Robust Adversarial Attacks Against DNN-Based Wireless Communication Systems

Alireza Bahramali
University of Massachusetts Amherst
abahramali@cs.umass.edu

Milad Nasr
University of Massachusetts Amherst
milad@cs.umass.edu

Amir Houmansadr
University of Massachusetts Amherst
amir@cs.umass.edu

Dennis Goeckel
University of Massachusetts Amherst
dgoeckel@engin.umass.edu

Don Towsley
University of Massachusetts Amherst
towsley@cs.umass.edu

ABSTRACT

There is significant enthusiasm for the employment of Deep Neural Networks (DNNs) for important tasks in major wireless communication systems: channel estimation and decoding in orthogonal frequency division multiplexing (OFDM) systems, end-to-end autoencoder system design, radio signal classification, and signal authentication. Unfortunately, DNNs can be susceptible to adversarial examples, potentially making such wireless systems fragile and vulnerable to attack. In this work, by designing robust adversarial examples that meet key criteria, we perform a comprehensive study of the threats facing DNN-based wireless systems.

We model the problem of adversarial wireless perturbations as an optimization problem that incorporates domain constraints specific to different wireless systems. This allows us to generate wireless adversarial perturbations that can be applied to wireless signals on-the-fly (i.e., with no need to know the target signals a priori), are undetectable from natural wireless noise, and are robust against removal. We show that even in the presence of significant defense mechanisms deployed by the communicating parties, our attack performs significantly better compared to existing attacks against DNN-based wireless systems. In particular, the results demonstrate that even when employing well-considered defenses, DNN-based wireless communication systems are vulnerable to adversarial attacks and call into question the employment of DNNs for a number of tasks in robust wireless communication.

CCS CONCEPTS

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

KEYWORDS

Wireless Communication Systems; Adversarial Examples; Universal Perturbations; Deep Neural Networks

ACM Reference format:
Alireza Bahramali, Milad Nasr, Amir Houmansadr, Dennis Goeckel, and Don Towsley. 2021. Robust Adversarial Attacks Against DNN-Based Wireless Communication Systems. In *Proceedings of Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15–19, 2021 (CCS '21),* 16 pages.
https://doi.org/10.1145/3460120.3484777

1 INTRODUCTION

Deep Neural Networks (DNNs) are becoming central to various key wireless communication systems, thanks to their promising performances, and their computational efficiency. In particular, the wireless community has leveraged DNNs in state-of-the-art autoencoder wireless communication systems [18, 21, 29, 33, 34, 37], modulation recognition (radio signal classification) [31, 38, 39, 41, 48], and OFDM channel estimation and signal detection [49, 50]. Such wireless systems are crucial to various applications; for example, OFDM is a popular modulation scheme that has been widely used in many existing standards, such as 4G LTE and the IEEE 802.11 family [20, 22], and new standards such as 5G [9].

Unfortunately, whereas there is significant enthusiasm for the employment of DNNs [6], such emerging DNN-based wireless systems face a security threat: DNNs are known to be susceptible to *adversarial examples* [11, 17, 32], i.e., small perturbations added to the inputs of a DNN causing it to misclassify the perturbed inputs; consequently, DNN-based wireless communication systems are also susceptible to such attacks, which may impact the security (e.g., correctness, availability) of such systems. And, due to the penetration of these techniques in both contemporary military and commercial systems, the cost could be devastating. For example, robust attacks on modulation classification could compromise the performance of commercial software-defined radios or the ability of a military system to detect, intercept, and/or jam an enemy [10]. Importantly, particularly if the attack on the modulation classifier is undetectable as for our scheme proposed here, such a compromise can impact important tactical decisions based on enemy status. In the multitude of systems where OFDM plays a key role, an unexpected high bit-error-rate at the receiver due to adversarial perturbations can cause significant disruption; for example, the impact on the performance of the 4G Internet, which is tuned carefully at multiple levels to anticipate users' performance based on system state measurements, would be significant.

Our work: In this paper, by first identifying key criteria of an effective adversarial attack and then designing based on such, we perform the *first* comprehensive study of the effect of adversarial examples against DNN-based wireless systems. In this setting, the goal of an attacker is to transmit a well-crafted perturbation signal over a channel so that the underlying DNN-based wireless system (e.g., a radio signal classifier) fails and misclassifies the perturbed signals. Note that while there exists a large body of work on adversarial examples against image classification tasks, e.g., FGSM [17], such works cannot be trivially applied to the setting of wireless systems where the input signals to be perturbed are *unknown* to the adversary. Recent work [1, 12, 23, 24, 42–44] has aimed at implementing adversarial examples on wireless systems; however, none

of them are practical as they ignore domain constraints of wireless systems.

Therefore, in this work we present a *systematic and generic* mechanism for generating adversarial examples against DNN-based wireless systems, with the goal of generating adversarial perturbations that satisfy the domain constraints of wireless systems. Specifically, we present a generic framework that models the problem as an optimization problem and incorporates domain constraints specific to target wireless systems. We particularly enforce three key constraints in generating wireless adversarial examples: first, they should be **input-agnostic** meaning that the attacker generates the perturbation signal without any knowledge about the incoming (unknown) input wireless signals. This is essential as, unlike traditional targets of adversarial examples (e.g., image classification tasks), in DNN-based wireless systems the signal to be perturbed is *not* known a priori to the adversary. We particularly build on Universal Adversarial Perturbations (UAPs) [32], a recent adversarial perturbation approach that is input-agnostic. Second, the perturbation should be **undetectable** in that one should not be able to distinguish between a generated adversarial perturbation and natural noise expected from the wireless channel; otherwise, a defender can design a classifier to identify (then, remove) the adversarial perturbations based on the perturbation's power or statistical behavior. Finally, the wireless perturbations need to be **robust** against countermeasures meaning that the defender (e.g., a wireless decoder) should not be able to remove the perturbation from the received signal. Our framework is generic and can be used to enforce other domain constraints needed for a target wireless application, e.g., we also need to design in the presence of an unknown phase rotation between the attacker and the receiver.

Below, we describe how we enforce each of the three key wireless domain constraints through our generic optimization problem.

Generating input-agnostic perturbations. We model the problem of adversarial wireless perturbations as an optimization problem, and solve it to produce a *perturbation generator model (PGM)* able to generate an extremely large number of input-agnostic adversarial examples vectors (i.e., UAPs) for the target wireless application. Therefore, instead of applying a single UAP vector (that can be easily identified and removed as we show through experiments), in our setting the attacker picks and applies a random UAP adversarial example from a very large set of available UAPs produced by our PGM. We also show that our PGM is effective in a black-box scenario where the attacker generates adversarial perturbations based on a DNN substitute model and uses them to attack the original wireless DNN model. Our experiments demonstrate that *our techniques outperform state-of-the-art adversarial attack works* - especially in the presence of defense mechanisms. Note that recent works [12, 42] also use a DNN model to generate perturbations; however, they do not provide undetectability and robustness for perturbations, and they only consider a white-box scenario where the adversary is aware of the target wireless DNN model.

Undetectability. We tailor our PGM to each target wireless communication system by enforcing constraints specific to such systems, with the goal of making the attack undetectable. In particular, we use generative adversarial networks (GAN) to enforce an undetectability constraint on the UAPs generated by our PGM, and constrain them to follow a Gaussian distribution, which is the expected noise distribution for additive white Gaussian noise (AWGN) wireless channels. We show that by using such an undetectability constraint, the PGM can completely fool a discriminator function, i.e., a DNN classifier that tries to distinguish between adversarial perturbations and natural Gaussian noise. Based on our experiments, enforcing our undetectability constraint can decrease the $f1\_score$ of the discriminator from 0.99 to 0.6 (where an $f1\_score = 0.5$ is the best undetectability as it represents random guessing) with only a slight degradation in the performance of our attack. The score can be further decreased at the cost of further attack performance degradation.

Robustness. We also enforce a robustness constraint on the UAPs generated by our PGM. This constraint aims at maximizing the distances between different UAPs generated by our PGM; this is because if the UAPs are similar, as we show, an adversary can remove their effect with the knowledge of as little as a single pilot UAP vector. We analyze the robustness of our attack in different scenarios (Adversarial Training and Perturbation Subtraction) based on different amounts of knowledge available to the defender, and show that it provides high robustness against defense techniques; by contrast, we show that a single vector UAP, as proposed in previous work [43, 44], can be trivially detected and removed. Our analysis suggests that even if a defender has knowledge about the structure of our PGM, she will not able to mitigate the effects of the attack.

Evaluation on major wireless systems. We have implemented and evaluated our attacks on three classes of DNN-based wireless systems, specifically, autoencoder communication systems [18, 21, 29, 33, 34, 37], radio signal classification [31, 38, 39, 41, 48], and OFDM channel estimation and signal detection [49, 50]. We show that *for all three applications, our attack is highly effective in corrupting the functionality of the underlying wireless systems, and at the same time offers strong undetectability and robustness.*

We also propose two **countermeasures**, Adversarial Training and Perturbation Subtraction, based on the knowledge of a defender about the attack. We evaluate the performance of our attack and the single vector UAP attack against our own countermeasures as well as an existing countermeasure from the literature [23] called randomized smoothing. Our results show that our attack provides higher robustness against these countermeasures than previous adversarial attacks such as the single vector UAP attack. For instance, for the autoencoder communication system, in the presence of an adversarial training defense, our attack can increase the block-error rate (BLER) by *four orders of magnitude* with a perturbation-to-signal ratio (PSR) of $-6dB$. However, with a similar PSR, the single vector UAP attack [43, 44] is ineffective in the presence of the same defense mechanism. Similarly, in the OFDM application, our attack results in a *9X increase* in bit error rate while the impact of a single vector UAP is negligible. Furthermore, our attack is robust to the presence of a perturbation subtraction defense (as will be introduced), e.g., in the modulation recognition task, our attack reduces classification accuracy from 0.69 to 0.23 despite the defense mechanism (by contrast, the single UAP attack is not effective as it reduces accuracy from 0.69 to only 0.67).

