ABSTRACT

Title of dissertation: ADVERSARIAL MACHINE LEARNING IN WIRELESS COMMUNICATION SYSTEMS

Brian Kim, Doctor of Philosophy, 2022

Dissertation directed by: Professor Sennur Ulukus
Department of Electrical and Computer Engineering

We consider adversarial machine learning settings in wireless communication systems with adversaries that attempt to manipulate the deep learning (DL)-based wireless communication tasks, such as modulation classification and signal classification. In particular, we consider the evasion attack, i.e., adversarial attack, to which deep neural networks (DNNs) are known to be highly susceptible even under small-scale attacks. The shared and broadcast nature of wireless medium increases the potential for adversaries to tamper with DL-based wireless communication tasks. In this dissertation, we study the vulnerability of the DNNs used for various wireless communication applications to adversarial attacks.

First, we present channel-aware adversarial attacks against DL-based wireless signal classifiers where a DNN is used at each receiver to classify over-the-air received signals to modulation types. We propose realistic attacks by considering channel effects from the adversary to each receiver, and a broadcast adversarial attack by crafting a common adversarial perturbation to simultaneously fool classifiers

at different receivers. To mitigate the effect of the adversarial attack, we develop a certified defense scheme to guarantee the robustness of the classifier.

Next, we consider an adversary that transmits adversarial perturbations using its multiple antennas to fool the classifier into misclassifying the received signals. From the adversarial machine learning perspective, we show how to utilize multiple antennas at the adversary to improve the adversarial attack performance. We consider power allocation among antennas and utilization of channel diversity while exploiting the multiple antennas at the adversary. We show that attack success increases as the number of antennas at the adversary increases.

Then, we consider the privacy of wireless communications from an eavesdropper that employs a DL classifier to detect transmissions. In this setting, a transmitter transmits to its receiver in the presence of an eavesdropper, where a cooperative jammer (CJ) with multiple antennas transmits carefully crafted adversarial perturbations over-the-air to fool the eavesdropper into classifying the received superposition of signals as noise. We show that this adversarial perturbation causes the eavesdropper to misclassify the received signals as noise with a high probability while increasing the bit error rate (BER) at the legitimate receiver only slightly.

Next, we consider an adversary that generates adversarial perturbation using a surrogate DNN model that is trained at the adversary. This surrogate model may differ from the transmitter's classifier significantly because the adversary and the transmitter experience different channels from the background emitter and therefore their classifiers are trained with different distributions of inputs. We consider different topologies to investigate how different surrogate models that are trained by the adversary (depending on the differences in channel effects experienced by the adversary) affect the performance of the adversarial attack.

Then, we consider beam prediction problem using DNN for initial access (IA) in 5G and beyond communication systems where the user equipments (UEs) select the beam with the highest received signal strength (RSS) to establish their initial connection. We propose an adversarial attack to manipulate the over-the-air captured RSSs as the input to the DNN. This attack reduces the IA performance significantly and fools the DNN into choosing the beams with small RSSs.

Next, we consider adversarial attacks on power allocation where the base station (BS) allocates its transmit power to multiple orthogonal subcarriers by using a DNN to serve multiple UEs. The DNN corresponds to a regression model which is trained with channel gains as the input and allocated transmit powers as the output. While the BS allocates the transmit power to the UEs to maximize rates for all UEs, an adversary aims to minimize these rates. We show that the regression-based DNN is susceptible to adversarial attacks, where the rate of communication is significantly affected.

Finally, we consider reconfigurable intelligent surface (RIS)-aided wireless communication systems that improve the spectral efficiency and the coverage of wireless systems by electronically controlling the electromagnetic material in the presence of an eavesdropper. While there is an ongoing transmission boosted by the RIS, both the intended receiver and an eavesdropper individually aim to detect this transmission using their own DNN classifiers. The RIS interaction vector is designed by balancing two potentially conflicting objectives of focusing the transmitted signal to the receiver and keeping the transmitted signal away from the eavesdropper. To boost covert communications, adversarial perturbations are added to signals at the transmitter to fool the eavesdropper's classifier while keeping the effect on the receiver low. We show that adversarial perturbation and RIS interaction vector can be jointly designed to effectively increase the signal detection accuracy at the receiver while reducing the detection accuracy at the eavesdropper to enable covert communications.

Adversarial Machine Learning in Wireless Communication Systems

by

Brian Kim

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2022

Advisory Committee:
Professor Sennur Ulukus, Chair/Advisor
Professor Behtash Babadi
Professor Gang Qu
Professor Manoj Franklin
Professor Ashok Agrawala

Dedication

To my friends, family, and parents for always believing in me.

Acknowledgments

I would like to express my deepest and sincere gratitude to my advisor Professor Sennur Ulukus for her continuous support and guidance during my Ph.D. studies. Her relentless dedication to research and invaluable insights have always inspired me through all the time of Ph.D. studies. I would like to thank her for giving me the opportunity to work with her and would like to tell her that I am proud of, and grateful for, being her academic son. My work in this dissertation would not have been possible without her endless encouragement and relentless help.

I would like to extend my sincere thanks to Professors Behtash Babadi, Gang Qu, Manoj Franklin, and Ashok Agrawala for being on my dissertation committee and offering their valuable feedback. Also, I am thankful to all the professors that I interacted with at the University of Maryland.

I am grateful to Professor Yalin Sagduyu, of Virginia Tech, Professor Tugba Erpek, of Virginia Tech, Professor Yi Shi, of Virginia Tech, and Dr. Kemal Davaslioglu for our collaborations and their insightful advice on my work. I am also thankful to Professor Joonhyuk Kang, of KAIST, for the opportunity to work with him during my master's studies and for the support that I have received for my doctoral applications.

I am thankful to all my lab mates at the University of Maryland for making AVW a nice place to share and discuss ideas. Many thanks to my lab mates Batuhan Arasli, Priyanka Kaswan, Matin Mortaheb, Cemil Vahapoglu, Sajani Vithana, Zhusheng Wang, Purbesh Mitra, Mustafa Doger, Shreya Meel, Subhankar Banerjee, Dr. Bat-

uralp Buyukates, Dr. Melih Bastopcu, Dr. Yi-Peng Wei, Dr. Karim Banawan, Dr. Abdulrahman Baknina, Dr. Ahmed Arafa.

I would like to personally thank my dearest friends at the University of Maryland. I thank Chen Bai, Bibhusa Rawal, Semih Kara, Dr. Kyungjun Lee, Dr. Yongwan Park, Meenwook Ha, Yaesop Lee, and Yeji Park. It has been an absolute pleasure to spend time with them in College Park.

Many thanks to my other friends for making this journey enjoyable. I thank all the members of the tennis club and the soccer club for taking care of my mental and physical health.

I am forever indebted to my family for their unconditional support and care not just during my Ph.D. studies but in life in general. My deepest gratitude goes to my parents and my sister for always believing in me. This dissertation is dedicated to all of them.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Advances in *deep learning* (DL) based on *deep neural networks* (DNNs) have supported numerous applications to learn from complex data domains such as in computer vision and speech recognition [1]. Following the success of these applications, DL has been applied to wireless communications, where channel, interference, and traffic effects jointly contribute to the high complexity of the spectrum data [2–14].

As adversaries can manipulate training and testing time input of machine learning (ML) algorithms, *adversarial ML (AML)* has emerged to study the operation of ML models in the presence of adversaries and support safe adoption of ML to the emerging applications [15]. In particular, DNNs are known to be highly susceptible to even small-scale adversarial attacks, as extensively demonstrated in the computer vision domain [16–19].

Recently, AML has been studied in wireless communication systems using DNNs [20–34]. The shared and broadcast nature of wireless medium increases the potential for adversaries to tamper with DL-based wireless communication tasks over-the-air. The attack and defense mechanisms from other data domains such as computer vision are not feasible in wireless communications due to its unique characteristics. First, a wireless adversary cannot directly manipulate input data to a DL algorithm running at a separate target and its manipulation needs to reach the target over-the-air through channel effects. Second, a wireless adversary can attack multiple targets (each with a different channel) simultaneously with a single (omnidirectional) transmission due to the broadcast nature of wireless communications. Third, a wireless adversary can use multiple antennas to send multiple attacks simultaneously to the target whereas adversarial attacks are limited to a single perturbation in computer vision applications. Finally, unlike the computer vision domain where the main goals are related to image, DL applications used in wireless communications have various goals such as modulation classification, signal classification, channel estimation, and power allocation. Thus, a thorough analysis of adversarial attacks on these applications is required.

Different types of wireless attacks have been developed with AML [35, 36]. *Exploratory (inference) attacks* have been considered in [37–39], where an adversary builds a DNN to learn the transmission pattern in the channel and jam transmissions that would be otherwise successful. Over-the-air spectrum *poisoning (causative) attacks* have been considered in [40–42], where an adversary poisons (falsifies) a transmitter's spectrum sensing data over-the-air by transmitting during the short spectrum sensing period of the transmitter. *Trojan attacks* have been studied in [43] against a signal classifier, where an adversary slightly manipulates training data by inserting Trojans in terms of modifying the phases and the labels of only few training data to a target label, and then transmits signals with the same phase shift in the inference time to fool the signal classifier. *Membership inference attacks* have been considered in [44, 45] to learn whether a given data sample has been used during training and thereby gaining information on waveform, channel, and radio hardware characteristics. AML has been studied in [46, 47] to launch *spoofing attacks* that aim to fool signal authentication systems based on DNNs. As a viable threat for emerging wireless systems, AML has been also studied to launch attacks against 5G and beyond communications [48–50].

Built upon AML, *adversarial attacks* (a.k.a *evasion attacks*) correspond to small modifications of the original input to the DNNs that make DL algorithm to misclassify the input. These small modifications are not just random noise but carefully designed in a way that changes the decision of the DL algorithm. As an evasion attack, [51] has shown that the end-to-end autoencoder communication systems, proposed in [52], are vulnerable to adversarial attacks in an additive white Gaussian noise (AWGN) channel environment, where the attack increases the block-error-rate at the receiver. Further, it has been shown that the modulation classifier used in [7] incurs major errors due to adversarial attacks in the AWGN channel. Similar evasion attacks and corresponding defense mechanisms have been studied in [20–27, 29, 30, 53, 54].

Common to all these works is the fact that they study the applications of machine learning in wireless communications where the adversary crafts adversarial perturbation without considering any channel effects. Due to the inherent nature of the wireless medium, it is critical to consider the channel effects to reach the target over-the-air. This motivates to study the performance of adversarial attacks taking channel effects into account.

In this dissertation, our goal is to analyze the vulnerability of machine learning applications in wireless communications by creating realistic adversarial perturbations. Since there exist various machine learning applications in wireless communications, we investigate the vulnerability of machine learning applications that are used in wireless communications for different purposes. Another goal of this dissertation is to study how these adversarial perturbations are used against the adversary that uses machine learning applications to make wireless communications covert. In the remainder of this introduction, we summarize the chapters of this dissertation.

In Chapter 2, we present realistic wireless attacks built upon AML and corresponding defense by accounting for channel and broadcast transmission effects in algorithm design for adversarial attacks. We show the need for a channel-aware adversarial attack that can simultaneously work against multiple receivers by showing that (a) the design of adversarial perturbations without taking realistic channel effects into account cannot fool a modulation classifier, and (b) an adversarial attack crafted for the channel to a particular receiver is not effective against another receiver with different channel characteristics (i.e., the attack is receiver specific and cannot be used for a broadcast attack launched simultaneously against multiple receivers). By considering a DNN at each receiver to classify wireless signals to modulation types, we design realistic adversarial attacks in the presence of channel effects (from the adversary to the receiver) and multiple classifiers at different receivers. For that purpose, we start with a single receiver and determine channel-aware adversarial perturbations to reduce the accuracy of detecting the modulation type at the receiver. We first propose two white-box attacks, a *targeted attack* with minimum power and a *non-targeted attack*, subject to channel effects known by the adversary. We show that the adversarial attack fails to fool the classifier using limited power if the channel between the adversary and the receiver is not considered when designing the adversarial perturbation. Then, we present algorithms to design adversarial perturbations by accounting for known channel effects and show that the classifier accuracy can be significantly reduced by channel-aware adversarial attacks. By relaxing the assumption of exact channel information at the adversary, we present a white-box adversarial attack with *limited channel information* available at the adversary. We further relax assumptions regarding the information availability of transmitter input and classifier model, and introduce a *black-box universal adversarial perturbation (UAP) attack* for an adversary with limited information. We also introduce a *broadcast* adversarial attack to fool classifiers at different receivers with a single perturbation transmission by leveraging the broadcast property of wireless communications.

