64-QAM. Recall that to obfuscate BPSK and 16-QAM as 64-QAM, there are 32 and 4 disjoint sets available, respectively. Hence, a longer sequence may reveal the specific pattern for obfuscated BPSK as opposed to obfuscated 16-QAM (and likewise, 2-state versus 4-state TCM).

3 THREAT MODEL

We consider a rate-adaptive wireless system with a transmitter (Alice) and a receiver (Bob). Alice can either transmit data to Bob using no obfuscation or use a modulation obfuscation technique. TCM can be a 2-state or a 4-state operation (see Section 2). Therefore, each obfuscated-modulation type has two sub-classes, the number of TCM states and the true modulation scheme (order).

There is Oscar, an adversary (or a system defender) who is in communication range of Alice and is entirely passive. He has full knowledge of how the modulation obfuscation algorithm works (i.e., a white-box model). He is then able to generate obfuscated traffic with true data labels for the training phase. His goal is to uncover the true modulation scheme of the obfuscated wireless traffic (i.e., symbols) that Alice and Bob are exchanging.

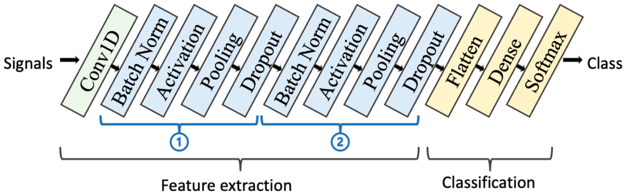


Figure 3: Proposed CNN model. It takes a series of symbols as input and outputs the class (i.e., modulation scheme).

4 ATTACKING MODULATION-OBFUSCATION

In this section, we provide the details of our neural network architecture, dataset, and evaluation metrics for the attack.

4.1 Model Architectures

We consider three DL techniques: CNN, LSTM, and transformer due to their ability to learn patterns from sequential data [23].

LSTM and Transformer Models— These are intuitively potentially suitable since they can learn the relationships in a long sequence of data. Instead of treating each point in a sequence independently, LSTMs can process an entire sequence of data at once using a series of "gates" to retain useful information about previous data in the sequence to help with processing new data points. Therefore, LSTM is specifically good at processing sequences of data (e.g., text, speech, and common time-series data). On the other hand, a transformer-based model applies an evolving set of mathematical approaches (i.e., self-attention) that allows it to "pay attention" to a series of data. Between an LSTM and a transformer-based model, the latter is faster. However, our results (see Section 5) show that neither is able to learn the underlying pattern efficiently. Our proposed LSTM and transformer models are similar to Figure 3, where the first layer is an LSTM or a transformer, respectively. Two separate series of inputs (one for I- and another for Q-values) are fed into this layer. We provide the details of the rest of the common layers below.

Convolution Neural Network (CNN)-based Model— Our model takes the received symbols as input and returns their class of modulation scheme. Then, the demodulator converts them into bits based on the identified modulation class. A CNN layer utilizes its filters to convolve across the input series to find similarities between different locations in the series. The input is fed into a one-dimensional convolution layer since the inputs are vectors of complex numbers (i.e., a wireless signal is represented as a series of complex numbers). This layer includes one-dimensional filters. Multiple filters are taken to slice through, map them one by one over the input vector, and learn the features.

Our proposed architecture consists of two main components: feature extraction and classification (the last three layers in Figure 3). We apply batch normalization, a regularization technique to speed up training and prevent overfitting, to the output of this layer. Next, an activation function is used over the batch normalization's output values. This function's task is to activate neurons based on specific features being present in those values. The pooling layer is applied next to reduce the number of learning parameters (hence, the amount of computation performed). The dropout layer randomly selects hidden incoming/outgoing connections and removes them while training to prevent overfitting. The output of the first dropout layer is the extracted features that we could feed into the classification portion, but we further repeat these four layers to optimize and eliminate redundant correlations from the extracted features (see ② in Figure 3). These special layers do not contain any neurons, but they contribute to reducing computational complexity, faster processing, and more accurate prediction than any model without them. ① quickly extracts the relevant features and ② filters out the redundant and low-correlated ones. In Section 5, we will show that this repetition improves our model's prediction performance without increasing complexity.