In summary, we make the following major contributions:

- We propose an input-agnostic, undetectable, and robust adversarial attack against DNN-based wireless communication systems. We show that our attack is more effective than previous attacks; in particular, our results indicate that our PGM attack is more robust than using a single vector UAP attack against different countermeasures.
- We evaluate our attack against three classes of wireless systems by performing extensive experiments, hence showing that our PGM attack is not specific to a DNN-based wireless application and can be generalized to any DNN-based wireless application system.
- To our knowledge, we are the first to apply adversarial attacks against DNN-based OFDM channel estimation and signal detection systems, which comprise the physical layer in contemporary WiFi and cellular systems.
- We propose different countermeasure techniques and evaluate the robustness of the target wireless systems against adversarial attacks. We also compare the robustness of our attack to previous adversarial attacks in wireless systems and show that our attack is more robust against different countermeasures than previous attacks that are based on a single vector UAP.

2 BACKGROUND ON DNN-BASED COMMUNICATION SYSTEMS

Historically, the prosperity of wireless communications has relied on its own model-based design paradigms, where accurate mathematical models and expert knowledge are required. However, the traditional model-based wireless techniques cannot address the new challenges of emerging applications, such as communicating under excessively complex scenarios with unknown channel models, low-latency requirement in large-scale super-dense networks [3], etc. To tackle these challenges, DNNs have recently begun to play an important role in wireless communication applications due to their promising performance [6]. In this work, *we focus on three major DNN-based wireless communication systems* introduced below:

End-to-End Autoencoder Communication Systems: Despite the widespread use of provably optimal statistical models for the wireless physical layer, such models exhibit many imperfections and non-linearities in practical scenarios that can only be captured approximately. On the other hand, a DNN-based communication system such as an end-to-end autoencoder, that does not require a mathematically tractable model and can be optimized for a specific hardware configuration and channel, might better be able to handle such imperfections. Autoencoders are increasingly used for end-to-end learning of communication systems [33, 34, 37], and they can outperform contemporary modularized designs of these systems. Such systems implement their encoders and decoders using DNNs that are able to learn the construction and reconstruction process of the information as well as the noisy environment of the physical channel. For instance, Oshea et al. [37] consider a communication system design as an end-to-end reconstruction task that tries to jointly optimize transmitter and receiver components in a single process. As another example, Nachmani et al. [34] use Recurrent Neural Networks (RNNs) to decode linear block codes.

Modulation Recognition: Radio signal classification or modulation recognition is the task of classifying the modulation of a received radio signal to understand the type of communication scheme used in a wireless system. This can be considered as an N-class decision problem where the input is a complex baseband time series representation of the received signal. Modulation recognition is a key enabler for spectrum interference monitoring, radio fault detection, dynamic spectrum access, and many other wireless applications. Prior to using DNNs, modulation recognition has been achieved by carefully handcrafting specialized feature extractors for specific signal types and properties. Then, compact decision boundaries or statistically learned boundaries are derived from them with low-dimensional feature spaces.

Recently, conventional methods have been replaced with DNNs in modulation recognition [38, 39, 48], i.e., [38] applies Convolutional Neural Networks (CNNs) to the complex-valued temporal radio signal domain. They use expert feature based methods instead of naively learned features to improve classification performance. Furthermore, West et al. in [39] survey the latest advances in machine learning with DNNs by applying them to the task of modulation recognition. Their results show that the performance of modulation recognition system can be improved by novel architectures and training methods.

Signal Detection in OFDM Systems: Orthogonal frequency division multiplexing (OFDM) is a popular modulation scheme that has been widely used in wireless systems. OFDM is currently being deployed in many standards such as the downlink of 4G LTE and IEEE 802.11 family [20, 22]. Furthermore, OFDM is an important candidate for emerging standards such as 5G [9]. A key component of OFDM is channel state information (CSI), which refers to known channel properties of a communication link. CSI can be estimated using pilot signals that are known to the wireless system prior to the detection of the transmitted data. With the estimated CSI, transmitted symbols can be recovered at the receiver. Traditionally, least square (LS) and minimum mean-square error (MMSE) estimation methods are used for channel estimation in OFDM systems and have been thoroughly studied in the literature [28].

Recently, DNNs have been introduced in OFDM systems to estimate CSI and recover transmitted symbols at the receiver. [49] and [50] deploy DNNs for channel estimation and signal detection in OFDM systems in an end-to-end manner. In [50], Zhao et al. use CNNs to design an OFDM receiver that outperforms conventional OFDM receivers based on Linear Minimum Mean Square Error channel estimators. Ye et al. [49] use DNNs to estimate the CSI implicitly and recover the transmitted symbols directly instead of estimating CSI explicitly and detecting the transmitted symbols using the estimated CSI.

3 BACKGROUND ON ADVERSARIAL EXAMPLES

An adversarial example is a crafted input that fools a target classifier or regression model into making incorrect classifications or predictions. The adversary's goal is to generate adversarial examples by adding minimal perturbations to the input data attributes. Previous works [11, 17, 32, 35] have suggested several ways to generate

adversarial examples. Most adversarial example techniques generate perturbations specific to the input meaning that the adversary needs to be aware of the input to generate its corresponding adversarial perturbation, e.g., the Fast Gradient Sign Method (FGSM) [17] algorithm generates adversarial perturbations based on the input and the sign of the model's gradient. Recently, Moosavi-Dezfooli et al. [32] introduced universal adversarial perturbations (UAP) where the adversary generates adversarial examples that are independent of the inputs.

3.1 Adversarial Examples Against DNN-based Wireless Systems

Similar to other DNN-based applications, DNN-based wireless systems are susceptible to adversarial attacks [1, 2, 7, 8, 12, 13, 19, 23, 24, 42–45, 47]. Flowers et al. [13] use the FGSM method to evaluate vulnerabilities of the raw in-phase and quadrant (IQ) based automatic modulation classification task. There is a body of work [1, 12, 23, 24, 42–44] concentrated on using the adversarial input-agnostic technique proposed in [43] to attack DNN-based wireless applications, e.g., in [43], Sadeghi and Larsson design a single UAP vector that, when added to the received signal in a DNN-based modulation recognition system, causes the receiver to misclassify the modulation used by the transmitter. [44] uses the same approach in an end-to-end autoencoder communication system where an attacker can craft effective physical black-box adversarial attacks to increase error rates.

As opposed to using a single vector UAP, [12, 42] use a DNN to generate perturbations, e.g., Flowers et al. [12] encapsulate the learned model for perturbation creation in an Adversarial Residual Network (ARN) in a white-box scenario to evade DNN-based modulation classification systems. In [42], Restuccia et al. formulate a Generalized Wireless Adversarial Machine Learning Problem (GWAP) against modulation recognition systems where they analyze the combined effect of the wireless channel and adversarial waveform on the efficacy of the adversarial attacks. Instead of computing the optimal perturbation for each input, they generate adversarial perturbations in a white-box scenario over a set of consecutive input samples. However, as opposed to our work, their perturbations are not input-agnostic since they are generated based on the knowledge of a set of consecutive inputs, which renders the attack impractical. Note that both works only consider a white-box scenario where the attacker is fully aware of the target DNN model. Moreover, these works only apply their attacks to the modulation recognition task while in this work we consider three types of wireless communication tasks.

Furthermore, Kokalj-Filipovic et al. [26] propose two countermeasure mechanisms to detect adversarial examples in modulation classification systems based on statistical tests. One test uses Peak-to-Average-Power-Ratio (PAPR) of received signals, while another statistical test uses the Softmax outputs of the DNN classifier. Furthermore, [23] uses a certified defense based on randomized smoothing against the modulation recognition task. They augment the training dataset using Gaussian noise, and then use a hypothesis test to make predictions in the test phase.

4 SYSTEM MODEL

We begin by presenting the system models of the three DNN-based wireless applications targeted in this paper. A general DNN-based wireless communication system consists of a transmitter, a channel, and a receiver. The input of the system is a message $s \in \mathcal{M} = \{1, 2, ..., M\}$ where $M = 2^k$ is the dimension of $\mathcal{M}$ and k is the number of encoded bits per message. The transmitter employs a modulation scheme and sends the modulated symbols through the channel. The receiver receives the transmitted symbols and demodulates them to reconstruct the original symbols with the least error. Depending on the wireless application, each part of the system behaves differently, as overviewed in the following.