In Chapter 3, building on Chapter 2, we consider multiple antennas at the adversary to generate multiple concurrent perturbations over different channel effects to the input of a DNN based modulation classifier at a wireless receiver. This problem setting is different from computer vision applications of adversarial attacks that are limited to a single perturbation that can be directly added to the DNN's input without facing uncertainties such as channel effects. We assume that the adversary has multiple antennas to transmit adversarial perturbations in the presence of realistic channel effects and aims to decrease the accuracy of a modulation classifier.

First, we show that just increasing the number of individual adversaries with single antennas (located at different positions) does not improve the attack performance. Next, we consider the use of multiple antennas at a single adversary and propose different methods to allocate power among antennas at the adversary and to exploit the channel diversity. We propose a genie aided adversarial attack where the adversary selects one antenna to transmit the perturbation such that it would result in the worst classification performance depending on the channel condition over the entire symbol block that corresponds to the input to the DNN at the receiver. Then, we consider transmitting with all the antennas at the adversary where the power allocation is based on the channel gains, either proportional or inversely proportional to the channel gains. Finally, we propose the elementwise maximum channel gain (EMCG) attack to utilize the channel diversity more efficiently by selecting the antenna with the best channel gain at the symbol level to transmit perturbations.

In Chapter 4, we study covert communications from an *adversarial machine learning* point of view. We consider an eavesdropper with a DL-based classifier to detect an ongoing transmission where this classifier achieves high accuracy for distinguishing the received signals from noise. We introduce a *cooperative jammer* (*CJ*) that has been extensively used in the physical layer security literature [55–57]. In this chapter, the CJ transmits signals over-the-air at the same time as the transmitter with the purpose of fooling the eavesdropper's classifier for covert communications. While a perturbation with a high power level transmitted by the CJ can easily fool the classifier, it would also increase the interference and the bit error rate (BER) at the intended receiver to an unacceptable level. Therefore, an upper bound on

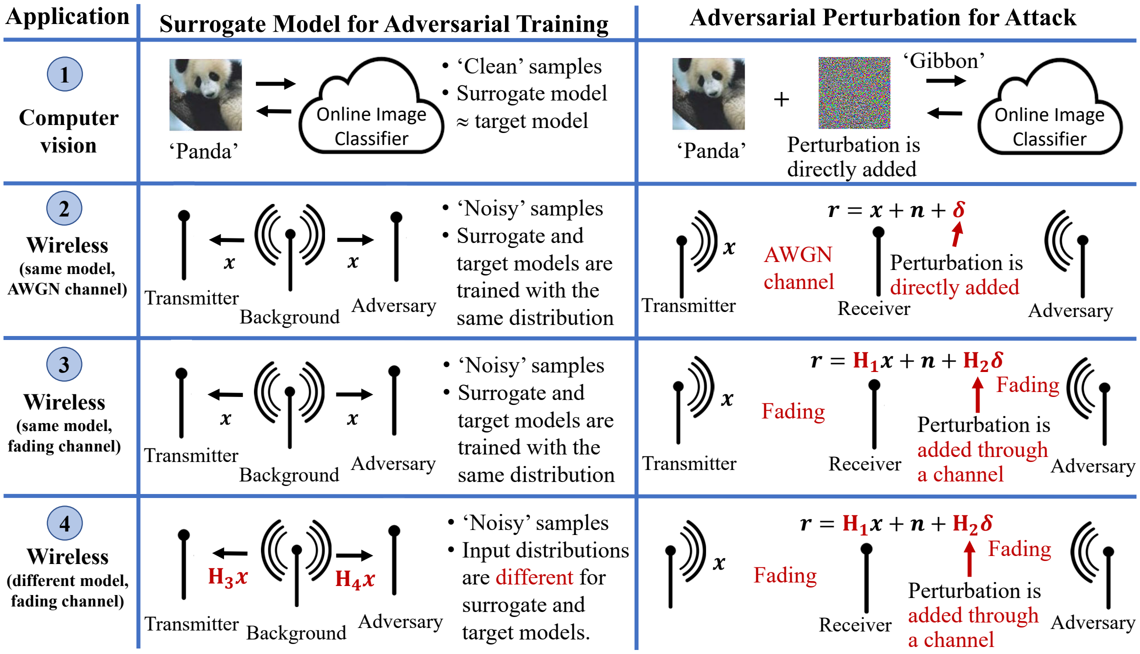

Application
Surrogate Model for Adversarial Training
Adversarial Perturbation for Attack
1
Computer vision
'Panda'
Online Image Classifier
• 'Clean' samples
• Surrogate model ≈ target model
'Panda'
+
Perturbation is directly added
'Gibbon'
Online Image Classifier
2
Wireless (same model, AWGN channel)
Transmitter
Background
Adversary
x
x
• 'Noisy' samples
• Surrogate and target models are trained with the same distribution
Transmitter
x
$r = x + n + \delta$
AWGN channel
Perturbation is directly added
Receiver
Adversary
3
Wireless (same model, fading channel)
Transmitter
Background
Adversary
x
x
• 'Noisy' samples
• Surrogate and target models are trained with the same distribution
Transmitter
x
$r = H_1x + n + H_2\delta$
Fading
Fading
Perturbation is added through a channel
Receiver
Adversary
4
Wireless (different model, fading channel)
Transmitter
Background
Adversary
H_3x
H_4x
• 'Noisy' samples
• Input distributions are different for surrogate and target models.
x
$r = H_1x + n + H_2\delta$
Fading
Fading
Perturbation is added through a channel
Receiver
Adversary

Figure 1.1: Adversarial attacks from computer vision to wireless applications.

the perturbation strength is imposed. Further, we extend the analysis to the use of *multiple antennas* at the CJ to generate multiple concurrent perturbations over different channel effects, as in Chapter 3, for better covert communications. We assume that the CJ has multiple antennas to transmit adversarial perturbations against the eavesdropper and aims to decrease the probability of detection at the eavesdropper.

In Chapter 5, we investigate the channel effects on the surrogate model that is built by the adversary through over-the-air spectrum observations and used to craft adversarial attacks against a DNN wireless signal classifier. We consider a wireless communications system with a background emitter, a transmitter, and an adversary where the transmitter collects I/Q data and uses its DNN classifier to detect the ongoing transmission of a background signal source for channel access decisions. On the other hand, the adversary builds a surrogate model of the DNN classifier used at the transmitter. For that purpose, the adversary collects the I/Q data, namely signals received from the background signal source, by exploiting the broadcast nature of transmissions and obtains the labels by listening to the potential signals from the transmitter. Using the surrogate model, the adversary generates adversarial perturbations to fool the classifier at the transmitter. Different topologies of the adversary are considered to investigate how the difference in the distribution of the training datasets affects the performance of the adversarial attack on the transmitter. Also, different approaches to select the transmit power at the adversary node using the surrogate model are considered. Finally, we relax the assumption that the adversary knows the exact input at the transmitter when determining the adversarial perturbation. Fig. 1.1 summarizes the difference of adversarial attacks and surrogate models in different scenarios including computer vision and wireless communications.

In Chapter 6, we show that DNNs that are used for millimeter wave (mmWave) beam prediction as part of the initial access (IA) process in 5G and beyond communications are susceptible to adversarial attacks. In this DL-based approach, the UE predicts the best beam from a large set of narrow beams by using only the RSSs for a subset of these possible beams. We introduce two different attack schemes, namely, non-targeted attack and k-worst beam attack. For non-targeted attack, we use fast gradient method (FGM) [58] to cause any misclassification at the DNN classifier at the receiver which is independent of the wrong labels for beams. Then, we propose k-worst beam attack that not only causes a misclassification but also enforces the DNN classifier to select one of the k worst beams as a false output to further reduce the IA performance. During the simulations, we compare the non-targeted FGM attack and k-worst beam attack with benchmark attacks with Gaussian and uniform noise added across RSSs from input beams. We show that the beam prediction of the IA process is highly vulnerable to adversarial attacks that can significantly reduce the beam prediction accuracy.

In Chapter 7, we consider a regression-based DNN at the BS for the power allocation problem to serve multiple users. To investigate the vulnerability of the regression-based DNN, we design an adversarial attack to change the DNN's input to reduce the minimum rate over all UEs subject to the condition that the perturbations to the inputs of the DNN are bounded. Specifically, the adversary crafts the adversarial perturbations to decrease the minimum rate by generating the perturbations to the DNN input based on the gradient of the minimum rate. We introduce two approaches to compute the gradient of the rate with respect to the inputs to the DNN to craft the perturbation. First, the adversary obtains the DNN power allocation outputs by computing the minimum rate based on these outputs of DNN using analytical means and then computes the gradient of the rate with respect to the changes to the DNN input. Second, the adversary aims to attack the DNN by calculating the gradient of the DNN's loss function using the FGM, where we define the DNN's loss function as the error with respect to the minimum rate.

In Chapter 8, we consider reconfigurable intelligent surface (RIS)-aided wireless communications, where a receiver uses its DNN classifier to detect the transmitter's signal that is reflected by the RIS. Concurrently, there exists an eavesdropper with another DNN classifier to identify an ongoing transmission for adversarial purposes. The transmitter adds adversarial attacks to its signals to fool the eavesdropper and reduce its detection accuracy. Minimum power is used for these adversarial perturbations to minimize the effect on the receiver's detection performance. Simultaneously, the RIS interaction vector is designed so that the RIS reflects the signal towards the intended receiver while keeping the reflection away from the eavesdropper. We consider different topologies and analyze how the design of the RIS interaction vector for covert communications adapts to different locations of the receiver and the eavesdropper. Our results show that the beam selection of the RIS is the crucial component for covert communications when the transmitter has a low power budget for the adversarial attack.

In Chapter 9, we present conclusions of this dissertation.

CHAPTER 2

Channel-Aware Adversarial Attacks Against Deep Learning-Based Wireless Signal Classifiers

2.1 Introduction

In this chapter, we present realistic wireless attacks that take channel effects into account while designing adversarial attacks. In [59], adversarial attacks have been studied for modulation classification of wireless signals, where fast gradient method (FGM) [58] is used to generate adversarial attacks. Specifically, targeted FGM attack has been used by enforcing the DNNs to misclassify the input signals to a target label. It has been shown that the modulation classifier used in [7] incurs major errors due to adversarial attacks in the AWGN channel. However, even a small channel effect would significantly reduce the impact of adversarial attacks by reducing the received perturbation power just below the necessary level such that the adversarial attack fails in changing classification decisions over-the-air. Motivated by these works, we study the vulnerability of the DNNs when adversarial attacks are crafted by considering channel effects.

By considering a DNN at each receiver to classify the modulation type of the wireless signals, we design adversarial attacks in the presence of channel effects from the adversary to each receiver and multiple classifiers at different receivers. We propose two white-box attacks, a *targeted attack* with minimum power and a *non-targeted attack*, that takes the channel effect into account while generating the attacks.

Then, we relax the assumption of knowing the exact channel information at the adversary and present a white-box adversarial attack with *limited channel information* available at the adversary. We apply principal component analysis (PCA) to generate adversarial attacks using a lower-dimensional representation of channel distribution. In addition to PCA, we also apply variational autoencoder (VAE) to capture the complexity of underlying data characteristics regarding the transmitter input and the channel. We further relax assumptions regarding the information availability of transmitter input and classifier model, and introduce a *black-box universal adversarial perturbation (UAP) attack* for an adversary with limited information.

We also introduce a *broadcast* adversarial attack to manipulate multiple classifiers at different receivers with a single perturbation transmission by using the broadcast nature of wireless communications. A practical scenario for this attack is a user authentication system with multiple signal sensors deployed at different locations. First, we show that the channel-aware adversarial perturbation is inherently selective in the sense that it can fool a target classifier at one receiver (whose channel is used in the attack design) but cannot fool a classifier at another receiver due to different channels experienced at different receivers. By considering channels from the adversary to all receivers jointly, we design the broadcast adversarial perturbation that can simultaneously fool multiple classifiers at different receivers. Using different levels of perturbation budget available relative to noise power, our results illustrate the need to utilize the channel information in designing over-the-air adversarial attacks.