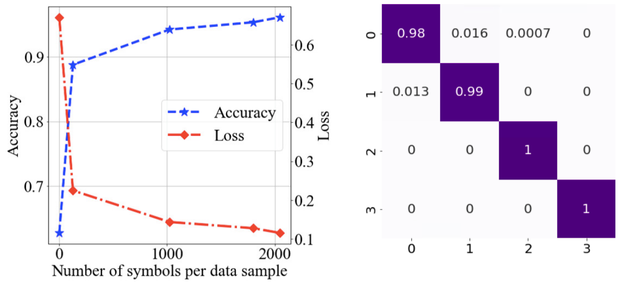
The first layer of the classification part is flattening, which is applied to create a single long feature vector. A dense layer is used to classify output from the flattening layer. Finally, we apply a softmax function (σ) in the output layer to predict a multinomial probability distribution, since our problem is multi-class classification. The output of the dense layer (y_j) is in the interval $[0, 1]$, and their summation will add up to 1, and they can be interpreted as probabilities. The standard softmax function is defined as $\sigma(y_j) = \frac{e^{y_j}}{\sum_{j=1}^{n} e^{y_j}}$.

The complexity of our CNN model is $O(kL)$– very efficient compared to the generic CNN complexity $O(kLd^2)$ as described in [23], where k represents the number of filters, L is the input sequence length, and d is the representation dimension. To reduce the computational complexity of our model, we utilized the series of complex values as one input and only one convolutional layer, unlike existing wireless traffic studies where they use two-dimensional input with multiple convolution layers (e.g., four or seven two-dimensional convolutional layers in [5] and [3], respectively).

4.2 Dataset & Metrics

Our first analysis is under an additive white Gaussian noise (AWGN) channel. We aim to explore the effect of applying a different number of states in TCM to obfuscate symbols; hence, we create modulation-obfuscated traffic according to the scheme outlined in [17]. Our dataset is balanced since each class contains the exact same number of data samples. We collect around 400, 000 signals. A wireless signal is represented as a series of complex numbers (symbols). The SNR under which we collect signals ranges from $0 - 20$ dB. The signals are obfuscated as 64-QAM, either using 2-state or 4-state TCM, but their true modulation orders are BPSK, quadrature phase-shift keying (QPSK), 16-QAM, and 64-QAM. The training and testing ratio for each analysis is 80% to 20%.

We evaluate the model performance based on accuracy, loss, F1-score, and confusion matrices. Accuracy is the ratio between the number of correct predictions and the number of total predictions. We use cross-entropy (or log loss), defined as $L_{ce} = -\sum_{i=1}^{n} T_i \log(p_i)$, by penalizing the probability based on how far it is from the true value and awarding if close. Here, n, T_i, and p_i denote the number of classes, true labels, and predicted probability of observation of class i. We also consider F1-score: *F1-score* $= \frac{t_p}{t_p + 0.5 \times (f_p + f_n)}$, where t_p denotes true positives, f_p denotes false positives, t_n denotes true negatives, and f_n denotes false negatives.



(a) Input length impact. **(b) Confusion matrix.**

Figure 4: (a) Overall performance; (b) Class 0-3 represent obfuscated BPSK, QPSK, 16-QAM, 64-QAM, respectively.

5 PERFORMANCE EVALUATION

We implemented our neural network models using *Keras* as the front-end and *Tensorflow* as the back-end in Python, and used a LabVIEW implementation of FCJ. We conducted the experiments on a Windows 10 Enterprise machine with an Intel Core i7 CPU running at 3.6 GHz and 32 GB of RAM, without using any GPU acceleration.