4.1 Autoencoder Communication Systems

In an autoencoder communication system, the transmitter and receiver are called *encoder* and *decoder*, respectively, and are implemented using DNNs. The transmitter generates a transmitted signal $x = e(s) \in \mathbb{R}^{2N}$ by applying the transformation $e : \mathcal{M} \rightarrow \mathbb{R}^{2N}$ to the message s. Note that the output of the transmitter is an N dimensional complex vector, which can be treated as a $2 \times N$ dimensional real vector. Then, the generated signal x is added to the channel noise, which we consider to be AWGN. Hence, the receiver receives a noisy signal $y = x + n$ and applies the transformation $d : \mathbb{R}^{2N} \rightarrow \mathcal{M}$ to create $\hat{s} = d(y)$, the reconstructed version of the message s. To enable a benchmark for comparison with the single vector UAP attack proposed in [44], we set $N = 7$ and $k = 4$; therefore, the input size of the DNN-based decoder is 2×7 where the first 7 elements are the in-phase components and the second 7 elements are the quadrature components of the received signal. For the training and test datasets, we randomly generate input messages.

4.2 Modulation Recognition Systems

DNN-based modulation recognition can be treated as a classification problem where the input is a complex base-band time series representation of the received signal and the goal of the model is to identify the modulation type of the transmitter. Similar to autoencoder systems, the modulated (transformed) input message x is added to the channel AWGN noise n, and the receiver receives a noisy complex base-band signal $y = x + n$. In this work, we will use the GNU radio ML dataset RML2016.10a [36] and its associated DNN [38]. This dataset is publicly available and also enables us to compare our attack with the single vector UAP attack proposed by [43].

The GNU radio ML dataset RML2016.10a contains 220000 input samples, where each sample is associated with one specific modulation scheme at a specific signal-to-noise ratio (SNR = 10 dB). It contains 11 different modulations: BPSK, QPSK, 8PSK, QAM16, QAM64, CPFSK, GFSK, PAM4, WBFM, AM-SSB, and AM-DSB. The samples are generated for 20 different SNR levels from -20 dB to 18 dB with a step size of 2 dB. The size of each input vector is 256, which corresponds to 128 in-phase and 128 quadrature components. Half of the samples are considered as the training set and the other half as the test set.

4.3 OFDM Channel Estimation and Signal Detection Systems

In an OFDM system, at the transmitter side, the transmitted symbols and pilot signals are converted into parallel data streams. Then, the inverse discrete cosine transform (IDFT) converts the data streams from the frequency domain to the time domain with a cyclic prefix (CP) inserted to mitigate the inter-symbol interference (ISI). The length of the CP should be no shorter than the maximum delay spread of the channel. Based on Ye et al. [49], we consider a sample-spaced multi-path channel described by the complex random vector h. On the receiver side, the received signal can be expressed as $y = x \circledast h + n$, where $\circledast$ denotes circular convolution while x and n represent the transmitted signal and the AWGN noise of the channel, respectively. At the receiver of the OFDM system, the frequency domain received signal is obtained after removing the CP and performing a discrete cosine transform (DFT).

We assume that the DNN model takes as input the received data consisting of one pilot block and one data block, and reconstructs the transmitted data in an end-to-end manner. To be consistent with [49], we consider 64 sub-carriers and a CP of length 16. Also, we use 64 pilots in each frame for channel estimation. Hence, the size of the input vector is 256, where the first 128 samples are the in-phase and quadrature components of the pilot block, and the second 128 samples are the in-phase and quadrature components of the following data block. We use [49]'s fully connected DNN model. For the training and test datasets, we randomly generate input messages.

5 ATTACK MODEL

In all of the aforementioned wireless applications, the goal of the **attacker**[1] is to transmit a well-designed perturbation signal over the channel such that the underlying DNN-based model fails to perform adequately. The generated perturbation is added to the transmitted signal and AWGN noise. The receiver receives the perturbed signal and applies the target DNN-based model to it. Note that in the OFDM system, the attacker adds the generated perturbation to each frame containing a pilot block and a data block. In the white-box scenario, we consider a strong attacker who is aware of the underlying DNN-based model, while in the black-box setting the adversary has no or limited knowledge of the underlying DNN-based model. Due to the challenges for the attacker to obtain robust phase synchronization with the transmitter at the receiver, which would likely require tight coordination with the communicating nodes, we assume that the perturbation generated by the attacker is subject to a random phase shift on the channel relative to the transmitter's signal.

As mentioned in Section 3, in the underlying wireless applications, the perturbation signal needs to be transmitted over-the-air, and therefore the perturbation signal should be input-agnostic i.e., universal (UAP). This allows the attacker to generate perturbation signals with no need to know the upcoming wireless signals. While some works [43, 44] have investigated such UAPs, they are easily detectable as the attacker uses a single perturbation vector. Such a perturbation vector can be inferred by the defender (e.g., through pilot signals) and consequently subtracted from the jammed signals. We demonstrate this through two defense mechanisms (namely, adversarial training and perturbation subtraction defenses presented in Section 9). For instance in an autoencoder communication system as shown in Figure 1, both of our defense mechanisms can easily defeat a single vector UAP attack as proposed by [43, 44].

[1]We use "attacker" and "adversary" interchangeably.

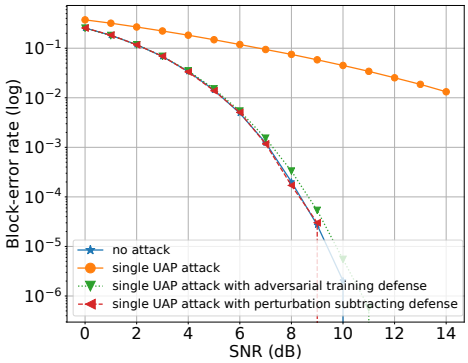


Figure 1: Performance of the single vector UAP attack in the presence of defense mechanisms for an autoencoder communication system. The single vector UAP attack [44] can be easily defeated using our two defense mechanisms.

Therefore, instead of designing a single UAP, our attacker learns the parameters of a PGM that generates separate perturbation vectors without any knowledge of the input. Using a PGM instead of a single noise vector provides the attacker with a large set of perturbations, and we can use existing optimization techniques such as Adam [25] to find the perturbations.

6 OUR PERTURBATION GENERATOR MODEL (PGM)

In this section, we provide details on how our attack is performed using a Perturbation Generator Model (PGM). Figure 2 illustrates the process.

6.1 General Formulation

We formulate the universal adversarial perturbation problem in a wireless communication system as:

$$\begin{aligned} &\arg\min_{\delta} \ ||\delta||_2 \\ &\text{s.t.} \ \ \forall y \in D : f(y+\delta) \neq f(y) \end{aligned} \tag{1}$$

where y is the transmitted signal plus the AWGN noise ($y = x + n$), f is the underlying DNN-based function in the wireless system, and D is the input domain of the wireless DNN model. The objective is to find a minimal (perturbation with minimum power) universal perturbation vector, δ, such that when added to *an arbitrary input* from a target input domain D, it will cause the underlying DNN-based model $f(.)$ to misclassify and therefore increase the DNN-based model loss function. Note that one cannot find a closed-form solution for this optimization problem since the DNN-based model $f(.)$ is a non-convex function, i.e., a deep neural network. Therefore,

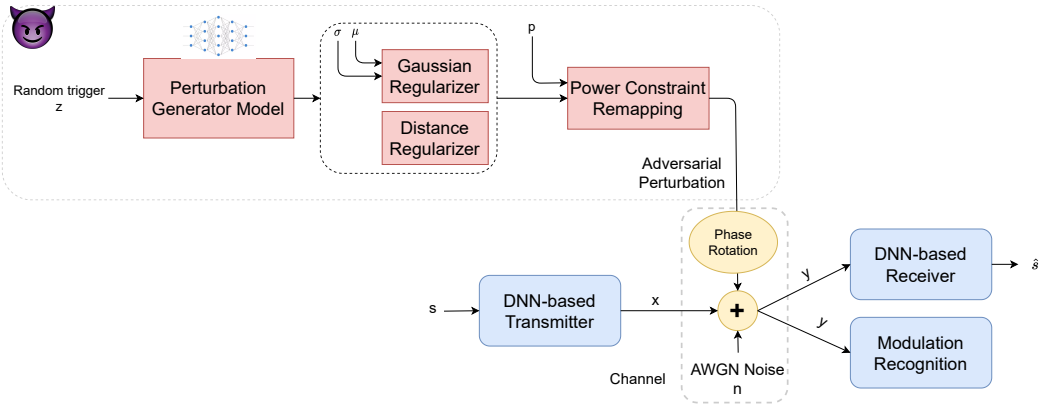


Figure 2: Our attack setting

(1) can be formulated as follows to numerically solve the problem using empirical approximation techniques:

$$\arg\max_{\delta} \sum_{y \in \mathcal{D}} l(f(y+\delta), f(y)) \tag{2}$$

where l is the DNN-based model loss function and $\mathcal{D} \subset D$ is the attacker's network training dataset.

As mentioned above, instead of learning a single UAP as suggested by [43, 44], we aim at learning the parameters of a PGM G to be able to generate UAPs without any knowledge of the system input. This generator model G will generate UAP vectors when provided with a random *trigger* parameter z (we denote the corresponding adversarial perturbation as $\delta_z = G(z)$), i.e., we can generate different perturbations for different values of z. Therefore, the goal of our optimization problem is to optimize the parameters of the PGM G (as opposed to optimizing a UAP δ in [44]). Hence, we formulate our optimization problem as:

$$\arg\max_{G} \mathop{\mathbb{E}}_{z \sim uniform(0,1)} [\sum_{y \in \mathcal{D}} l(f(y+G(z)), f(y))] \tag{3}$$

We can use existing optimization techniques (e.g., Adam [25]) to solve this problem. In each iteration of the training, our algorithm selects a batch from the training dataset and a random trigger z, then computes the objective function.