As a countermeasure, we present a *defense* method to reduce the impact of adversarial perturbations on the classifier performance. Following the *randomized smoothing* approach from [60–62] (previously applied in computer vision), the training data for modulation classifier is augmented with isotropic Gaussian noise to make the trained model robust to adversarial perturbations in test time. We show that this defense is effective in reducing the impact of adversarial attacks on the classifier performance. We further consider *certified defense* to guarantee classifier robustness of the classifier accuracy against adversarial attacks. The classifier is certified by augmenting the received signals with Gaussian noise samples during test time and checking statistical significance of classification results.

2.2 System Model

We consider a wireless communications system that consists of a transmitter, m receivers, and an adversary as shown in Fig. 2.1. All nodes are equipped with a single antenna and operate on the same channel. Each receiver classifies its received signals with a DNN to the modulation type that is used by the transmitter. In the meantime, the adversary transmits perturbation signals over the air to fool the

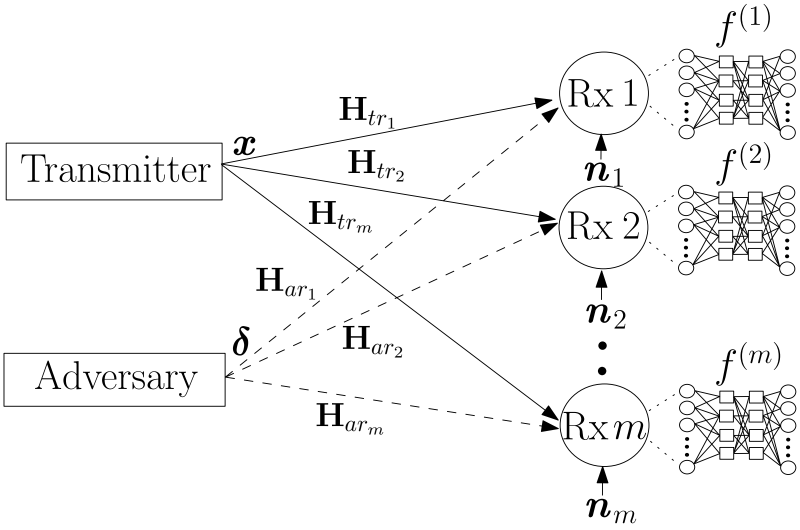


Figure 2.1: Wireless communication system with a transmitter, m receivers, and an adversary.

classifier at the receiver into making errors in modulation classification.

The DNN classifier at the ith receiver is denoted by $f^{(i)}(\cdot;\boldsymbol{\theta}_i) : \mathcal{X} \to \mathbb{R}^C$, where $\boldsymbol{\theta}_i$ is the parameters of the DNN at receiver i and C is the number of modulation types. Note $\mathcal{X} \subset \mathbb{C}^p$, where p is the dimension of the complex-valued (in-phase/quadrature) inputs can be also represented by concatenation of two real-valued inputs. The classifier $f^{(i)}$ assigns $\hat{l}^{(i)}(\boldsymbol{x}, \boldsymbol{\theta}_i) = \arg\max_k f_k^{(i)}(\boldsymbol{x}, \boldsymbol{\theta}_i)$, a modulation type, to every input $\boldsymbol{x} \in \mathcal{X}$ where $f_k^{(i)}(\boldsymbol{x}, \boldsymbol{\theta}_i)$ is the output of the ith classifier corresponding to the kth modulation type.

The channel from the transmitter to the ith receiver is denoted by $\boldsymbol{h}_{tr_i}$ and the channel from the adversary to the ith receiver is denoted by $\boldsymbol{h}_{ar_i}$, where $\boldsymbol{h}_{tr_i} = [h_{tr_i,1}, h_{tr_i,2}, \cdots, h_{tr_i,p}]^T \in \mathbb{C}^{p\times 1}$ and $\boldsymbol{h}_{ar_i} = [h_{ar_i,1}, h_{ar_i,2}, \cdots, h_{ar_i,p}]^T \in \mathbb{C}^{p\times 1}$. If the transmitter transmits $\boldsymbol{x}$, the ith receiver receives $\boldsymbol{r}_{tr_i} = \mathbf{H}_{tr_i}\boldsymbol{x} + \boldsymbol{n}_i$, if there is no adversarial attack, or receives $\boldsymbol{r}_{ar_i}(\boldsymbol{\delta}) = \mathbf{H}_{tr_i}\boldsymbol{x} + \mathbf{H}_{ar_i}\boldsymbol{\delta} + \boldsymbol{n}_i$, if the adversary launches an adversarial attack by transmitting the perturbation signal $\boldsymbol{\delta}$, where

$\mathbf{H}_{tr_i} = \text{diag}\{h_{tr_i,1}, \cdots, h_{tr_i,p}\} \in \mathbb{C}^{p\times p}, \mathbf{H}_{ar_i} = \text{diag}\{h_{ar_i,1}, \cdots, h_{ar_i,p}\} \in \mathbb{C}^{p\times p}$, $\boldsymbol{\delta} \in \mathbb{C}^{p\times 1}$ and $\boldsymbol{n}_i \in \mathbb{C}^{p\times 1}$ is complex Gaussian noise. We assume that the adversarial perturbation $\boldsymbol{\delta}$ is synchronized with the transmitter's signal. To make the attack stealthy (i.e., hard to detect) and energy efficient, the adversarial perturbation $\boldsymbol{\delta}$ is restricted as $\|\boldsymbol{\delta}\|_2^2 \leq P_{max}$ for some suitable power budget defined by P_{max}.

The adversary determines the (common) adversarial perturbation $\boldsymbol{\delta}$ for the input $\boldsymbol{x}$ and all of the classifiers $f^{(i)}$ for $i = 1, 2, \cdots, m$ by solving the optimization (2.1) problem:

$$\begin{aligned} \underset{\boldsymbol{\delta}}{\arg\min} \quad & \|\boldsymbol{\delta}\|_2 \\ \text{subject to} \quad & \hat{l}^{(i)}(\boldsymbol{r}_{tr_i}, \boldsymbol{\theta}_i) \neq \hat{l}^{(i)}(\boldsymbol{r}_{ar_i}(\boldsymbol{\delta}), \boldsymbol{\theta}_i) \quad i = 1, 2, \ldots, m \\ & \|\boldsymbol{\delta}\|_2^2 \leq P_{max}. \end{aligned} \tag{2.1}$$

In (2.1), the objective is to minimize the perturbation power subject to two constraints (a) each receiver misclassifies the received signal, and (b) the budget for total perturbation power is not exceeded. Note that the best solution is not necessarily realized at $\|\boldsymbol{\delta}\|_2^2 = P_{max}$ due to the complicated decision boundary of the DNN that depends on the power and phase of perturbation.

In practice, solving (2.1) is difficult because of the nonlinearity of the DNN. Thus, different methods have been proposed (primarily in the computer vision domain) to approximate the adversarial perturbation. For instance, FGM is a computationally efficient method for crafting adversarial attacks by linearizing the loss function of the DNN classifier. Let $L(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$ denote the loss function of the model (at any given receiver), where $\boldsymbol{y} \in \{0,1\}^C$ is the class vector. Then, FGM linearizes the loss function in a neighborhood of $\boldsymbol{x}$ and uses this linearized function for optimization.

We consider two types of attacks called *targeted attacks* and *non-targeted attacks* that involve different objective functions for the adversary to optimize. In a targeted attack, the adversary aims to generate a perturbation that causes the classifier at the receiver to misclassify the input to a target class (label), e.g., a QPSK modulated signal is classified as QAM16. In a non-targeted FGM attack, the adversary searches for an attack that causes any misclassification (independent of target class). More details on these two types of attacks are provided in Section 2.3.

Our goal in this chapter is to design adversarial perturbation attacks to fool potentially multiple classifiers while considering the channel effects. For the white-box adversarial attacks, we assume that for all i the adversary knows (a) the DNN architecture ($\boldsymbol{\theta}_i$ and $L^{(i)}(\cdot)$) of the classifier at the ith receiver, (b) the input at the ith receiver, and consequently (c) the channel $\boldsymbol{h}_{ar_i}$ between the adversary and the ith receiver. We will relax these assumptions in Sections 2.5 and 2.6.

We compare the performances of the attacks proposed in this work with the benchmark attack from [59] that adds the adversarial perturbation directly to the receiver signal without accounting for the channel effects from the adversary to a receiver beyond the AWGN channel. For performance evaluation, we use the VT-CNN2 classifier from [52] as the modulation classifier (also used in [59]), where the classifier consists of two convolution layers and two fully connected layers, and train it with GNU radio ML dataset RML2016.10a [63]. The dataset contains 220,000 samples. Each sample corresponds to a specific modulation scheme at a specific signal-to-noise ratio (SNR). There are eleven modulations in the dataset: BPSK, QPSK, 8PSK, QAM16, QAM64, CPFSK, GFSK, PAM4, WBFM, AM-SSB, and AM-DSB. We follow the same setup of [52], using Keras with TensorFlow backend, where the input sample to the modulation classifier is 128 I/Q (in-phase/quadrature) channel symbols. Half of the samples are used for training and the other half are used in test time.

2.3 Targeted White-Box Adversarial Attacks using Channel Information

We start with a single receiver, i.e., $m = 1$, and omit the index i of the ith receiver for simplicity in Sections 2.3-2.6. We will extend the setup to multiple receivers in Section 2.7. For the targeted attack, the adversary minimizes $L(\boldsymbol{\theta}, \boldsymbol{r}_{ar}, \boldsymbol{y}^{target})$ with respect to $\boldsymbol{\delta}$, where $\boldsymbol{y}^{target}$ is one-hot encoded desired target class (label). FGM is used to linearize the loss function as $L(\boldsymbol{\theta}, \boldsymbol{r}_{ar}, \boldsymbol{y}^{target}) \approx L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^{target}) + (\mathbf{H}_{ar}\boldsymbol{\delta})^T \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^{target})$ that is minimized by setting $\mathbf{H}_{ar}\boldsymbol{\delta} = -\alpha \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^{target})$, where α is a scaling factor to constrain the adversarial perturbation power to P_{max}.

The adversary can generate different targeted attacks with respect to different $\boldsymbol{y}^{target}$ that causes the classifier at the receiver to misclassify the received signals to $C - 1$ different modulation types. Thus, as in [59], the adversary can craft targeted attacks for all $C - 1$ modulation types and choose the target modulation that requires the least power. The case considered in [59] corresponds to $\mathbf{H}_{ar} = \mathbf{I}$ without realistic channel effects. We call the targeted attack perturbation in [59] as $\boldsymbol{\delta}^{NoCh}$, which is an optimal targeted attack without channel consideration (this corresponds to Algorithm 1 by setting $\mathbf{H}_{ar} = \mathbf{I}$). In the following subsections, we propose three targeted adversarial attacks to account for the effects of the channel.

2.3.1 Channel Inversion Attack

We first begin with a naive attack, where the adversary designs its attack by inverting the channel in the optimal targeted attack $\boldsymbol{\delta}^{NoCh}$, which is obtained using Algorithm 1 with $\mathbf{H}_{ar} = \mathbf{I}$. Since the adversarial attack goes through channel $\boldsymbol{h}_{ar}$, the jth element of the perturbation $\boldsymbol{\delta}$ is simply designed as $\delta_j = \frac{\delta_j^{NoCh}}{h_{ar,j}}$ such that after going through the channel it has the same direction as δ_j^{NoCh} for $j = 1, \cdots, p$. Furthermore, to satisfy the power constraint P_{max} at the adversary, a scaling factor α is introduced such that $\boldsymbol{\delta}^{div} = -\alpha\boldsymbol{\delta}$, where $\alpha = \frac{\sqrt{P_{max}}}{\|\boldsymbol{\delta}\|_2}$. Thus, the perturbation received at the receiver becomes $\mathbf{H}_{ar}\boldsymbol{\delta}^{div} = -\alpha\boldsymbol{\delta}^{NoCh}$.