5.1 Alternative Learning Models

We first study the CNN, LSTM, and transformer-based classifiers' performances. We use a subset of our dataset (6000 wireless signals under AWGN channel, each with 500 symbols). The performance accuracy we achieved using LSTM, transformer, and CNN-based models are 27%, 33%, and 79% accuracy, respectively. We argue that this is because of LSTM's "short-term" memory and the transformer's processing of each signal as a whole, rather than symbol-by-symbol, hence failing to extract the rate-dependent pattern. We also noticed that the CNN is faster in training than the other two models (LSTM, transformer, and CNN models took 510 min 21 sec, 3 min 42 sec, and 2 min 7 sec, respectively). We conclude from this evaluation that a CNN-based model would be the right choice to identify the true class (modulation scheme) of obfuscated signals. However, we further require more data samples and longer series of symbols to improve the classifier performance.

5.2 Performance under Ideal Channels

Next, we performed an extensive study with our CNN model under an ideal channel, including its class-by-class performances.

Model Input & Impact of its Length. Each symbol is a complex number with real and imaginary parts. Unlike a ML method, a CNN-based model can directly deal with complex numbers as input, instead of requiring two separate series of real and imaginary numbers (also reduces the computational complexity). To find the optimal sequence length, we explored different input lengths ranging from 1 to 2048 symbols. In practice, input series may exceed the maximum length of 2048 symbols, in which case longer traces can be truncated and shorter ones can be padded to form fixed-length inputs. Our model achieves 99.7% accuracy and F1-score when each wireless signal has 2000 or more symbols, as shown in Figure 4(a). The class-by-class performance is visualized using a confusion matrix in Figure 4(b).

Table 1: Search range and final value of the parameters to optimize the performance of the CNN model.

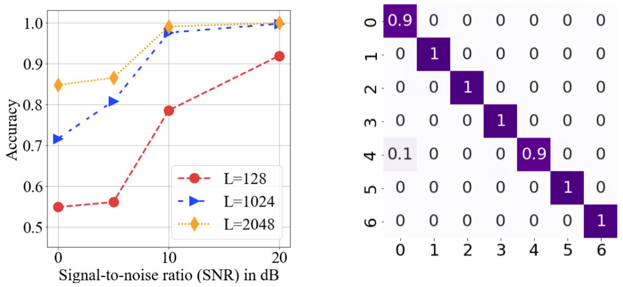
Hyperparameters	Search Range	Best Value
Optimizer	SGD, Adam, Adamax	SGD
Learning Rate	[1, 0.1, 0.01, 0.001]	0.01
Momentum	[0.1, 0.2, 0.4, 0.6, 0.8]	0.6
Training epochs	[1, ..., 100]	5
Batch size	[10, 20, 64, 128, 256, 512]	10
First activation	LReLU, ELU	ELU
Number of filters	[8, 16, 32, 64]	32
Dropout rate	[0, 0.1, 0.3, 0.5, 0.8]	0

Model Training & Overfitting Concern. Overfitting occurs when a model only works on its training data, and cannot be generalized. DL models, like other learning models, are susceptible to overfitting. To counter this, we use 10% of the training data for validating training performance. The three parts of our dataset (train, validation, and test) are mutually exclusive. We find that the difference between the training and validation cross-entropy loss is less than 0.0081, indicating that overfitting is not occurring.

Performance Optimization. We tuned the parameters of our model to their best values to optimize classification performance by performing an exhaustive search through the ranges shown in Table 1. We explored several optimization functions, such as stochastic gradient descent (SGD), adaptive moment estimation (Adam), Adamax with different learning rates and momentum as reported in Table 1. Note that we do not consider standard activation functions such as sigmoid and rectified linear unit (ReLU) as they do not activate on negative input values (a symbol can be negative). Therefore, we included leaky ReLU and exponential linear unit (ELU) in the activation function search space, as they can handle negative values.