Algorithm 1 summarizes our approach to generate UAPs. In each iteration, Algorithm 1 computes the gradient of the objective function w.r.t. the perturbation for given inputs, and optimizes it by moving in the direction of the gradient. The algorithm enforces the underlying constraints of the wireless system using various remapping and regularization functions that will be discussed in the following sections. We use the iterative mini-batch stochastic gradient ascent [15] technique.

6.2 Incorporating Power Undetectability Constraint

As our first constraint on UAPs, we introduce a constraint on the attacker's perturbation power. We enforce this constraint to make the perturbations unnoticeable by the receiver. This constraint defines an upper bound on the generated perturbation's power.

Algorithm 1 Generating UAPs using PGM

$\mathcal{D} \leftarrow$ adversary training data
$f \leftarrow$ DNN-based model
$y \leftarrow$ training input
$l_f \leftarrow$ DNN-based loss function
$\mathcal{M} \leftarrow$ domain remapping function
$\mathcal{R} \leftarrow$ domain regularization function
$G(z) \leftarrow$ initialize the blind adversarial perturbation model parameters (θ_G)
$p \leftarrow$ the upper bound of the generated perturbations
$T \leftarrow$ number of epochs
for epoch $t \in \{1 \cdots T\}$ **do**
  for all mini-batch b_i in $\mathcal{D}$ **do**
    $z \sim$ Uniform (0,1)
    Rotate $\mathcal{M}(y, G(z))$ based on the channel phase shift
    $J = -(\frac{1}{|b_i|} \sum_{x \in b_i,} l(f(\mathcal{M}(y, G(z), p)), f(x))) + \mathcal{R}(G(z))$
    Update G to minimize J
  end for
end for
return G

To enforce this power constraint, we use a *remapping function* $\mathcal{M}$ while creating the UAP. Here $\mathcal{M}$ adjusts the perturbed signal to comply with the power constraint. Therefore, we reformulate our optimization problem by including the remapping function $\mathcal{M}$:

$$\arg\max_{G} \mathop{\mathbb{E}}_{z \sim uniform(0,1)} [\sum_{y \in \mathcal{D}} l(f(\mathcal{M}(y, G(z), p)), f(y))] \tag{4}$$

where p is the upper bound on the power of the generated perturbations. Each time we want to create the perturbation signal, we check its power and, if it violates the constraint, we normalize it to satisfy the power constraint. The following shows the remapping function we used to preserve the power constraint of the perturbations in the target wireless applications.

$$\mathcal{M}(y, G(z), p) = y + \begin{cases} \sqrt{p} \frac{G(z)}{||G(z)||_2}, & ||G(z)||_2^2 > p, \\ G(z), & ||G(z)||_2^2 \leq p. \end{cases}$$

Algorithm 2 GAN-based noise regularizer

$\mathcal{D} \leftarrow$ training data
$f \leftarrow$ DNN-based model
$G \leftarrow$ PGM
$D \leftarrow$ discriminator model
$\mu, \sigma^2 \leftarrow$ target desired Gaussian distribution parameters
for $t \in \{1, 2, \cdots, T\}$ **do**
  $z' \sim$ Gaussian(μ, σ^2)
  $z \sim$ Uniform
  train D on $G(z)$ with label 1 and z' with label 0
  train G on $\mathcal{D}$ using regularizer $\mathcal{R}$
end for
return G

6.3 Incorporating Statistical Undetectability Constraint

As mentioned earlier, the adversarial perturbation is not noise but a deliberately optimized vector in the feature space of the input domain, and hence is easily distinguishable from the expected behavior of the noise in the communication system environment. To make our adversarial perturbation undetectable to statistical inference, we enforce a statistical behavior (such as Gaussian behavior) that is expected from the physical channel of a communication system on our adversarial perturbations. We use a *regularizer* $\mathcal{R}$ in the training process of our PGM to enforce a Gaussian distribution for the perturbation. To do this, we use a generative adversarial network (GAN) [16]: we design a discriminator model $D(G(z))$ that tries to distinguish the generated perturbations from a Gaussian distribution. Then we use this discriminator as our regularizer function to enforce the distribution of the crafted perturbations to be similar to a Gaussian distribution. We simultaneously train the blind perturbation model and the discriminator model. Hence, we rewrite (4) as follows:

$$\arg\max_{G} \mathop{\mathbb{E}}_{z \sim uniform(0,1)} [(\sum_{y \in \mathcal{D}} l(f(\mathcal{M}(y, G(z), p)), f(y))) + \alpha \mathcal{R}(G(z))] \quad (5)$$

where α is the weight of the regularizer relative to the main objective function. Algorithm 2 shows the details of our technique to generate perturbations that follow a Gaussian distribution with an average μ and standard deviation σ. Note that we use the Gaussian distribution as the desired distribution for our noise since it is expected from the environment where a wireless communication system operates, and thus it cannot be distinguished from the normal AWGN noise of the channel.

6.4 Incorporating Robustness Constraint

As mentioned earlier, using a PGM instead of a single UAP to perform the adversarial attack provides the adversary an extremely large set of perturbations. This makes the attack more robust against countermeasures compared to the single vector UAP attack. However, if the generated perturbations are similar to each other, an ad hoc defender (as discussed in Section 9) can use pilot signals to accurately estimate (and remove) the perturbations. To prevent this, we force Algorithm 1 to generate non-similar perturbations. To this aim, we add the l_2 distance between consecutively generated perturbations as a regularizer to the objective function. Hence, in the training process, our model tries to maximize the distance between perturbations. In Section 9, we see that incorporating this constraint prevents an ad hoc defender from accurately estimating the generated perturbations.

6.5 Incorporating Channel Phase Rotation

As mentioned in Section 5, in order to model the lack of phase synchronization between the attacker and the transmitter, we add a relative random phase to the perturbation generated by the perturbation model and rotate its signal. For each adversarial perturbation, we generate a random phase θ and rotate the perturbation based on it. Note that the channel effect is applied on the perturbation after applying all of the mentioned constraints on the perturbation. Assume $\delta = \mathcal{M}(x, G(z))$ is a perturbation generated by the PGM after applying the power constraint, and δ_R and δ_I are the real and imaginary parts of the perturbation, respectively. Using the random phase shift caused by the channel the rotated perturbation can be derived as follows:

$$\text{For all } i = 1, 2, \cdots, N: \quad \begin{cases} \delta'_{i,R} = \delta_{i,R} \cos(\theta) - \delta_{i,I} \sin(\theta) \\ \delta'_{i,I} = \delta_{i,I} \cos(\theta) + \delta_{i,R} \sin(\theta) \end{cases} \quad (6)$$

where N is the length of the perturbation, and δ'_R and δ'_I are the real and imaginary parts of the rotated perturbation.

7 EXPERIMENTAL SETUP

For the three target wireless applications, we use the same setup as their original papers [37, 38, 49]. The target models for each application are the same as the ones proposed in their original papers. Table 4 in Appendix A illustrates the input size and parameters of each target DNN model. To enable a benchmark for comparison, we obtain the code of the UAP adversarial attack proposed by [43, 44] and transform it from TensorFlow to Pytorch. We then compare the results of our adversarial attack using a PGM with the single vector UAP adversarial attack. On the other hand, we are the first to apply adversarial attacks on the OFDM channel estimation and signal detection task; hence, we implement both the PGM attack and the single vector UAP attack against the OFDM system for comparison.

We use fully connected layers for the PGM with different numbers of hidden layers and different numbers of neurons in each layer based on the wireless application. Table 1 contains the details of the structure used for each PGM.

For the discriminator model mentioned in Section 6, we use a fully connected DNN model with one hidden layer of size 50 and a ReLU activation function. The discriminator generates a single output that can be interpreted as the probability of following a Gaussian distribution for a signal. In the training process of our discriminator, we set μ and σ to the average of the means and standard deviations of the generated perturbations. To train the discriminator we use the Adam [25] optimizer with a learning rate of 10^{-5}.

Table 1: Details of the perturbation generation model in each wireless application

	Autoencoder End-to-End Communication	Modulation Recognition	OFDM Channel Estimation
input size	2×7	2×128	256
hidden layers sizes	100	5000, 1000	5000, 1000
hidden layers activations	ReLU	Leaky ReLU, Leaky ReLU	Leaky ReLU, Leaky ReLU
loss function	Cross Entropy	Cross Entropy	MSE
metric	Block-Error Rate (BLER)	Accuracy	Bit Error Rate (BER)
optimizer	Adam	Adam	Adam
learning rate	10^{-4}	10^{-3}	10^{-2}

8 EVALUATION OF ATTACK PERFORMANCE

In this section, we first evaluate our attack against three target wireless applications without any undetectability constraint. We also compare our attack with the single vector UAP attack. As mentioned in Section 5, to consider the channel effect, we add a relative random phase shift to our perturbations. Second, we evaluate our attack while enforcing the undetectability constraint mentioned in Section 6 on the autoencoder communication system. Figure 3 highlights the performance of our attack compared to the single vector UAP attack for the three target applications. We discuss the results in detail as follows.

8.1 Performance Without Statistical Undetectability

First, we evaluate the performance of our attack without the statistical undetectability constraint, but only with the basic, power undetectability constraint (the robustness and phase rotation constraints are always enforced).

Note that, in the figures of the modulation recognition application, we use the PSR of the perturbation for the x-axis while for the other two applications, the SNR is used for the x-axis. This is because the GNU radio ML dataset that we use in our modulation recognition experiments only contains samples at the specific SNR of 10 dB.