2.3.2 Minimum Mean Squared Error (MMSE) Attack

In the MMSE attack, the adversary designs the perturbation $\boldsymbol{\delta}^{MMSE}$ so that the distance between the perturbation after going through the channel and the optimal targeted perturbation at the receiver (which corresponds to $\mathbf{H}_{ar} = \mathbf{I}$) is minimized. By designing the attack in this way, the received perturbation at the receiver is close to the optimal targeted attack as much as possible while satisfying the power constraint at the adversary. However, since the classifier is sensitive to not only the direction but also to the power of perturbation, the squared error criterion might penalize the candidates of $\boldsymbol{\delta}^{MMSE}$ that have more power with the direction of $\boldsymbol{\delta}^{NoCh}$, i.e., we set $\boldsymbol{\delta}^{MMSE} = \gamma\boldsymbol{\delta}^{NoCh}$ to search for all magnitudes of the $\boldsymbol{\delta}^{NoCh}$. Therefore, we formulate the optimization problem to select the perturbation $\boldsymbol{\delta}^{MMSE}$ as

$$\begin{aligned} \min_{\boldsymbol{\delta}^{MMSE}} \quad & \|\mathbf{H}_{ar}\boldsymbol{\delta}^{MMSE} - \gamma\boldsymbol{\delta}^{NoCh}\|_2^2 \\ \text{subject to} \quad & \|\boldsymbol{\delta}^{MMSE}\|_2^2 \leq P_{max}, \end{aligned} \tag{2.2}$$

where γ is optimized by line search. We can write (2.2) as

$$\begin{aligned} \min_{\delta_j^{MMSE}} \quad & \sum_{j=1}^{p} \|h_{ar,j}\delta_j^{MMSE} - \gamma\delta_j^{NoCh}\|_2^2 \\ \text{subject to} \quad & \sum_{j=1}^{p} \|\delta_j^{MMSE}\|_2^2 \leq P_{max}. \end{aligned} \tag{2.3}$$

We solve the convex optimization problem (2.3) by using the Lagrangian method. The Lagrangian for (2.3) is given by

$$\mathcal{L} = \sum_{j=1}^{p} \|h_{ar,j}\delta_j^{MMSE} - \gamma\delta_j^{NoCh}\|_2^2 + \lambda\left(\sum_{j=1}^{p} \|\delta_j^{MMSE}\|_2^2 - P_{max}\right), \tag{2.4}$$

where $\lambda \geq 0$. The Karush–Kuhn–Tucker (KKT) conditions are given by

$$h_{ar,j}^{*}(h_{ar,j}\delta_j^{MMSE} - \gamma\delta_j^{NoCh}) + \lambda\delta_j^{MMSE} = 0, \tag{2.5}$$

Algorithm 1: MRPP attack

```
Inputs: input $\boldsymbol{r}_{tr}$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$ and model of
  the classifier
Initialize: $\boldsymbol{\varepsilon} \leftarrow \mathbf{0}^{C\times 1}$
for class-index c in range(C) do
    $\varepsilon_{max} \leftarrow P_{max}, \varepsilon_{min} \leftarrow 0$
    $\boldsymbol{\delta}^c_{norm} = \frac{\mathbf{H}^*_{ar}\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{r}_{tr},\boldsymbol{y}^c)}{(\|\mathbf{H}^*_{ar}\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{r}_{tr},\boldsymbol{y}^c)\|_2)}$
    while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
        $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
        $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x} - \varepsilon_{avg}\mathbf{H}_{ar}\boldsymbol{\delta}^c_{norm}$
        if $\hat{l}(\boldsymbol{x}_{adv}) == l_{true}$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
        else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
    end
    $\varepsilon[c] = \varepsilon_{max}$
end
$target = \arg\min \boldsymbol{\varepsilon},\ \boldsymbol{\delta}^{MRPP} = -\sqrt{P_{max}}\boldsymbol{\delta}^{target}_{norm}$
```

for $j = 1, \cdots, p$. From the KKT conditions, we obtain the perturbation of the MMSE attack as

$$\delta_j^{MMSE} = -\frac{\gamma h^*_{ar,j}\delta_j^{NoCh}}{h^*_{ar,j}h_{ar,j} + \lambda}, \tag{2.6}$$

for $j = 1, \cdots, p$, where λ is determined by the power constraint at the adversary. Note that the received perturbation at the receiver is $\mathbf{H}_{ar}\boldsymbol{\delta}^{MMSE} = -\boldsymbol{\alpha}^T\boldsymbol{\delta}^{NoCh}$, where $\boldsymbol{\alpha} \in \mathbb{R}^{p\times 1}$ and the jth element of $\boldsymbol{\alpha}$ is $\alpha_j = \frac{\gamma h_{ar,j}h^*_{ar,j}}{h^*_{ar,j}h_{ar,j}+\lambda}$.

2.3.3 Maximum Received Perturbation Power (MRPP) Attack

In the MRPP attack, the adversary selects the perturbation $\boldsymbol{\delta}^{MRPP}$ to maximize the received perturbation power at the receiver and analyzes how the received perturbation power affects the decision process of the classifier. To maximize the received perturbation power and effectively fool the classifier into making a specific classification error, the adversary has to fully utilize the channel between the adversary and the receiver. Thus, if the targeted attack δ_j^{target} is multiplied by the conjugate of the channel, $h^*_{ar,j}$, to maximize the received perturbation power along the channel $h_{ar,j}$, then the received perturbation after going through the channel becomes $\|h_{ar,j}\|_2^2\delta_j^{target}$. In the MRPP attack, not only the direction of the adversarial perturbation is unaffected after going through the channel, but also the power of the adversarial perturbation is maximized by utilizing the channel. Finally, the adversary generates the targeted attack for every possible modulation type to decide the target class and calculate the scaling factor to satisfy the power constraint at the adversary. The details are presented in Algorithm 1.

2.3.4 Attack Performance

We assume that the channel between the adversary and the receiver is subject to Rayleigh fading with path-loss and shadowing, i.e., $h_{ar,j} = K(\frac{d_0}{d})^\gamma \psi h_{ray,j}$ where $K = 1, d_0 = 1, d = 10, \gamma = 2.7, \psi \sim$ Lognormal$(0, 8)$ and $h_{ray,j} \sim$ Rayleigh$(0, 1)$. We use the perturbation-to-noise ratio (PNR) metric from [59] that captures the relative perturbation with respect to the noise and measure how the increase in the PNR affects the accuracy of the classifier. Note that as the PNR increases, it is more likely for the attack to be detected by the receiver. In the performance evaluation figures, we denote the targeted attack by ‘TA’, the non-targeted attack by ‘NTA’ and the broadcast attack by ‘BC’.

Fig. 2.2 presents the classifier accuracy versus the PNR under the proposed

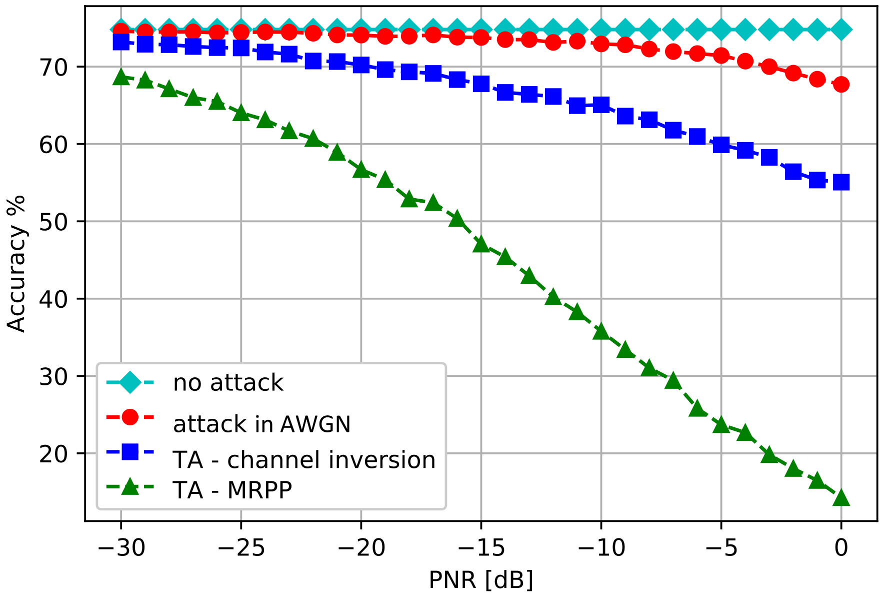


Figure 2.2: Classifier accuracy under adversarial attacks with and without considering wireless channel effects when SNR = 10 dB.

targeted white-box adversarial attack with exact channel information and compares them with the adversarial attack studied in [59] where the adversarial attack is generated in AWGN channel. As expected, the white-box attack in [59] without considering the necessary channel effects from the adversary to the receiver has poor performance that is close to no attack case in the low PNR region. The reason is that the wireless channel changes the phase and the magnitude of the perturbations perceived at the receiver. Further, the targeted channel inversion attack does not perform well compared to the targeted MRPP attack, indicating the importance of the received power of perturbations for the performance of the classifier at the receiver.

Algorithm 2: Naive non-targeted attack

```
Inputs: input $\boldsymbol{r}_{tr}$, number of iterations $E$, power constraint $P_{max}$, true class
  $\boldsymbol{y}^{true}$ and model of the classifier
Initialize: Sum of gradient $\Delta \leftarrow 0$ , $\boldsymbol{x} \leftarrow \boldsymbol{r}_{tr}$
for $e$ in range(E) do
    $\boldsymbol{\delta}_{norm} = \frac{\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{y}^{true})}{(\|\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{y}^{true})\|_2)}$
    $\boldsymbol{x} \leftarrow \boldsymbol{x} + \sqrt{\frac{P_{max}}{E}}\mathbf{H}_{ar}\boldsymbol{\delta}_{norm}$
    $\Delta \leftarrow \Delta + \sqrt{\frac{P_{max}}{E}}\boldsymbol{\delta}_{norm}$
end
$\boldsymbol{\delta}^{naive} = \sqrt{P_{max}}\frac{\boldsymbol{\Delta}}{\|\boldsymbol{\Delta}\|_2}$
```

2.4 Non-Targeted White-Box Adversarial Attacks Using Channel Information

The aim of the non-targeted attack is to maximize the loss function $L(\boldsymbol{\theta}, \boldsymbol{r}_{ar}, \boldsymbol{y}^{true})$, where $\boldsymbol{y}^{true}$ is the true class of $\boldsymbol{x}$. FGM is used to linearize the loss function as $L(\boldsymbol{\theta}, \boldsymbol{r}_{ar}, \boldsymbol{y}^{true}) \approx L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^{true}) + (\mathbf{H}_{ar}\boldsymbol{\delta})^T \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^{true})$ that is maximized by setting $\mathbf{H}_{ar}\boldsymbol{\delta} = \alpha \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^{true})$, where α is a scaling factor to constrain the adversarial perturbation power to P_{max}. Based on FGM, we introduce three non-targeted adversarial attacks described in the following subsections.

2.4.1 Naive Non-Targeted Attack

First, the adversary divides its power P_{max} into the number of iterations, E, in Algorithm 2 and allocates $\frac{P_{max}}{E}$ amount of power to the gradient of loss function to tilt the transmitted signal from the transmitter. Next, the adversary calculates the gradient again with respect to the transmitted signal from the transmitter and added perturbation. Then, the adversary adds another perturbation with power $\frac{P_{max}}{E}$ using the new gradient $\boldsymbol{\delta}_{norm}$ as

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \sqrt{\frac{P_{max}}{E}}\mathbf{H}_{ar}\boldsymbol{\delta}_{norm}. \tag{2.7}$$

This scheme generates the best direction to increase the loss function at that specific instance. Finally, the adversary repeats this procedure E times and sums all the gradients of the loss function that were added to the transmitted signal from the transmitter since the adversary can send only one perturbation at a time over the air. Finally, a scaling factor is introduced to satisfy the power constraint at the adversary. The details are presented in Algorithm 2.

2.4.2 Minimum Mean Squared Error (MMSE) Attack

The non-targeted MMSE attack is designed similar to the targeted MMSE attack. The adversary first obtains $\boldsymbol{\delta}^{NoCh}$ from the naive non-targeted attack with $\mathbf{H}_{ar} = \mathbf{I}$ and uses it to solve problem (2.2). Thus, the solution is the same as the solution to (2.2) except that it has the opposite direction to maximize the loss function, whereas the targeted attack minimizes the loss function. Therefore, the perturbation selected by the MMSE scheme for the non-targeted attack is $\boldsymbol{\delta}^{MMSE} = \boldsymbol{\alpha}^T\boldsymbol{\delta}^{NoCh}$, where $\boldsymbol{\alpha} \in \mathbb{R}^p$ and the jth element of $\boldsymbol{\alpha}$ is $\alpha_j = \frac{\gamma h^*_{ar,j}}{h^*_{ar,i}h_{ar,j}+\lambda}$.