Results & Computation Complexity. Figure 4(a) shows that the classifier can successfully identify if the received traffic trace is obfuscated or not with more than 98% accuracy and F1-score when the input series length exceeds 1000 symbols. If we do not include ② in the model, then the accuracy and F1-score drop to 95% with more than 2000 symbols. In contrast to previous work [19], which achieved only random success, our model accurately identifies the true modulation scheme, as it can go beyond statistical information. We also examined the training time and observed that our machine took a total of 139 min and 5 sec for extensive candidate search for best parameter selection– a *one-time* cost without ② in our model (Figure 3). Including this repetition, it took 79 min and 33 sec for the same exhaustive search. Training with the best parameters (Table 1) took only 57 sec– extremely efficient. Therefore, it is inexpensive to train the model with more (recent) traffic traces on a regular basis.

5.2.1 Performance under Different SNR Levels. We studied performance under different SNRs. We utilized the AWGN symbols[4] to generate an SNR-based dataset using MATLAB simulations. The classification performance of our CNN model over different SNR levels is shown in Figure 5(a). We see that the accuracy and F1-score at 20 dB SNR is 99%. For the cases when the signal and noise power are the same at the receiver (i.e., 0 dB) , the model still achieves almost 85% accuracy with the input length 2000 or greater. Our model's performance with obfuscated traffic at a low SNR level is similar to a complex MC model where modulation is not obfuscated [15].



	0	1	2	3	4	5	6
0	0.9	0	0	0	0	0	0
1	0	1	0	0	0	0	0
2	0	0	1	0	0	0	0
3	0	0	0	1	0	0	0
4	0.1	0	0	0	0.9	0	0
5	0	0	0	0	0	1	0
6	0	0	0	0	0	0	1

(a) Accuracy (*L* is symbols per input). **(b) Confusion matrix**

Figure 5: Our attack performance (a) under different SNR levels, (b) in identifying the states of the obfuscated symbols. Here, 0: obfuscated BPSK, 1-3: 2-state, and 4-6: 4-state obfuscated QPSK, 16-QAM, 64-QAM, respectively.

5.2.2 Performance under Different States. We further studied whether our model can differentiate between the obfuscated symbols that are generated using a different number of states in TCM. Our dataset contains obfuscated symbols that are created using 2-state and 4-state TCM. Our observation revealed that the performance of 2-state and 4-state BPSK signals was indistinguishable, as the classifier identified them as the same class. We argue that this similarity in performance is due to the fact that BPSK symbols contain only one bit (either 0 or 1), and the introduction of additional states does not significantly alter the obfuscated symbol dependency, regardless of the number of states. Hence, we combined both types of obfuscated BPSK as one class, and then train the classifier. The classifier achieves 97% of overall testing accuracy with input length of 500 symbols when we combine 2- and 4-state obfuscated BPSK (see Figure 5(b)).

5.2.3 Performance under Mixed Traffic. We found that if a transmitter chooses to switch between obfuscated and non-obfuscated signals randomly (as a defense mechanism), our attack can differentiate between obfuscated and non-obfuscated symbols. To do that, we studied the performance of our model with combined non-obfuscated and obfuscated wireless traffic. The results show that our model can identify the underlying modulation scheme with 95% accuracy and F1-score, even under those conditions.

5.3 USRP Experiment

We further evaluate the performance via hardware experiments in a line-of-sight scenario. We ran the FCJ obfuscation technique on a USRP testbed (see Figure 6) that consists of two Ettus N210 devices controlled by the LabVIEW USRP driver and connected to an Intel Core i7 host running Windows 10 Enterprise. We collect over-the-air modulation-obfuscated transmissions, balancing the classes by taking an equal number of data samples and considering 100 symbols from each. Our model achieves around 60% overall accuracy, which is close to our simulation results for 100 symbols (see Figure 4(a)). We could not use more than 100 symbols due to the

[4] AWGN symbols refer to symbols that have been affected only by additive white Gaussian noise at the receiver, causing demodulation errors.