Autoencoder Communication System: Figure 3a shows the block-error rate (BLER) performance of the autoencoder communication system under adversarial attack while using a PGM and the single vector UAP attack proposed in [44]. To be consistent with [44], we set $N = 7$ and $k = 4$. We sweep SNR from $0dB$ to $14dB$ with steps of $1dB$, and for each value, we calculate the BLER of the autoencoder system. Similar to [44], to compare the power of the adversarial perturbation at the receiver with the received signal, we introduce a parameter, the *perturbation-to-signal ratio (PSR)*, which equals the ratio of the received perturbation power to the received signal power.

We see that *for different PSR ratios, using a PGM to generate the UAPs increases the performance of the adversarial attack in comparison to learning a single UAP.* As mentioned in Section 6, the reason is that by learning the parameters of a generator model instead of learning a single perturbation vector, we can leverage existing learning techniques (such as momentum-based ones) to ease the process of learning and prevent common learning problems such as getting stuck in local minima.

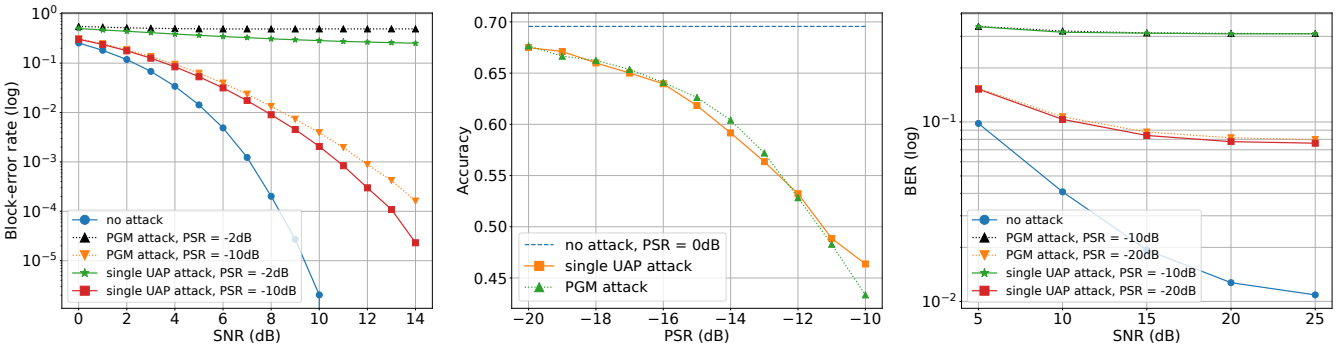
Modulation Recognition: Figure 3b shows the performance of our adversarial attack using a PGM against the modulation recognition application over different values of PSR. The GNU [36] dataset contains samples for different values of SNR; however, we apply our attack using samples with SNR of 10 dB. Figure 3b also compares our attack with the single vector UAP attack proposed by [43] against the modulation recognition task. We see that using a PGM does not provide a significant improvement over the single vector UAP method in terms of attack performance when the wireless system does not employ defenses. In Section 9, we will demonstrate that the PGM is significantly more robust in the presence of defense mechanisms compared to the single vector UAP technique.

OFDM Channel Estimation and Signal Detection: Similar to the above two applications, we apply each of the single vector UAP and adversarial PGM attacks on the DNN-based OFDM system proposed by [49]. Because there is no reported prior adversarial attack on the DNN-based OFDM system, we do not have a baseline for comparison. Figure 3c shows the Bit Error Rate (BER) of the OFDM system against the two mentioned adversarial attacks. The SNR is varied from $5dB$ to $25dB$ and we evaluate our attack for two values of PSR, $-10dB$ and $-20dB$. We see that using a PGM improves the adversarial attack slightly compared to the single vector UAP attack if the wireless system does not employ any defenses. However, in Section 9, we will see that using our PGM makes the attack significantly more robust against possible defenses.

8.2 Performance With Statistical Undetectability

In this section, we evaluate our attack while enforcing a statistical undetectability constraint using a GAN in the autoencoder communication system. Note that this technique is easily applicable to other systems since it uses a discriminator network independent of the underlying DNN model in the wireless application. To investigate the undetectability of the generated perturbations, we train our discriminator in two scenarios: first, we train our discriminator to be able to distinguish between adversarial noise and natural Gaussian noise without enforcing an undetectability constraint on the adversarial generator model. In the second scenario, we train the discriminator while enforcing a Gaussian distribution on our adversarial noise in the PGM. The evaluation metric for the discriminator is the $f1\_score$ which can be interpreted as a weighted average of the precision and recall metrics.

Figure 4 shows the performance of the discriminator as well as the generator in these two scenarios for different values of α



(a) Autoencoder communication system **(b) Modulation recognition system, SNR=10dB** **(c) OFDM system**

Figure 3: Performance of our attack and the single vector UAP attack for the three target wireless systems in the absence of defenses. The case where the receiver employs significant defenses, which is the case of most interest, is considered in Section 9, where the significant advantages of our attack will be demonstrated.

(denoting the strength of the undetectability constraint used in (5)) while the PSR of the generated perturbations is $-6dB$. For each scenario and each value of α, we have evaluated the discriminator for different values of SNR and reported the average $f1\_score$. We see that without enforcing the undetectability constraint in the PGM ($\alpha = 0$), the discriminator is able to distinguish between generated adversarial noise and Gaussian noise: the $f1\_score$ is nearly 1. On the other hand, when we enforce an undetectability constraint in the PGM with $\alpha = 50$, the average $f1\_score$ is 0.61, which means that the discriminator misclassifies the generated perturbations as Gaussian noise to some extent while the performance of our attack slightly decreases compared to a scenario where there is no undetectability constraint. By increasing α to 500, the average $f1\_score$ becomes 0.53 making our generated noise more undetectable since the discriminator cannot distinguish between Gaussian noise and adversarial noise. On the other hand with $\alpha = 500$, our attack performs much worse than the case with $\alpha = 50$; however, it still degrades the performance of the autoencoder significantly. **Therefore, by enforcing an undetectability constraint, we can achieve high undetectability with only a small degradation in the performance of our attack.**

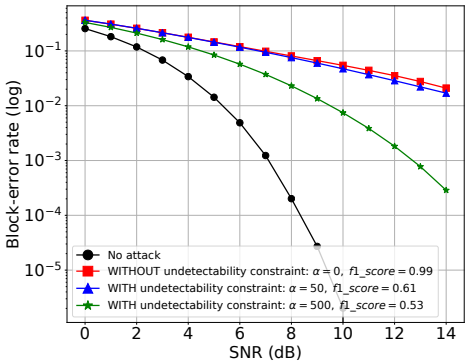
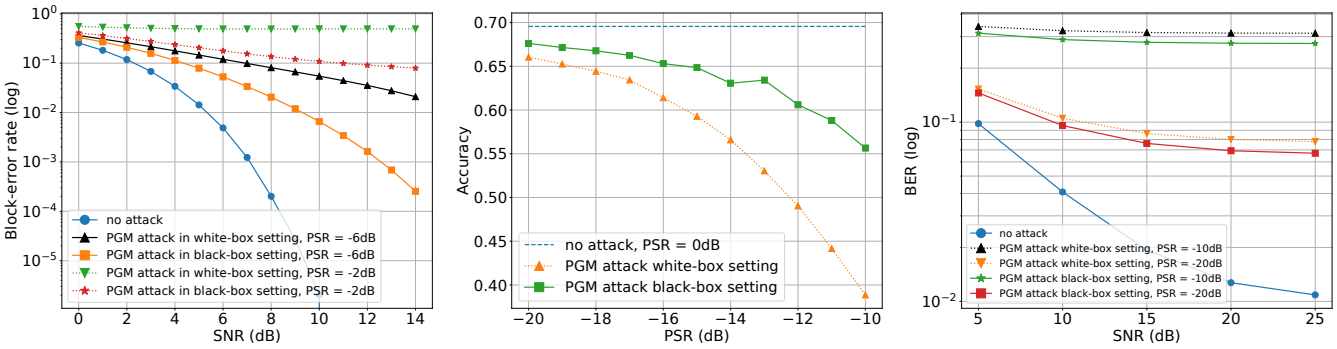


Figure 4: Performance of the autoencoder communication system against PGM attack with and without the undetectability constraint and the corresponding $f1\_score$ of the discriminator.

8.3 Performance In the Black-box Setting

All the previous evaluations were made assuming a (strong) white-box adversary (see Section 5), i.e., an adversary who is aware of the underlying DNN-based model including its structure and parameters. In this section, we evaluate our PGM in the black-box setting where the attacker does not have any knowledge about the underlying DNN model. Instead, the attacker uses its own *substitute model* and then designs a white-box attack for it, as it has the perfect knowledge of the substitute model. The attacker then uses the crafted perturbations to attack the original unknown underlying DNN model. This is called a black-box adversarial attack [40]. Note that in this setting, we assume that the attacker has access to a training dataset from the same distribution of training data used by the underlying target model.

Using this approach, for each target application, we use a substitute model to design our PGM. We then use the PGM learned on the substitute model to generate perturbations and apply them on the original wireless DNN model. Note that this approach is general such that the attacker can use any other DNN-based wireless model to generate perturbations and attack the original underlying DNN model. Table 5 in Appendix A shows the structure and parameters of the substitute models. Figure 5 compares the performance of our PGM attack in white-box and black-box scenarios for three target wireless applications. **Although our PGM attack performs slightly worse in the black-box setting than in the white-box setting, we observe that the attack is still effective and degrades the performance of the three underlying DNN models.**



(a) Autoencoder communication system **(b) Modulation recognition system, SNR=10dB** **(c) OFDM system**

Figure 5: Performance of our attack with white-box and black-box scenarios for the three target wireless systems. Although our PGM attack performs slightly worse in the black-box setting than in the white-box setting, we observe that the attack is still effective and degrades the performance of the three underlying DNN models.