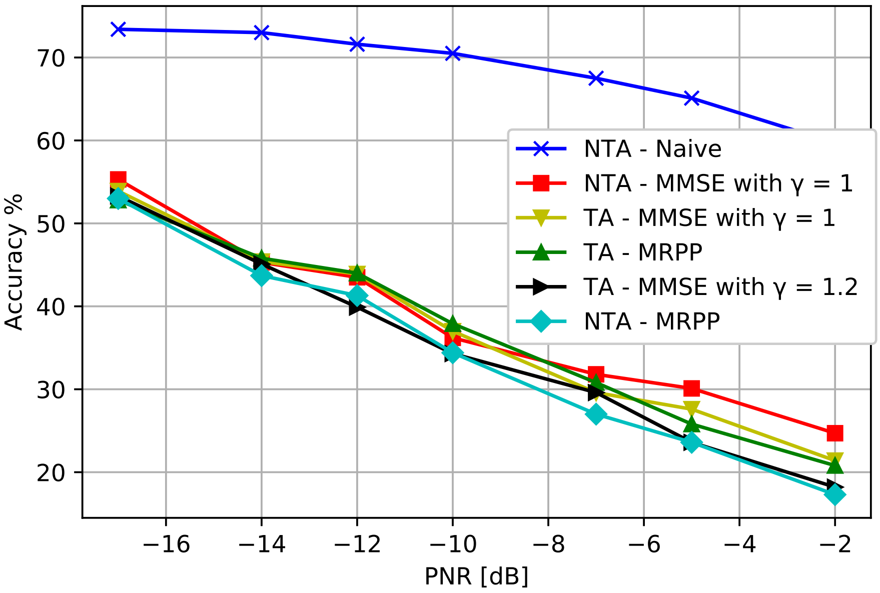


Figure 2.3: Classifier accuracy under different white-box adversarial attacks when SNR = 10 dB.

2.4.3 Maximum Received Perturbation Power (MRPP) Attack

As we have seen in the targeted MRPP attack, the attack should be in the form of $\boldsymbol{\delta}^{MRPP} = \mathbf{H}_{ar}^{*}\boldsymbol{\delta}^{target}$ to maximize the received perturbation power at the receiver. Thus, the naive non-targeted attack is changed to create the MRPP non-targeted attack by replacing $\boldsymbol{\delta}_{norm}$ in Line 5 of Algorithm 2 with

$$\boldsymbol{\delta}_{norm} = \frac{\mathbf{H}_{ar}^{*}\nabla_{\boldsymbol{x}}L(\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{y}^{true})}{(\|\mathbf{H}_{ar}^{*}\nabla_{\boldsymbol{x}}L(\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{y}^{true})\|_2)}. \tag{2.8}$$

2.4.4 Attack Performance

The performances of the proposed white-box attacks are compared in Fig. 2.3 under the same simulation environment used in Section 2.3.4. As we discussed in Section 2.3.2, γ can be optimized with line search for the targeted MMSE attack, e.g., it performs better with $\gamma = 1.2$ compared to $\gamma = 1$. The naive non-targeted attack performs poorly compared to other attacks and the non-targeted MRPP attack outperforms all other attacks for most of the PNR values. This can be explained by the freedom of the direction that the non-targeted adversarial attack can take. For targeted attacks, there are only 10 different directions since there are 11 modulation types. However, the non-targeted attack does not have such a restriction. Thus, it is more likely that the non-targeted attack chooses a better direction to enforce a misclassification. Moreover, the computation complexity for the non-targeted attacks is lower compared to the targeted attacks which involve iterations to reach the desired accuracy.

2.5 White-Box Adversarial Attack with Limited Channel Information

The adversarial attacks that are designed in the previous sections use the exact information of the channel (from the adversary to the receiver). However, this information may not always be available in practical scenarios. Therefore, in this section, we present Algorithm 3 to generate adversarial attacks using a lower-dimensional representation of channel distribution. A classical approach for dimensional reduction is the principal component analysis (PCA) algorithm that was also used in [59] for the case when the signal is directly manipulated at the receiver. PCA is used to obtain the principal component that has the largest variance, i.e., PCA finds the principal component that provides the most information about the data in a reduced

Algorithm 3: Adversarial attack with limited channel information using PCA

```
Inputs: $N$ generated channels $\{\mathbf{H}_{ar}^{(1)}, \mathbf{H}_{ar}^{(2)}, \cdots, \mathbf{H}_{ar}^{(N)}\}$, input $\boldsymbol{r}_{tr}$ and model
  of the classifier
Initialize: $\Delta \leftarrow 0$
for $n$ in range(N) do
    Find $\boldsymbol{\delta}^{(n)}$ from white-box attack algorithm using $\mathbf{H}_{ar}^{(n)}$ and $\boldsymbol{r}_{tr}$
    Stack $\boldsymbol{\delta}^{(n)}$ to $\Delta$
end
Compute the first principle direction $\boldsymbol{v}_1$ of $\Delta$ using PCA
$\Delta = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ and $\boldsymbol{v}_1 = \mathbf{V}\boldsymbol{e}_1$
$\boldsymbol{\delta}^{limited} = \sqrt{P_{max}}\boldsymbol{v}_1$
```

dimension by projecting the data onto it. PCA can be calculated by the eigenvalue decomposition of the data covariance matrix or the singular value decomposition of a data matrix.

To generate an adversarial attack with limited channel information, the adversary first observes N realizations of the channel from the adversary to the receiver $\{\mathbf{H}_{ar}^{(1)}, \mathbf{H}_{ar}^{(2)}, \cdots, \mathbf{H}_{ar}^{(N)}\}$. Then, it generates N adversarial attacks using white-box attack algorithms from the previous sections, either targeted or non-targeted, using N realizations of the observed channel and the known input $\boldsymbol{r}_{tr}$ at the classifier. Finally, it stacks N generated adversarial attacks in a matrix and finds the principal component of the matrix to use it as the adversarial attack with limited channel information. The details are presented in Algorithm 3.

2.6 Universal Adversarial Perturbation (UAP) Attack

In the previous sections, the adversary designs a white-box attack with the assumptions that it knows the model of the classifier at the receiver, the channel between the adversary and the receiver, and the exact input at the receiver. However, these assumptions are not always practical in real wireless communications systems. Thus, in this section, we relax these assumptions and present the UAP attacks.

2.6.1 UAP Attack With Pre-Collected Input at the Receiver

We first relax the assumption that the adversary knows the exact input of the classifier. The adversary in the previous attacks generates an input-dependent perturbation, i.e., $\boldsymbol{\delta}$ is designed given the exact input $\boldsymbol{r}_{tr}$. This requires the adversary to always know the input of the classifier, which is not a practical assumption to make and hard to achieve due to synchronization issues. Thus, it is more practical to design an input-independent UAP. We consider two different methods to design an input-independent UAP. We assume that the adversary collects some arbitrary set of inputs $\{\boldsymbol{r}_{tr}^{(1)}, \boldsymbol{r}_{tr}^{(2)}, \cdots, \boldsymbol{r}_{tr}^{(N)}\}$ and associated classes.

2.6.1.1 PCA-Based Input-Independent UAP Attack

We present an algorithm to design the UAP attack using PCA. First, the adversary generates perturbations $\{\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(N)}\}$ with respect to the obtained arbitrary set of inputs $\{\boldsymbol{r}_{tr}^{(1)}, \boldsymbol{r}_{tr}^{(2)}, \cdots, \boldsymbol{r}_{tr}^{(N)}\}$ and the exact channel information using algorithms from the previous sections such as the MRPP attack. To reflect the common characteristics of $\{\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(N)}\}$ in the UAP, we stack these perturbations into a matrix and perform PCA to find the first component of the matrix with the largest eigenvalue. Hence, we use the direction of the first principal component as the direction of UAP for channel $\mathbf{H}_{ar}$. The algorithm for the UAP with N pre-collected input data is similar to Algorithm 3 where we use pre-collected input with exact channel information.

2.6.1.2 VAE-Based Input-Independent UAP Attack

We present an algorithm to design the UAP attack using a variational autoencoder (VAE) that is known to effectively capture complex data characteristics. An autoencoder consists of two DNNs, an encoder to map the input latent space and a decoder to recover the input data from this latent space. VAE extends this setting by describing a probability distribution for each latent attribute instead of providing a single value to describe each latent state attribute. We first collect an arbitrary set of inputs $\{\boldsymbol{r}_{tr}^{(1)}, \boldsymbol{r}_{tr}^{(2)}, \cdots, \boldsymbol{r}_{tr}^{(N)}\}$ to create corresponding perturbations $\{\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(N)}\}$ using Algorithm 1 with $\mathbf{H}_{ar} = \mathbf{I}$. Then, we train the VAE using these perturbations. The encoder structure of the VAE is presented in Table 2.1 and the decoder structure is presented in Table 2.2. Since the encoder learns an efficient way to compress the data into the lower-dimensional feature space, we first use the encoder to decrease the dimension of the perturbations. Specifically, we use k number of perturbations $\{\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(k)}\}$ as inputs to the encoder and get the corresponding outputs $\{\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \cdots, \boldsymbol{z}^{(k)}\}$, where $\boldsymbol{z}^{(i)} \in \mathbb{R}^{q\times 1}, i = 1, \cdots, k$. To reflect the common characteristics of $\{\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(k)}\}$, we average the corresponding outputs $\{\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \cdots, \boldsymbol{z}^{(k)}\}$ and use $\boldsymbol{z}^{avg} = \frac{1}{k}\sum_{i=1}^{k} \boldsymbol{z}^{(i)}$ in the decoder to generate $\boldsymbol{\delta}^{avg}$. Finally, we use channel information to design an input-independent

Table 2.1: Encoder part of the VAE layout

Layer	Output dimension
Input	2 × 128
Conv	2 × 128 ×128
Conv	2×128×40
Flatten	10240
Dense	16
Dense	4

Table 2.2: Decoder part of the VAE layout

Layer	Output dimension
Input	4
Dense	10240
Deconv	2×128×40
Deconv	2 × 128 ×128
Deconv	2×128

UAP as $\boldsymbol{\delta}^{VAE} = \sqrt{P_{max}} \frac{\mathbf{H}_{ar}^* \boldsymbol{\delta}^{avg}}{\|\mathbf{H}_{ar}^* \boldsymbol{\delta}^{avg}\|}$.

2.6.2 UAP Attack With Limited Channel Information

Now, we further relax the assumption that the adversary knows the exact channel between the adversary and the receiver and assume that the adversary only knows the distribution of this channel. We consider two different methods to design the UAP attack with limited channel information.

2.6.2.1 PCA-Based Channel-Independent UAP Attack

To design the UAP attack when the adversary only knows the distribution of the channel, we first generate random realizations of the channels $\{\mathbf{H}_{ar}^{(1)}, \mathbf{H}_{ar}^{(2)}, \cdots, \mathbf{H}_{ar}^{(N)}\}$ from the distribution. Then, we generate $\boldsymbol{\delta}^{(n)}$ using $\boldsymbol{r}_{tr}^{(n)}$ and $\mathbf{H}_{ar}^{(n)}$ for $n = 1, \cdots, N$ instead of using the real channel $\mathbf{H}_{ar}$. We use PCA to find the first component of the matrix and use it as the direction of UAP. The algorithm for PCA-based channel-independent UAP attack is analogous to Algorithm 3 except that we have pre-collected input data and random realizations of channels as opposed to exact input used in Algorithm 3.

2.6.2.2 VAE-Based Channel-Independent UAP Attack

Similar to the PCA-based channel-independent UAP attack, we first generate random realizations of the channel $\{\mathbf{H}_{ar}^{(1)}, \mathbf{H}_{ar}^{(2)}, \cdots, \mathbf{H}_{ar}^{(N)}\}$ from the distribution to train the VAE. We use the encoder to capture the common characteristics of the channel by using k number of channels as inputs to the encoder and average the outputs of the encoder to put into the decoder as we have done in the VAE-based input-independent UAP attack to get $\mathbf{H}_{avg}$. Note that we use the same encoder and decoder structure that is used for VAE-based input-independent UAP attack. Further, we use the steps used in the VAE-based input-independent UAP attack to create $\boldsymbol{\delta}^{avg}$ and design $\boldsymbol{\delta}^{VAE} = \sqrt{P_{max}} \frac{\mathbf{H}_{avg}^{*}\boldsymbol{\delta}^{avg}}{\|\mathbf{H}_{avg}^{*}\boldsymbol{\delta}^{avg}\|}$.