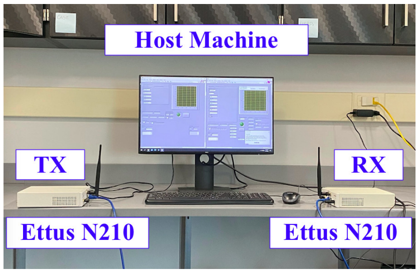


Figure 6: Two Ettus N210s placed 2 meters away from each other during our experiments. Here, they are kept close for illustration purposes only.

problem of accumulating carrier frequency offset (the gradual drift of the local oscillator frequency over time and deviating from the intended transmission frequency) in long sequences that arises due to the instability of the local oscillator used for signal generation and reception and mobility as well as frequency offset estimation errors at the receiver due to noise. We plan to address this problem in future work by more precisely estimating the frequency offset and applying a correction to the received signal.

6 POSSIBLE DEFENSE APPROACHES

Our results indicate that longer traffic traces increase the classification success rate, as shown in Figure 4. A possible mitigation is to transmit shorter series, or frames, such as 128 to 500 symbols (e.g., 16 bytes in BPSK), across the classes to ensure random success. However, this would impact the spectrum utilization and the receiver processing, as the receiver will need to process more frames (since the total message would be sent as a small number of symbols). For example, a 2304-byte frame payload will be 144 frames. We emphasize the significance of adopting a dynamic defense to disrupt or randomize the static pattern of a modulation obfuscation scheme, preventing adversaries from patiently learning the unique characteristics and launching successful MC attacks against a victim's traffic. This approach will have the potential to avert the risk of revealing the true modulation scheme.

7 CONCLUSION

In this paper, we exposed that existing modulation-obfuscation defenses cannot protect the system from MC attacks. Our CNN-based model is able to classify the obfuscated wireless traffic and successfully identify the true modulation scheme with 99% accuracy. Our simulations and USRP experiments show that our technique is robust even in noisy (AWGN channel) scenarios and achieved 85% accuracy for 0 dB SNR. In conclusion, our work underscores the limitations of modulation-obfuscation techniques and highlights the need for a dynamic approach to protect from MC attacks.

ACKNOWLEDGMENTS

This material is based upon work supported by the ESL Global Cybersecurity Institute at Rochester Institute of Technology. We also thank our shepherd and the anonymous reviewers for their feedback which greatly helped improve the paper.