8.4 Attack's Computational Complexity

We compare the complexity of our PGM attack algorithm with the single vector UAP attack. Table 2 shows the average time required to train a PGM compared to the average time a single vector UAP attack needs to be trained for each target wireless application. Note that the training time is calculated for a single PSR. Table 2 also shows the average runtime of generating a single perturbation using the PGM attack during testing. We do not define this runtime in the single vector UAP attack since the single perturbation is generated once and applied across all the inputs. Furthermore, We see that although there are more parameters to learn for a PGM attack, **the times to perform the PGM and single vector UAP attacks are comparable**. The reason is that by learning the parameters of a PGM instead of learning a single perturbation vector, we can leverage existing learning techniques (such as momentum-based ones) to speed up the process and prevent common learning problems such as getting stuck in local minima. Moreover, the attacker needs to train the PGM **only once** for each wireless application and he can use it for any input sample to attack the wireless system.

9 COUNTERMEASURES AND ROBUSTNESS ANALYSIS

In this section, we propose two countermeasures as defense mechanisms against adversarial attacks in wireless communication systems. We apply these defenses to both the single vector UAP and our PGM attacks and evaluate the performance of these attacks. The performance of the defenses depends on what the defender knows about the attack algorithms. In the following, we make assumptions regarding what the adversary knows about both attacks.

Single Vector UAP Attack: We mentioned earlier that a single UAP vector can be identified using pilot signals. A defender can transmit known pilot signals and receive the perturbed signal. Since the transmitted signal is known, the defender can subtract it from the received signal and produce an estimate of the perturbation. Therefore, in the single vector UAP attack, we assume that the defender has knowledge about the perturbation generated by the attacker.

Perturbation Generator Model: When the attacker uses a PGM, the defender cannot identify the perturbations since the PGM generates a different perturbation vector for each input sample. Hence, in this case, the defender can only obtain knowledge about the PGM or an estimate of the generated perturbations. Based on this knowledge, we assume three scenarios for the defender:

- *Ad hoc defender:* In this scenario, the defender is not aware of the PGM, and similar to the defense mechanism proposed for the single vector UAP attack, the defender uses pilot signals to estimate the generated perturbations, e.g., the defender transmits pilot signals and subtracts them from their corresponding received signals, then she takes an average of the results to obtain an estimate of the perturbations generated by the PGM.
- *Structure-aware defender:* In this scenario, we assume that the defender is aware of the structure of the PGM but not its parameters. Hence, the defender needs to train the PGM on her own training data and obtain the learned parameters. We also assume that the defender has the same training dataset as the adversary.
- *Model-aware defender:* In this scenario, we assume that the defender is aware of both the structure of the adversary's PGM and its learned parameters. This is an impractical assumption as the defender cannot obtain the learned parameters of the PGM using pilot signals or any other techniques; we still evaluate our attacks against this impractical adversary.

Note that, in the above mentioned scenarios, we assume that the defender is aware of the power constraint (p) of the adversary. We also evaluate our system against random smoothing defense in the modulation recognition task; however, it is not as effective as other proposed defenses.

Table 2: Comparing the complexity of the PGM attack and the single vector UAP attack

	PGM Attack			Single Vector UAP Attack		
	Training Time	Test Time	Number of Parameters	Training Time	Test Time	Number of Parameters
Autoencoder Encoder	53s	1.69×10^{-8}s	2914	67s	–	14
Modulation Recognition	5s	2.34×10^{-7}s	6542256	20s	–	256
OFDM Channel Estimation	46s	8.78×10^{-8}	6542256	45s	–	256

Algorithm 3 Adversarial Training against adversarial attacks in wireless communication systems

Randomly initialize underlying DNN-based network N
$\mathcal{D}^{tr} \leftarrow$ training data
$\mathcal{L}_f \leftarrow$ DNN-based loss function
$\mathcal{M} \leftarrow$ domain remapping function
$\mathcal{R} \leftarrow$ domain regularizations function
$T \leftarrow$ epochs
$Z \leftarrow []$ // List of adversarial perturbations
for epoch $t \in \{1 \cdots T\}$ **do**
  Train the model N for one epoch on training dataset $\mathcal{D}^{tr}$
  $Z \leftarrow$ generate crafted adversarial samples using generated perturbations by Algorithm 1 (or the single vector UAP)
end for
$\mathcal{D}^{tr}$.extend($\mathcal{D}^{tr} + Z$)
return N

9.1 The Adversarial Training Defense

Many defenses have been designed for adversarial examples in image classification applications, particularly, adversarial training, gradient masking, and region-based classification. In adversarial training [27, 30, 46], in each iteration of training, the defender generates a set of adversarial examples and uses them in the training phase by expanding the training dataset. In our work, we use adversarial training where the defender uses UAPs crafted by our attack to make the target DNN-based wireless model robust against the attacks. The defender trains the DNN-based model for one epoch and then generates input-agnostic adversarial perturbations from all possible settings using Algorithm 1. Then, she extends the training dataset by including all of the adversarial samples generated by the adversary and trains the DNN-based model on the augmented training dataset. Algorithm 3 is a high level description of our defense algorithm.

In the case of a single vector UAP attack, the defender uses a single perturbation vector to generate adversarial samples and train the target model. For the perturbation generator attack, we assume that the defender is structure-aware and uses her trained PGM to generate adversarial samples and train the target model.

9.2 The Perturbation Subtraction Defense

This is a defense specialized to our domain. In this defense, at the receiver side, the defender performs operations on the perturbed received signal based on her knowledge of the adversary to remove the effect of the perturbation and reconstruct the originally transmitted signal. For the single vector UAP attack, since we assume that the defender has identified an estimate of the perturbation, she can easily subtract her estimate of perturbation from the received signal and obtain the originally transmitted signal. If the adversary uses a PGM, as mentioned above, we consider three scenarios based on the knowledge of the defender: ad hoc defender, structure-aware, and model-aware defenders. In all of the scenarios, once the defender receives the received perturbed signal, she generates a perturbation using her PGM and subtracts it from the received signal.

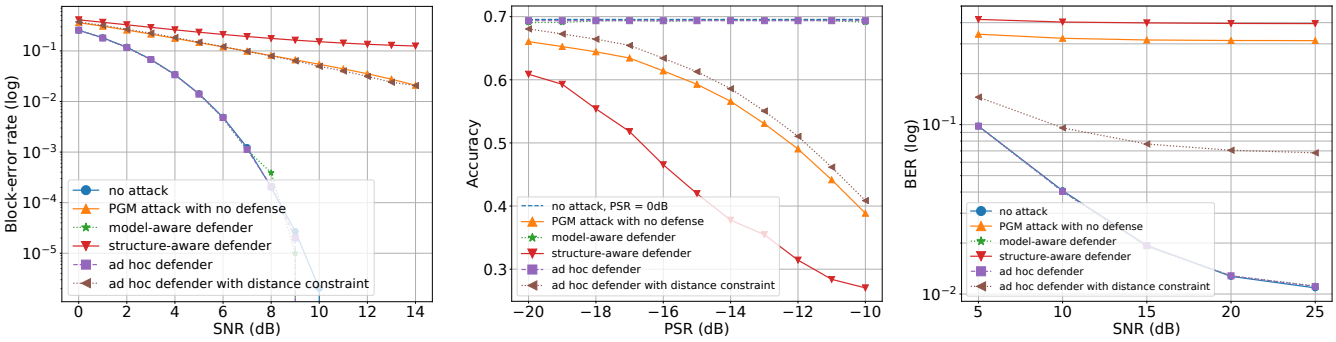
9.3 Randomized Smoothing Defense

We evaluate our attack against a defense mechanism called *certified defense* [4, 5, 14] that relies on *randomized smoothing*. Randomized smoothing is a certified defense approach against adversarial examples, which augments the training set with Gaussian noise to increase the robustness of the classifier to multiple gradient directions. The standard deviation of the Gaussian noise σ and the number of the noisy samples added to each training sample k are the two parameters the defender can control in randomized smoothing. In the prediction phase, for each perturbed test sample, we draw k Gaussian noise samples with the same standard deviation and label them with the DNN model. The defense results can be certified with a desired confidence using a two-sided hypothesis test. If the test is satisfied, the DNN model is very confident in its prediction, and if not, the DNN model abstains and does not make a prediction. Recently, Kim et al. [23] utilize randomized smoothing as a defense mechanism against single vector UAP attack in the modulation recognition application. We use the same approach to evaluate randomized smoothing against our attack.

9.4 Results

First, we evaluate the performance of our PGM attack against the perturbation subtraction defense for the three defender types. Figure 6 shows the performance of each target wireless communication system against the PGM adversarial attack in the presence of the ad hoc, structure-aware, and model-aware defenders. In the experiments involving the ad hoc defender, we use 10000 pilot signals and take the average of the obtained noise to estimate the perturbation. We see that a model-aware defender can completely degrade the performance of the adversarial attack; however, as mentioned earlier, a model-aware defender is not practical. Although our adversarial attack provides the attacker a large set of adversarial perturbations, using the same PGM removes the effect of the adversarial attack. However, a structure-aware defender that trains her PGM and obtains its parameters that are different from the PGM parameters used by the attacker not only degrades the effect of the attack but improves its performance as well.