2.6.3 Black-Box UAP Attack

The last assumption that we will relax is the information about the DNN classifier model at the receiver. To relax this assumption, we use the transferability property of adversarial attacks [64]. This property states that the adversarial attack crafted to fool a specific DNN can also fool other DNNs with different models with high probability. Therefore, the adversary generates UAP attacks using a substitute DNN that is trained using an arbitrary collected dataset and applies them to fool the actual DNN at the receiver.

2.6.4 Attack Performance

Fig. 2.4 shows the performance of the adversarial attacks with respect to different levels of information availability. First, we observe that the input-independent UAP using VAE with 40 pre-collected data inputs, where the adversary knows the exact channel information, outperforms other attacks with limited information and even the channel inversion attack. Note that VAE is trained with 2000 samples but only 40 samples are used to actually create one UAP. This result shows the importance of the channel state information over the exact input data when crafting an adversarial attack since the input-dependent attack significantly outperforms the attack with limited channel information. Also, the input-independent UAP using VAE outperforms the input-independent UAP using PCA. However, the performance is similar when VAE and PCA are compared for the channel-independent attack suggesting that the UAP using PCA might be enough to generate an effective UAP attack. The input-independent UAP with 40 pre-collected data inputs even outperforms the targeted channel inversion attack in the high PNR region, where the adversary knows not only the exact channel but also the exact input at the receiver. Furthermore, the black-box UAP with the same architecture performs similar to the attack with limited channel information where we use the substitute DNN that has the same structure of the classifier but is trained with a different training set so that the substitute DNN has different weights. Lastly, when the adversary uses a different DNN architecture to create an attack, its attack performance is comparable to the channel-unaware attack in the high PNR region.

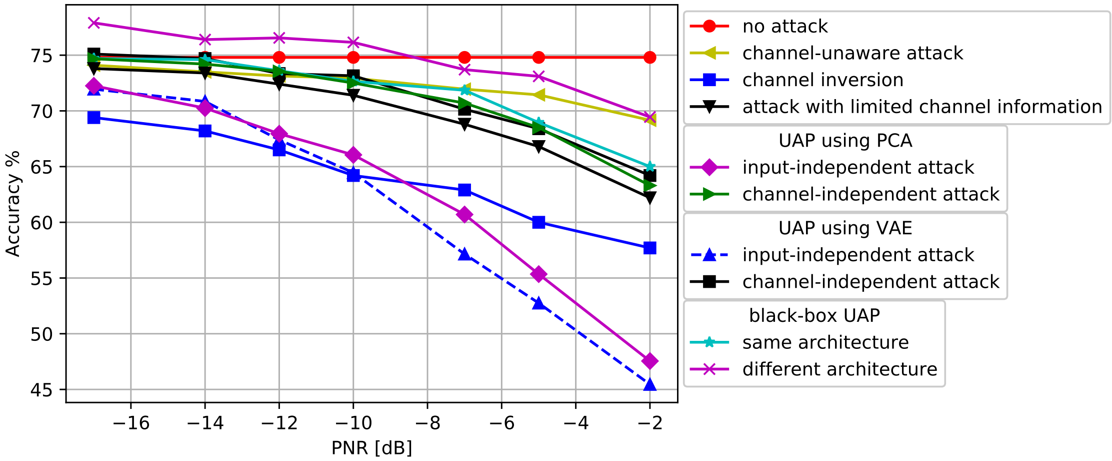


Figure 2.4: Classifier accuracy under UAP attacks with different levels of information availability when SNR = 10 dB.

2.7 Broadcast Adversarial Attack

We extend the adversarial attack to a broadcast (common) adversarial perturbation that is transmitted by the adversary to simultaneously fool multiple classifiers at different receivers. First, we show that the channel-aware adversarial attack built for a particular channel is inherently selective, namely it is only effective against the receiver with that experienced channel and not effective against another receiver with a different experienced channel. Next, we show how to design a common adversarial perturbation by jointly accounting for multiple channels such that the adversary can fool classifiers at multiple receivers simultaneously with a single (omnidirectional) transmission. We present the design of broadcast adversarial attack only for the case of the targeted adversarial attacks, as other types of attacks can be formulated similarly. Note that again we assume the adversary knows the DNN architecture of the receiver, input at the ith receiver, and the channel between the adversary and the ith receiver in this section.

2.7.1 Individual Design of Broadcast Adversarial Attack (IDBA)

We design IDBA by building upon Algorithm 1 (that was designed against a signal receiver). By applying Algorithm 1 separately against each of m receivers, we obtain m adversarial perturbations, $\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(m)}$, where $\boldsymbol{\delta}^{(i)}$ is the adversarial perturbation generated using Algorithm 1 for the ith receiver. Since the adversary aims to fool all the receivers by broadcasting a single adversarial perturbation, we combine individual perturbations as a weighted sum, $\sum_{i=1}^{m} w_i \boldsymbol{\delta}^{(i)}$, where w_i is the weight for adversarial perturbation $\boldsymbol{\delta}^{(i)}$, and then normalize it to satisfy the power constraint at the adversary (higher w_i implies more priority given to $\boldsymbol{\delta}^{(i)}$). Note that the optimal weights can be found by line search subject to $\sum_{i=1}^{m} w_i = 1$ and $w_i \geq 0$ for each i. Numerical results suggest that the search space can be constrained by selecting weights inversely proportional to channel gains of corresponding receivers.

2.7.2 Joint Design of Broadcast Adversarial Attack (JDBA)

We generate the adversarial perturbation JDBA by modifying Algorithm 1. Specifically, we change the computation of $\boldsymbol{\delta}^{c}_{norm}$ in Line 5 of Algorithm 1 from (2.8) to

$$\boldsymbol{\delta}^{c}_{norm} = \frac{\sum_{i=1}^{m} w_i \mathbf{H}_{ar}^{(i)*} \nabla_{\boldsymbol{x}} L_i(\boldsymbol{\theta}, \boldsymbol{r}_{tr}^{(i)}, \boldsymbol{y}^c)}{(\| \sum_{i=1}^{m} w_i \mathbf{H}_{ar}^{(i)*} \nabla_{\boldsymbol{x}} L_i(\boldsymbol{\theta}, \boldsymbol{r}_{tr}^{(i)}, \boldsymbol{y}^c) \|_2)}, \tag{2.9}$$

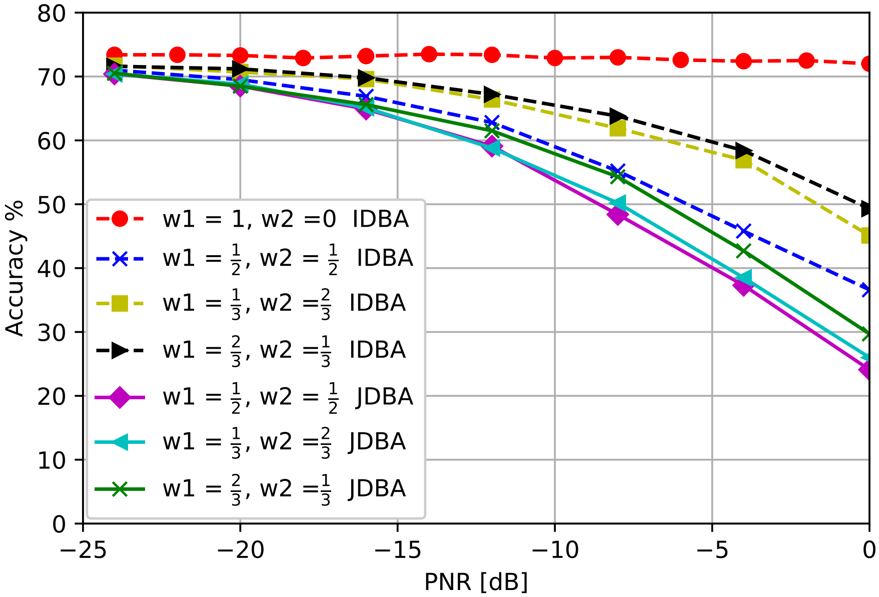


Figure 2.5: Classifier accuracy under broadcast adversarial attacks when SNR = 10 dB.

search over all $C - 1$ classes, and choose the class that fools most of the receivers. Note that JDBA searches for a common direction of $\boldsymbol{\delta}^c_{norm}$ for all receivers. On the other hand, IDBA searches separately for directions of adversarial perturbations for different receivers and then combines them to a common direction.

2.7.3 Attack Performance

For performance evaluation, we assume that there are two receivers, $m = 2$, with different classifiers which have the same architecture, but trained with different training sets and the adversary generates a broadcast adversarial perturbation to fool both receivers simultaneously. We assume Rayleigh fading for both channel $h_{ar_1,j} = K(\frac{d_0}{d})^\gamma \psi h_{ray_1,j}$ from the adversary to receiver 1, and the channel $h_{ar_2,j} = K(\frac{d_0}{d})^\gamma \psi h_{ray_2,j}$ from the adversary to receiver 2, where $h_{ray_1,j}$ and $h_{ray_2,j} \sim$

Rayleigh$(0, 1)$.

Fig. 2.5 shows the accuracy of the classifiers with respect to different broadcast adversarial attacks. Note that the accuracy is the same for both classifiers since we assume that the broadcast adversarial attack is successful if it fools both receivers at the same time. JDBA outperforms IDBA for all PNRs. Furthermore, the weight selection $w_1 = \frac{1}{2}, w_2 = \frac{1}{2}$ outperforms other weight selections suggesting that the weight selection is related to the channel distribution. Also, when weights $w_1 = 1$ and $w_2 = 0$ are selected in IDBA, the attack generated only for receiver 1 has no effect on receiver 2, i.e., the attack selectively fools receiver 1 due to different channel effects. This observation validates the need to craft a broadcast adversarial perturbation.

Next, we distinguish channel distributions for the two receivers and evaluate the classifier accuracy under the JDBA attack in Fig. 2.6. The channel from the adversary to receiver 1, and the channel from the adversary to receiver 2 are modeled with Rayleigh fading of different variances, i.e., $h_{ray_1,j} \sim$ Rayleigh$(0, 1)$ and $h_{ray_2,j} \sim$ Rayleigh$(0, 2)$. The weight selection $w_1 = \frac{2}{3}$ and $w_2 = \frac{1}{3}$ (where the weights are selected as inversely proportional to the variance of the channel as we have seen in Fig. 2.5) outperforms other weight selections.

2.8 Defense Against Adversarial Attacks

In this section, we introduce a defense method to increase the robustness of wireless signal classification models against adversarial perturbations. There are several

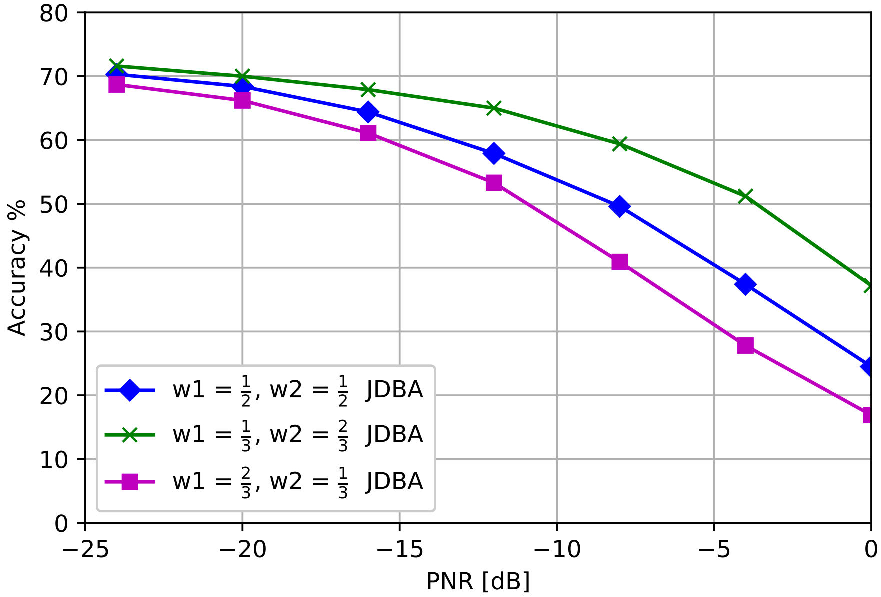


Figure 2.6: Classifier accuracy under broadcast adversarial attacks at receivers with different channel distributions when SNR = 10 dB.

methods developed in the computer vision domain to defend against adversarial attacks. One method is *adversarial training* that calculates an adversarial perturbation and adds the perturbed samples to the training set to increase the model robustness. However, adversarial training only strengthens the classifier in a few perturbation directions, and as shown in [62], the classifier defended with adversarial training remains susceptible to perturbations in other directions making the classifier still vulnerable to black-box attacks. In response, a line of work on *certified defense* has been considered [60–62, 65]. A classifier is said to be *certifiably robust* if the classifier's prediction of a sample x is constant in a small neighborhood around x, often defined by an ℓ_2 or ℓ_∞ ball. *Randomized smoothing* (also referred to as Gaussian smoothing) is a certified defense approach, which augments the training set with Gaussian noise to increase the robustness of the classifier to multiple directions. Recent work has shown that a tight robustness guarantee in ℓ_2 norm can be achieved with randomized smoothing with Gaussian noise [60]. Hence, we apply randomized smoothing as a defense mechanism to make the modulation classifier robust against wireless adversarial attacks over the air.