REFERENCES

[1] Samer Hanna, Chris Dick, and Danijela Cabric. Signal processing-based deep learning for blind symbol decoding and modulation classification. *IEEE J. Sel. Areas Commun.*, 40(1):82–96, January 2022.
[2] Changbo Hou, Guowei Liu, Qiao Tian, Zhichao Zhou, Lijie Hua, and Yun Lin. Multisignal modulation classification using sliding window detection and complex convolutional network in frequency domain. *IEEE Internet Things J.*, 9(19):19438–19449, 2022.
[3] Ade Pitra Hermawan, Rizki Rivai Ginanjar, Dong-Seong Kim, and Jae-Min Lee. CNN-based automatic modulation classification for beyond 5G communications. *IEEE Commun. Lett.*, 24(5):1038–1041, 2020.
[4] Jered Pawlak, Yuchen Li, Joshua Price, Matthew Wright, Khair Al Shamaileh, Quamar Niyaz, and Vijay Devabhaktuni. A machine learning approach for detecting and classifying jamming attacks against UAVs. In *Proc. ACM Workshop Wireless Secur. Mach. Learn. (WiseML)*, Abu Dhabi, UAE, June 2021.
[5] Timothy James O'Shea, Tamoghna Roy, and T. Charles Clancy. Over-the-air deep learning based radio signal classification. *IEEE J. Sel. Topics Signal Process.*, 12(1):168–179, 2018.
[6] Erma Perenda, Sreeraj Rajendran, Gerome Bovet, Sofie Pollin, and Mariya Zheleva. Learning the unknown: Improving modulation classification performance in unseen scenarios. In *Proc. IEEE INFOCOM*, Virtual, May 2021.
[7] Wei Xiong, Lin Zhang, Maxwell McNeil, Petko Bogdanov, and Mariya Zheleva. Exploiting self-similarity for under-determined MIMO modulation recognition. In *Proc. IEEE INFOCOM*, pages 1201–1210, Virtual, July 2020.
[8] Mansi Patel, Xuyu Wang, and Shiwen Mao. Data augmentation with conditional GAN for automatic modulation classification. In *Proc. ACM Workshop Wireless Secur. Mach. Learn. (WiseML)*, pages 31–36, Virtual, July 2020.
[9] Alireza Bahramali, Milad Nasr, Amir Houmansadr, Dennis Goeckel, and Don Towsley. Robust adversarial attacks against DNN-based wireless communication systems. In *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, pages 126–140, Virtual/Republic of Korea, November 2021.
[10] Ya Tu, Yun Lin, Changbo Hou, and Shiwen Mao. Complex-valued networks for automatic modulation classification. *IEEE Trans. Veh. Technol*, 69(9):10085–10089, 2020.
[11] Aaron Smith, Michael Evans, and Joseph Downey. Modulation classification of satellite communication signals using cumulants and neural networks. In *Cogn. Commun. Aerosp. Appl. Workshop (CCAA)*, Cleveland, USA, June 2017.
[12] Sai Huang, Chunsheng Lin, Wenjun Xu, Yue Gao, Zhiyong Feng, and Fusheng Zhu. Identification of active attacks in internet of things: Joint model- and data-driven automatic modulation classification approach. *IEEE Internet Things J.*, 8(3):2051–2065, 2021.
[13] J S Atkinson, O Adetoye, M Rio, J E Mitchell, and G Matich. Your WiFi is leaking: Inferring user behaviour, encryption irrelevant. In *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, pages 1097–1102, Shanghai, China, April 2013.
[14] Guevara Noubir, Rajmohan Rajaraman, Bo Sheng, and Bishal Thapa. On the robustness of IEEE 802.11 rate adaptation algorithms against smart jamming. In *Proc. ACM Conf. Secur. Privacy Wireless Mobile Netw. (WiSec)*, pages 97–108, Hamburg, Germany, June 2011.
[15] Wei Xiong, Petko Bogdanov, and Mariya Zheleva. Robust and efficient modulation recognition based on local sequential IQ features. In *Proc. IEEE INFOCOM*, pages 1612–1620, Paris, France, April 2019.
[16] Yi Shi, Kemal Davaslioglu, and Yalin E. Sagduyu. Over-the-air membership inference attacks as privacy threats for deep learning-based wireless signal classifiers. In *Proc. ACM Workshop Wireless Secur. Mach. Learn. (WiseML)*, pages 61–66, Virtual, July 2020.
[17] Hanif Rahbari and Marwan Krunz. Full frame encryption and modulation obfuscation using channel-independent preamble identifier. *IEEE Trans. Inf. Forensics Security*, 11(12):2732–2747, 2016.
[18] Triet D. Vo-Huu and Guevara Noubir. Mitigating rate attacks through crypto-coded modulation. In *Proc. ACM Intl. Symp. Mobile Ad Hoc Netw. Comput. (MobiHoc)*, pages 237–246, Hangzhou, China, June 2015.
[19] Naureen Hoque and Hanif Rahbari. Poster: A tough nut to crack: Attempting to break modulation obfuscation. In *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, pages 2402–2404, Virtual/Republic of Korea, November 2021.
[20] Wei Xiong, Petko Bogdanov, and Mariya Zheleva. MODELESS: MODulation rEcognition with LimitEd SuperviSion. In *Proc. IEEE Int. Conf. Sens., Commun. Netw. (SECON)*, Virtual, July 2021.
[21] RF datasets for machine learning. Accessed: February 17, 2023.
[22] Chunsheng Lin, Juanjuan Huang, Sai Huang, Yuanyuan Yao, and Xin Guo. Features fusion based automatic modulation classification using convolutional neural network. In *Proc. IEEE INFOCOM Workshops (INFOCOM WKSHPS)*, pages 1099–1104, Virtual, July 2020.
[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances Neural Inf. Process. Syst.*, volume 30, 2017.