Furthermore, we see that the ad hoc defender also degrades the performance of the adversarial attack using pilot signals, which shows that the generated perturbations are too similar to each other



(a) Autoencoder communication system, PSR=$-6dB$ **(b) Modulation recognition system, SNR=$10dB$** **(c) OFDM system, PSR=$-10dB$**

Figure 6: Performance of the three target communication systems against the PGM attack with the presence of a perturbation subtraction defense for the structure-aware, model-aware, and ad hoc defenders. Enforcing the distance constraint in the training process of the PGM removes the effect of the ad hoc defender.

such that the defender cancels them by just subtracting out a simple estimate of the perturbations. To prevent this, the attacker enforces a distance constraint as discussed in Section 6 to maximize the l_2 distance between the generated perturbations. Figure 6 shows that by enforcing the distance constraint in the training process of the PGM, the attacker removes the effect of the ad hoc defender. In such a case, the ad hoc defender cannot generate a precise estimate of the perturbations. We believe that even if a defender takes the distance constraint into account by performing adversarial training, the attack remains robust as we will show in this Section that the PGM attack is robust against adversarial training defense.

We also compare the performance of our attack and the single vector UAP attack against different defense algorithms. Figure 7 shows the performance of our adversarial attack and the single vector UAP attack against the mentioned defense methods. Note that with both defense methods, we only consider a structure-aware defender that only knows the structure of the PGM model used by the attacker and not its parameters. Furthermore, as mentioned above, in the single vector UAP attack, we assume that the defender can use pilot signals to estimate the single perturbation. In our experiments, we obtain this estimate by sending 10000 pilot signals and averaging their resulted perturbation. Hence, in the adversarial training defense, we only use the estimated perturbation to generate adversarial samples and train the target DNN-based model.

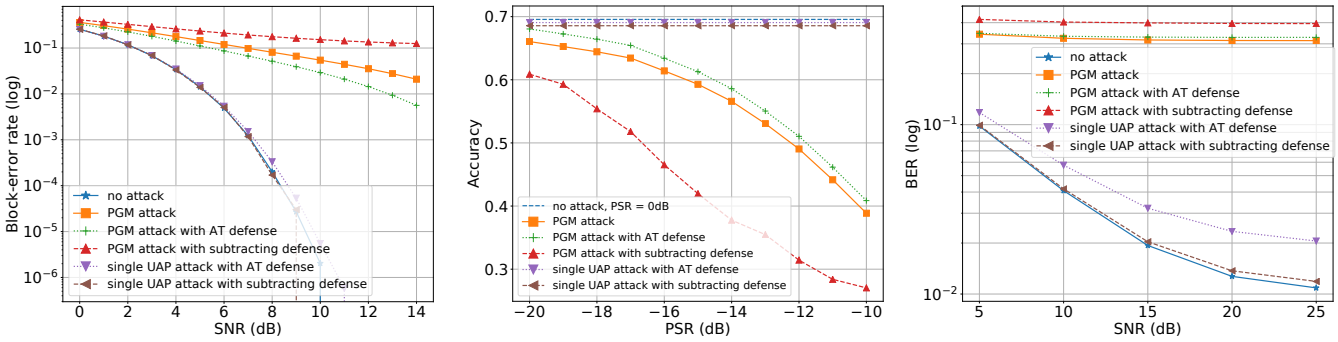
In the single vector UAP attack, as Figure 7 illustrates, subtracting the estimated noise from the received signal at the receiver side destroys the effect of the attack completely; However, the same defense mechanism is not effective against the PGM attack which implies that the PGM attack shows more robustness against the perturbation subtraction defense compared to the single vector UAP attack.

Furthermore, while using the adversarial training defense by the defender, our PGM attack is still more robust than the single vector UAP attack. The reason is that the underlying DNN-based model can learn the single perturbation generated by the attacker for all of the inputs. Using a PGM enables the attacker to access an extremely large set of perturbations which makes it infeasible for the defender to learn the DNN-based model based on all of them. **Based on our results, we conclude that using a PGM to perform adversarial attacks against wireless communication systems is more robust against various defense mechanisms in comparison to using only a single perturbation vector.**

We also evaluate the performance of our PGM attack on modulation recognition while deploying the randomized smoothing defense. Table 3 shows the performance of the modulation recognition system for different values of k and *sigma* and a 95% confidence threshold. We see that for some values of k and σ using this defense alleviates the effect of the PGM attack and increases the accuracy of the DNN-based classifier in the modulation recognition task. However, these high accuracies are just for the samples satisfying the confidence threshold, while as Table 3 illustrates the classifier has confidence to make predictions for only less than **half** of the test samples, thus showing that this defense is *not practical against our attack*. Moreover, for real-time applications such as autoencoder communication systems and OFDM systems, not having enough confidence for half of the received signals would be unacceptable.

10 DISCUSSIONS

Our work demonstrates a major vulnerability of the rapidly emerging DNN-based wireless communication systems. We show that an adversary can leverage well-crafted universal adversarial perturbations that are tailored to specific domain constraints of these systems to degrade their functionality. We see that wireless domain constraints bring up major challenges (synchronizing the attacker and the receiver) for the succession of the attack. However, important features of adversarial attacks such as transferability and learnability enable us to overcome these challenges by designing a input-agnostic, undetectable, and robust adversarial attack. Furthermore, our experiments show that existing defense mechanisms against adversarial examples cannot mitigate the performance of our attack urging that more sophisticated defense mechanisms should be deployed to alleviate this vulnerability.



(a) Autoencoder communication system, PSR=$-6dB$ **(b)** Modulation recognition system, SNR=$10dB$ **(c)** OFDM system, PSR=$-10dB$

Figure 7: Performance of the single vector UAP attack and PGM attack against adversarial training and perturbation subtraction defenses. The adversarial training defense is less effective than perturbation subtraction; however, even with the adversarial training defense, our attack is still robust compared to the single vector UAP attack.

Table 3: Performance of the modulation recognition system against PGM attack while deploying randomized smoothing defense.

Randomized smoothing parameters	Modulation recognition accuracy	Fraction of samples satisfying confidence threshold
$k = 10, \sigma = 0.01$	25.76%	0.2576
$k = 20, \sigma = 0.01$	16.04%	0.5315
$k = 10, \sigma = 0.001$	87.97%	0.2984
$k = 20, \sigma = 0.001$	81.91%	0.4712
$k = 10, \sigma = 0.0001$	87.68%	0.3239
$k = 20, \sigma = 0.0001$	89.99%	0.4976

Similar to the previous works on this (emerging) topic [1, 12, 23, 24, 42–44], our evaluations mainly rely on simulations based on radio datasets. Future work can validate our results by experimenting on actual wireless systems.

Another direction of future work can focus on investigating more sophisticated defenses against adversarial perturbations. As we have shown in this work, our PGM attack is robust against various defense mechanisms even the ones that are specific to the target wireless system's domain (i.e., ad hoc defenders). Nevertheless, a defense mechanism specifically designed for universal adversarial perturbations may offer greater success in defending against the presented attacks.

11 CONCLUSION

In this paper, we propose an adversarial attack using a perturbation generator model (PGM) against DNN-based wireless communication systems. In the absence of countermeasures deployed by the communicating parties, our attacks perform slightly better than existing adversarial attacks. More importantly, against communicating parties employing significant defenses, our techniques are robust and show significant gains over previous approaches. We also show that our attack is effective in a black-box scenario where the attacker generates the adversarial perturbations using a substitute DNN wireless model and uses the perturbation to attack the original DNN wireless model. We use remapping and regularizer functions to enforce an undetectability constraint for the perturbations, which makes the perturbations indistinguishable from random Gaussian noise. Furthermore, we use a regularizer function to enforce a distance constraint to degrade the performance of an ad hoc defender who tries to obtain an estimation of the perturbations using pilot signals. Our work shows that even in the presence of substantial defense mechanisms deployed by communication parties, DNN-based wireless systems are highly vulnerable to adversarial attacks. Hence, whereas there has been significant enthusiasm about such DNN-based approaches, our work suggests that there are also significant challenges to obtaining robust performance with such schemes.

ACKNOWLEDGEMENTS

We would like to thank our shepherd Yuan Tian and anonymous reviewers for their comments. The work was supported by the NSF CAREER grant CNS-1553301, NSF grant ECCS-2029323, and by DARPA and NIWC under contract N66001-15-C-4067. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. Milad Nasr was supported by a Google PhD Fellowship in Security and Privacy.

REFERENCES

[1] Abdullatif Albaseer, Bekir Sait Ciftler, and Mohamed M Abdallah. 2020. Performance Evaluation of Physical Attacks against E2E Autoencoder over Rayleigh Fading Channel. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*.

[2] Samuel Bair, Matthew DelVecchio, Bryse Flowers, Alan J Michaels, and William C Headley. 2019. On the limitations of targeted adversarial evasion attacks against deep learning enabled modulation recognition. In *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*.

[3] Federico Boccardi, Robert W Heath, Angel Lozano, Thomas L Marzetta, and Petar Popovski. 2014. Five disruptive technology directions for 5G. *IEEE communications magazine* (2014).

[4] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. 2019. On evaluating adversarial robustness. In *arXiv preprint arXiv:1902.06705*.