In randomized smoothing, there are two parameters σ and k that the defender can control, namely the standard deviation of the Gaussian noise σ and the number of noisy samples k added to each training sample r_i, i.e., $r_i + n_1, r_i + n_2, \ldots, r_i + n_k$, where n_j is Gaussian noise with variance σ^2. These two parameters balance the trade-off between the resulting classification accuracy and the robustness against perturbations. Note that unlike the images used as input samples in computer vision, the received signals in wireless communications are already subject to noise, and randomized smoothing slightly increases the noise level. However, as we will see in Section 2.8.1, as the number of data samples in the training set is less than the number of parameters in the neural network, data augmentation in fact improves the classifier performance and Gaussian smoothing does not cause any degradation.

2.8.1 Randomized Smoothing During Training

We evaluate the accuracy of the classifier for different values of σ and k selected for Gaussian noise augmentation. First, we fix $k = 10$ and change σ in Fig. 2.7 to study the impact of the power of the Gaussian noise added to the training set. The accuracy of the classifier trained with small values of σ such as $\sigma = 0.001$ and

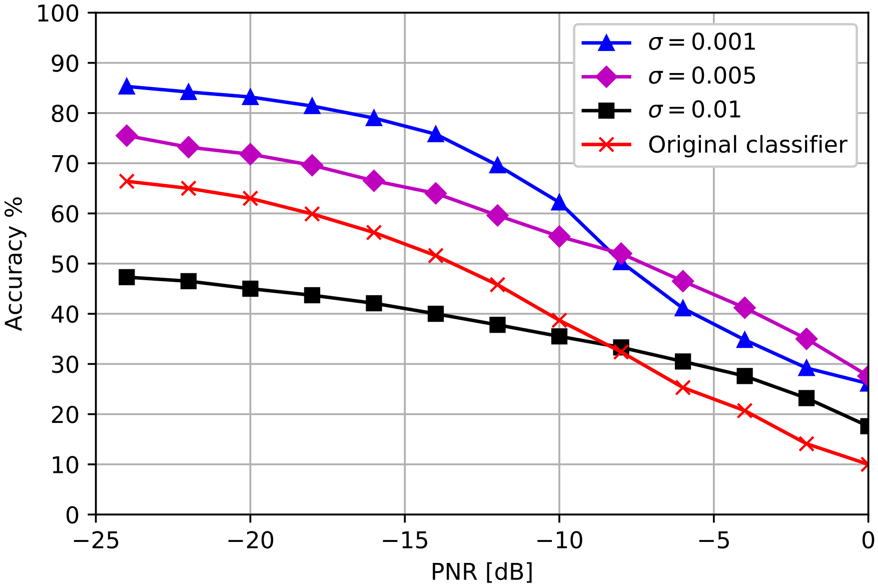


Figure 2.7: Classifier accuracy when trained with $(10, \sigma)$ Gaussian noise augmentation with different σ and SNR $= 10$ dB.

$\sigma = 0.005$ is larger than that of the original classifier across all PNR values. This result shows that randomized smoothing as training data augmentation improves the classifier accuracy for a small σ value, while degrading the performance when noise with a large variance is introduced.

Furthermore, there is an intersection between accuracy versus PNR curves for $\sigma = 0.001$ and $\sigma = 0.005$ due to the density of the norm ball that is created around each training set. These results suggest that when σ is small, the classifier becomes more robust to a small perturbation, but more susceptible to a large perturbation. We observe a similar crossover for large σ values (e.g., $\sigma = 0.01$) where the classifier is more robust to a large perturbation, but susceptible to a small perturbation.

In Fig. 2.8, we keep the noise variance σ constant and vary the number of samples k added to the training set to investigate its effect on the classifier robustness. Our results demonstrate that by adding more augmentation samples to training

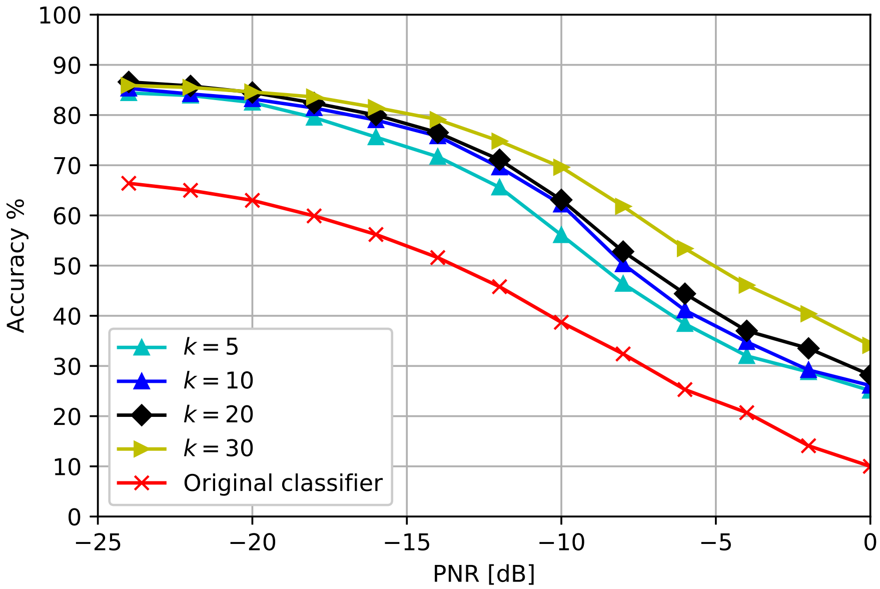


Figure 2.8: Classifier accuracy when trained with $(k, 0.001)$ Gaussian noise augmentation with different k and SNR $= 10$ dB.

set increases the robustness of the classifier against the adversarial perturbation when compared to the original classifier trained without any defense. However, this defense advantage comes at the cost of increased training time.

2.8.2 Certified Defense in Test Time

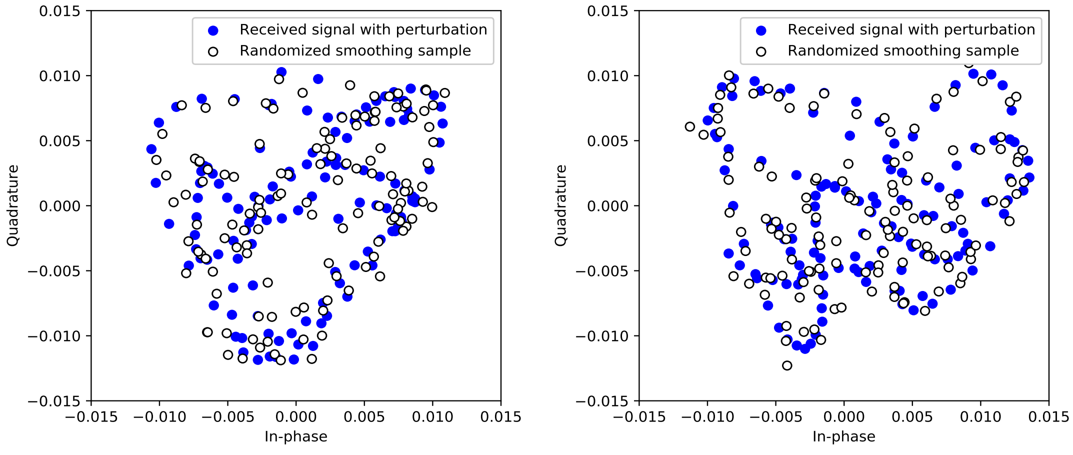
The defense results can be certified with a desired confidence by using randomized smoothing in test time. Consider the classifier trained with $k = 20$ and $\sigma = 0.001$. For each perturbed test sample, we draw k Gaussian noise samples with the same variance and label them with the classifier. Let $\hat{c}_A$ and $\hat{c}_B$ denote the classes that occurred most and second most, and n_A and n_B represent the occurrence of these two classes, respectively. We then apply a two-sided hypothesis test and check if $\text{BinomPValue}(n_A, n_A + n_B, q) \leq \alpha$ condition is satisfied, where $\text{BinomPValue}(\cdot)$

returns p-value of the two-sided hypothesis test that $n_A \sim \text{Binomial}(n_A+n_B, q)$ [60], $1-q$ is the confidence level, and α is the probability of returning an incorrect answer. If the condition is satisfied, the classifier is very confident in its prediction. If not, then the classifier abstains and does not make a prediction. For example, when we consider 95% confidence, we observe that a test sample that belongs to class "QAM64" is perturbed by the adversary to be classified as "QAM16" at the receiver. When randomized smoothing is applied in test time, we observe that the classifier infers the samples as class "QAM16" for 6 times and as class "QAM64" for 14 times, resulting in an $\text{BinomPValue}(\cdot) = 0.115$. Since the condition is not satisfied, the classifier abstains.

Another example is a test sample of "QAM64" that is perturbed by the adversary to be classified as "8PSK". When randomized smoothing is applied in test time, the classifier correctly infers all k samples as "QAM64" and the defense is certified with 95% confidence. The constellation of the two examples are shown respectively in Fig. 2.9(a)-(b).

2.9 Conclusion

We presented over-the-air adversarial attacks against DL-based modulation classifiers by accounting for realistic channel and broadcast transmission effects. Specifically, we considered targeted, non-targeted and UAP attacks with different levels of uncertainty regarding channels, transmitter inputs and DNN classifier models. We showed that these channel-aware adversarial attacks can successfully fool a modu-



(a)

(b)

Figure 2.9: Constellation points of the received signal with adversarial perturbation and randomized smoothing samples for the cases (a) when the classifier abstains and (b) when the classifier recovers the perturbed signal and correctly infers the label.

lation classifier over-the-air. Then, we introduced broadcast adversarial attacks to simultaneously fool multiple classifiers at different receivers with a single perturbation transmission. First, we showed that an adversarial perturbation designed for a particular receiver is not effective against another receiver due to channel differences. Therefore, we designed a common adversarial perturbation by considering all channel effects jointly and showed that this broadcast attack can fool all receivers. Finally, we presented a certified defense method using randomized smoothing, and showed that it is effective in reducing the impact of adversarial attacks on the modulation classifier performance.

CHAPTER 3

Adversarial Attacks with Multiple Antennas Against Deep Learning-Based Modulation Classifiers

3.1 Introduction

In Chapter 2, we show that DNNs, modulation classifiers, are susceptible to adversarial attacks that are generated by an adversary with a single antenna. In an extended scenario, there could be an adversary with multiple antennas trying to manipulate the target using more channel diversity. Thus, in this chapter, we consider an adversary that has multiple antennas to transmit adversarial perturbations in the presence of channel effects and aims to cause misclassification at the modulation classifier. We investigate how to use multiple antennas to generate multiple concurrent perturbations over different channel effects (subject to a total power budget) to enforce the misclassification at the DNN-based modulation classifier. As shown in Chapter 2, transmitting random (e.g., Gaussian) noise to decrease the accuracy of the classifier at the receiver is ineffective as an adversarial attack, since random noise cannot manipulate the input to the DNN in a specific direction as needed in an adversarial attack. Therefore, increasing the perturbation power with random noise transmitted over multiple antennas remains ineffective. Instead, the adversary needs to carefully craft the perturbation for each antenna.

We design a white-box attack where the adversary knows the receiver's classifier architecture, input at the receiver, and the channel between the adversary and the receiver. The adversary signal is time-aligned with the transmitted signal and uses the maximum received perturbation power (MRPP) attack that was introduced in Chapter 2. We propose a genie aided adversarial attack where the adversary selects only one antenna aided by genie to transmit the perturbation that would result in the worst classification performance at the receiver. Then, we consider transmitting using all the antennas at the adversary where the power allocation is based on the channel gains, either proportional or inversely proportional to the channel gains. Finally, we propose the elementwise maximum channel gain (EMCG) attack to utilize the channel diversity efficiently by selecting the antenna with the best channel gain at the symbol level to transmit perturbations. During the simulations, we show that increasing the number of antennas at the adversary significantly improves the attack performance by better exploiting the channel diversity to craft and transmit adversarial perturbations.