[5] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*.

[6] Linglong Dai, Ruicheng Jiao, Fumiyuki Adachi, H Vincent Poor, and Lajos Hanzo. 2020. Deep learning for wireless communications: An emerging interdisciplinary paradigm. *IEEE Wireless Communications* (2020).

[7] Matthew DelVecchio, Vanessa Arndorfer, and William C Headley. 2020. Investigating a Spectral Deception Loss Metric for Training Machine Learning-based Evasion Attacks. *arXiv preprint arXiv:2005.13124* (2020).

[8] Matthew DelVecchio, Bryse Flowers, and William C Headley. 2020. Effects of Forward Error Correction on Communications Aware Evasion Attacks. *arXiv preprint arXiv:2005.13123* (2020).

[9] Ali Fatih Demir, Mohamed Elkourdi, Mostafa Ibrahim, and Huseyin Arslan. 2019. Waveform design for 5G and beyond. *arXiv preprint arXiv:1902.05999* (2019).

[10] O. Dobre, A. Abdi, Y. Bar-Ness, and Wei Su. 2007. Survey of automatic modulation classification techniques: classical approaches and new trends. (2007).

[11] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. 2018. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[12] Bryse Flowers, R Michael Buehrer, and William C Headley. 2019. Communications aware adversarial residual networks for over the air evasion attacks. In *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*.

[13] Bryse Flowers, R Michael Buehrer, and William C Headley. 2019. Evaluating adversarial evasion attacks in the context of wireless communications. *IEEE Transactions on Information Forensics and Security* (2019).

[14] Jean-Yves Franceschi, Alhussein Fawzi, and Omar Fawzi. 2018. Robustness of classifiers to uniform l_p and Gaussian noise. In *International Conference on Artificial Intelligence and Statistics*.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*.

[17] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[18] Tobias Gruber, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. 2017. On deep learning-based channel decoding. In *2017 51st Annual Conference on Information Sciences and Systems (CISS)*.

[19] Muhammad Zaid Hameed, Andras Gyorgy, and Deniz Gunduz. 2019. Communication without interception: Defense against deep-learning-based modulation detection. *arXiv preprint arXiv:1902.10674* (2019).

[20] Taewon Hwang, Chenyang Yang, Gang Wu, Shaoqian Li, and Geoffrey Ye Li. 2008. OFDM and its wireless applications: A survey. *IEEE transactions on Vehicular Technology* (2008).

[21] Yihan Jiang, Hyeji Kim, Himanshu Asnani, Sreeram Kannan, Sewoong Oh, and Pramod Viswanath. 2019. Turbo autoencoder: Deep learning based channel codes for point-to-point communication channels. In *Advances in Neural Information Processing Systems*.

[22] Evgeny Khorov, Anton Kiryanov, Andrey Lyakhov, and Giuseppe Bianchi. 2018. A tutorial on IEEE 802.11 ax high efficiency WLANs. *IEEE Communications Surveys & Tutorials* (2018).

[23] Brian Kim, Yalin E Sagduyu, Kemal Davaslioglu, Tugba Erpek, and Sennur Ulukus. 2020. Channel-aware adversarial attacks against deep learning-based wireless signal classifiers. *arXiv preprint arXiv:2005.05321* (2020).

[24] Brian Kim, Yalin E Sagduyu, Kemal Davaslioglu, Tugba Erpek, and Sennur Ulukus. 2020. Over-the-Air Adversarial Attacks on Deep Learning Based Modulation Classifier over Wireless Channels. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*.

[25] Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (2014).

[26] Silvija Kokalj-Filipovic, Rob Miller, and Garrett Vanhoy. 2019. Adversarial examples in RF deep learning: Detection and physical robustness. In *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*.

[27] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236* (2016).

[28] YLJC Li, Leonard J Cimini, and Nelson R Sollenberger. 1998. Robust channel estimation for OFDM systems with rapid dispersive fading channels. *IEEE Transactions on communications* (1998).

[29] Fei Liang, Cong Shen, and Feng Wu. 2018. An iterative BP-CNN architecture for channel decoding. *IEEE Journal of Selected Topics in Signal Processing* (2018).

[30] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[31] Fan Meng, Peng Chen, Lenan Wu, and Xianbin Wang. 2018. Automatic modulation classification: A deep learning enabled approach. *IEEE Transactions on Vehicular Technology* (2018).

[32] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[33] Eliya Nachmani, Yair Be'ery, and David Burshtein. 2016. Learning to decode linear codes using deep learning. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*.

[34] Eliya Nachmani, Elad Marciano, David Burshtein, and Yair Be'ery. 2017. RNN decoding of linear block codes. *arXiv preprint arXiv:1702.07560* (2017).

[35] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. 2021. Defeating DNN-Based Traffic Analysis Systems in Real-Time With Blind Adversarial Perturbations. In *30th USENIX Security Symposium (USENIX Security 21)*.

[36] Timothy J O'shea and Nathan West. 2016. Radio machine learning dataset generation with gnu radio. In *Proceedings of the GNU Radio Conference*.

[37] Timothy O'Shea and Jakob Hoydis. 2017. An introduction to deep learning for the physical layer. *IEEE Transactions on Cognitive Communications and Networking* (2017).

[38] Timothy J O'Shea, Johnathan Corgan, and T Charles Clancy. 2016. Convolutional radio modulation recognition networks. In *International conference on engineering applications of neural networks*.

[39] Timothy James O'Shea, Tamoghna Roy, and T Charles Clancy. 2018. Over-the-air deep learning based radio signal classification. *IEEE Journal of Selected Topics in Signal Processing* (2018).

[40] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277* (2016).

[41] Sharan Ramjee, Shengtai Ju, Diyu Yang, Xiaoyu Liu, Aly El Gamal, and Yonina C Eldar. 2019. Fast deep learning for automatic modulation classification. *arXiv preprint arXiv:1901.05850* (2019).

[42] Francesco Restuccia, Salvatore D'Oro, Amani Al-Shawabka, Bruno Costa Rendon, Kaushik Chowdhury, Stratis Ioannidis, and Tommaso Melodia. 2020. Generalized wireless adversarial deep learning. In *Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning*.

[43] Meysam Sadeghi and Erik G Larsson. 2018. Adversarial attacks on deep-learning based radio signal classification. *IEEE Wireless Communications Letters* (2018).

[44] Meysam Sadeghi and Erik G Larsson. 2019. Physical adversarial attacks against end-to-end autoencoder communication systems. *IEEE Communications Letters* (2019).

[45] Yi Shi, Kemal Davaslioglu, and Yalin E Sagduyu. 2020. Generative Adversarial Network in the Air: Deep Adversarial Learning for Wireless Signal Spoofing. *IEEE Transactions on Cognitive Communications and Networking* (2020).

[46] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204* (2017).

[47] Muhammad Usama, Muhammad Asim, Junaid Qadir, Ala Al-Fuqaha, and Muhammad Ali Imran. 2019. Adversarial Machine Learning Attack on Modulation Classification. In *2019 UK/China Emerging Technologies (UCET)*.

[48] Nathan E West and Tim O'Shea. 2017. Deep architectures for modulation recognition. In *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*.

[49] Hao Ye, Geoffrey Ye Li, and Biing-Hwang Juang. 2017. Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wireless Communications Letters* (2017).

[50] Zhongyuan Zhao, Mehmet C Vuran, Fujuan Guo, and Stephen Scott. 2018. Deepwaveform: A learned OFDM receiver based on deep complex convolutional networks. *arXiv preprint arXiv:1810.07181* (2018).

A MODELS PARAMETERS

Table 4 illustrates the input size and parameters of each target wireless DNN model. Table 5 shows the structure and parameters of the substitute models in the black-box setting.

Table 4: Details of the target models for each wireless application

	Autoencoder Encoder	Autoencoder Decoder	Modulation Recognition	OFDM Channel Estimation
input size	16	2×7	$1 \times 1 \times 256$	256
hidden layers sizes	16	16	10560, 256	500, 250, 120
hidden layers activations	eLU	ReLU	Leaky ReLU, Softmax	ReLU, ReLU, ReLU
kernels	_	_	256, 80	_
kernel size	_	_	$(1, 3)$, $(2, 3)$	_
stride	_	_	$(1, 1)$, $(1, 1)$	_
padding	_	_	$(0, 2)$, $(0, 2)$	_
convolutional activations	_	_	Leaky ReLU, Leaky ReLU	_
output size	2×7	16	11	16

Table 5: Details of the substitute models in the black-box setting for each wireless application

	Autoencoder Encoder	Autoencoder Decoder	Modulation Recognition	OFDM Channel Estimation
input size	16	$1 \times 2 \times 7$	256	$1 \times 4 \times 64$
hidden layers sizes	16, 272	112, 32	1024, 1024, 512, 128	3456, 500, 250, 120
hidden layers activations	eLU, _	ReLU, _	ReLU, ReLU, ReLU, ReLU, Softmax	ReLU, ReLU, ReLU, Sigmoid
kernels	16	16, 8	_	32, 64
kernel size	6	$(2, 3)$, $(2, 3)$	_	$(2, 8)$, $(2, 8)$
stride	1	$(1, 1)$, $(1, 1)$	_	$(2, 1)$, $(1, 1)$
padding	3	$(1, 1)$, $(1, 0)$	_	$(0, 1)$, $(0, 1)$
convolutional activations	ReLU	ReLU, ReLU	_	ReLU, ReLU
output size	2×7	16	11	16