3.2 System Model

We consider a wireless communication system that consists of a transmitter, a receiver, and an adversary as shown in Fig. 3.1. Both the transmitter and the receiver are equipped with a single antenna. The receiver uses a pre-trained DL-based

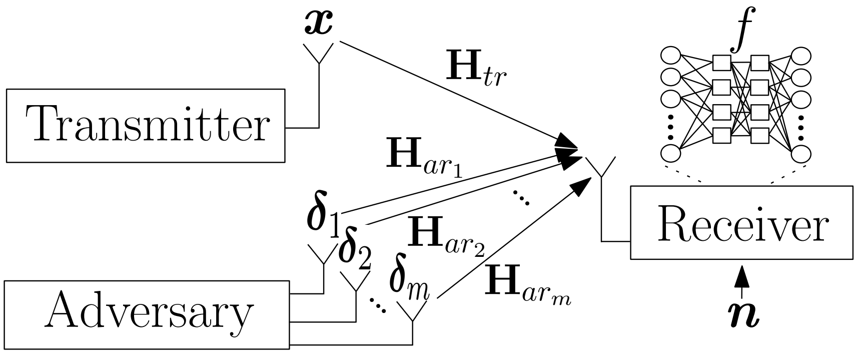


Figure 3.1: Wireless communication system with a transmitter, a receiver, and an adversary with multiple antennas.

classifier on the received signals to classify the modulation type that is used at the transmitter. The adversary has m antennas to launch a white-box adversarial attack to cause misclassification at the receiver. The white-box attack can be considered as an upper-bound for other attacks with limited information. The assumptions on the knowledge of the adversary can be relaxed as shown in Chapter 2.

The DNN classifier at the receiver is denoted by $f(\cdot;\boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}^C$, where $\boldsymbol{\theta}$ is the set of parameters of the DNN decided in the training phase and C is the number of modulation types. Note $\mathcal{X} \subset \mathbb{C}^p$, where p is the dimension of the complex-valued I/Q (in-phase/quadrature) inputs to the DNN that can also be represented by concatenation of two real-valued inputs. A modulation type $\hat{l}(\boldsymbol{x}_{in}, \boldsymbol{\theta}) = \arg\max_k f_k(\boldsymbol{x}_{in}, \boldsymbol{\theta})$ is assigned by f to input $\boldsymbol{x}_{in} \in \mathcal{X}$ where $f_k(\boldsymbol{x}_{in}, \boldsymbol{\theta})$ is the output of classifier f corresponding to the kth modulation type.

The channel from the transmitter to the receiver is $\boldsymbol{h}_{tr}$ and the channel from ith antenna of the adversary to the receiver is $\boldsymbol{h}_{ar_i}$, where $\boldsymbol{h}_{tr} = [h_{tr,1}, h_{tr,2}, \cdots, h_{tr,p}]^T \in \mathbb{C}^{p\times 1}$ and $\boldsymbol{h}_{ar_i} = [h_{ar_i,1}, h_{ar_i,2}, \cdots, h_{ar_i,p}]^T \in \mathbb{C}^{p\times 1}$. If the transmitter transmits $\boldsymbol{x}$, the receiver receives $\boldsymbol{r}_t = \mathbf{H}_{tr}\boldsymbol{x} + \boldsymbol{n}$, if there is no adversarial attack, or receives $\boldsymbol{r}_a = \mathbf{H}_{tr}\boldsymbol{x} + \sum_{i=1}^{m} \mathbf{H}_{ar_i}\boldsymbol{\delta}_i + \boldsymbol{n}$, if the adversary transmits the perturbation signal $\boldsymbol{\delta}_i$ at the ith antenna, where $\mathbf{H}_{tr} = \text{diag}\{h_{tr,1}, \cdots, h_{tr,p}\} \in \mathbb{C}^{p\times p}, \mathbf{H}_{ar_i} = \text{diag}\{h_{ar_i,1}, \cdots, h_{ar_i,p}\} \in \mathbb{C}^{p\times p}$, $\boldsymbol{\delta}_i \in \mathbb{C}^{p\times 1}$ and $\boldsymbol{n} \in \mathbb{C}^{p\times 1}$ is complex Gaussian noise. For a stealth attack, the adversarial perturbations on antennas are constrained as $\sum_{i=1}^{m} \|\boldsymbol{\delta}_i\|_2^2 \leq P_{max}$ for some suitable power P_{max}. To determine these perturbations with respect to the transmitted signal $\boldsymbol{x}$, the adversary solves the following optimization problem

$$\begin{aligned} \underset{\{\boldsymbol{\delta}_i\}}{\arg\min} \quad & \sum_{i=1}^{m} \|\boldsymbol{\delta}_i\|_2^2 \\ \text{subject to} \quad & \hat{l}(\boldsymbol{r}_t, \boldsymbol{\theta}) \neq \hat{l}(\boldsymbol{r}_a, \boldsymbol{\theta}), \\ & \sum_{i=1}^{m} \|\boldsymbol{\delta}_i\|_2^2 \leq P_{max}. \end{aligned} \tag{3.1}$$

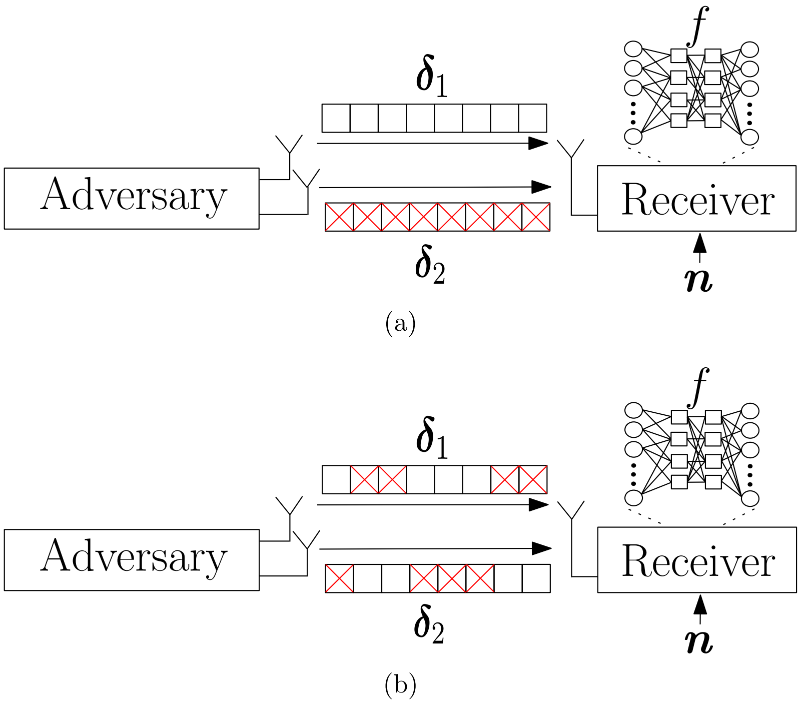
In (3.1), the objective is to minimize the perturbation power subject to two constraints where the receiver misclassifies the received signal and the budget for perturbation power is not exceeded. However, solving optimization problem (3.1) is difficult because of the inherent structure of the DNN. Thus, different methods have been proposed to approximate the adversarial perturbation. For instance, FGM is a computationally efficient method for generating adversarial attacks by linearizing the loss function of the DNN classifier. We denote the loss function of the model by $L(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{y} \in \{0, 1\}^C$ is the one-hot encoded class vector. Then, FGM linearizes this loss function in a neighborhood of $\boldsymbol{x}$ and uses this linearized function for optimization. Since the adversary uses more than one antenna, the adversary needs to utilize the diversity of channels to craft more effective perturbations. For that purpose, we introduce different methods in Section 3.3.

3.3 Adversarial Attacks Using Multiple Antennas

In this section, we introduce different methods to utilize multiple antennas at the adversary to improve the attack performance. Note that the adversary can allocate power differently to each antenna and increase the channel diversity by using multiple antennas. In this work, we apply the targeted MRPP attack in Chapter 2, which has been developed from the attack in [59] by accounting for additional channel effects. The MRPP attack searches over all modulation types to cause misclassification at the receiver and chooses one modulation type that needs the least power to cause the misclassification.

3.3.1 Single-Antenna Genie-Aided (SAGA) Attack

We first begin with an attack where the adversary allocates all the power to only one antenna for the entire symbol block of an input to the classifier at the receiver as shown in Fig. 3.2(a). In this attack, we assume that the adversary is aided by a genie and thus knows in advance the best antenna out of m antennas that causes a misclassification. Then, the genie-aided adversary puts all the power to that one specific antenna to transmit the adversarial perturbation.



(a)

(b)

Figure 3.2: Illustration of (a) SAGA attack and (b) EMCG attack.

3.3.2 Proportional to Channel Gain (PCG) Attack

To exploit the channel with the better channel gain, the adversary allocates more power to better channels. Specifically, the power allocation for the ith antenna is proportional to the channel gain $\|\mathbf{h}_{ar_i}\|_2$. The adversarial perturbation that is transmitted by each antenna is generated using the MRPP attack as before and transmitted with the power allocated to each antenna. During the attack generation process, the adversary can set the common target modulation type of misclassification for all antennas or independent target modulation type of misclassification for each antenna.

Algorithm 4: PCG attack with common target

```
Inputs: input $\boldsymbol{r}_t$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$ and model of
  the classifier $L(\boldsymbol{\theta}, \cdot, \cdot)$
Initialize: $\boldsymbol{\varepsilon} \leftarrow \mathbf{0}^{C\times 1}$, $w_i = \frac{\|\mathbf{h}_{ar_i}\|_2}{\sum_{j=1}^{m}\|\mathbf{h}_{ar_j}\|_2}, i = 1, \cdots, m$
for class-index c in range(C) do
    $\varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0$
    for $i = 1$ to $m$ do
        $\boldsymbol{\delta}_i^c = \frac{\mathbf{H}^*_{ar_i}\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{r}_{tr},\boldsymbol{y}^c)}{(\|\mathbf{H}^*_{ar_i}\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{r}_{tr},\boldsymbol{y}^c)\|_2)}$
    end
    while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
        $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
        $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x} - \varepsilon_{avg} \sum_{i=1}^{m} w_i \mathbf{H}_{ar_i} \boldsymbol{\delta}_i^c$
        if $\hat{l}(\boldsymbol{x}_{adv}) == l_{true}$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
        else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
    end
    $\boldsymbol{\varepsilon}[c] = \varepsilon_{max}$
end
$target = \arg\min \boldsymbol{\varepsilon}$, $\boldsymbol{\delta}_i = \boldsymbol{\varepsilon}[target] w_i \boldsymbol{\delta}_i^{target}$ for $\forall i$
```

3.3.2.1 PCG Attack With Common Target

The adversary sets a common target modulation type for all antennas to cause the specific misclassification at the receiver. The adversary decides the common target modulation type that needs the least power to fool the receiver. The details are presented in Algorithm 4.

3.3.2.2 PCG Attack With Independent Targets

For the ith antenna, the adversary decides the individual target modulation type for perturbation $\boldsymbol{\delta}_i$. Each antenna independently chooses the target modulation type which uses the least power to cause misclassification at the receiver. These modulation types may differ from each other. By setting individual target modulation type

Algorithm 5: PCG attack with independent targets

```
Inputs: input $\boldsymbol{r}_t$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$ and model of
  the classifier $L(\boldsymbol{\theta}, \cdot, \cdot)$
Initialize: $\boldsymbol{\varepsilon} \leftarrow \mathbf{0}^{C\times 1}$, $w_i = \frac{\|\mathbf{h}_{ar_i}\|_2}{\sum_{j=1}^{m} \|\mathbf{h}_{ar_j}\|_2}, i = 1, \cdots, m$
for $i = 1$ to $m$ do
    for class-index $c$ in range($C$) do
        $\varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0$
        $\boldsymbol{\delta}_i^c = \frac{\mathbf{H}^*_{ar_i} \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^c)}{(\|\mathbf{H}^*_{ar_i} \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^c)\|_2)}$
        while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
            $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
            $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x} - \varepsilon_{avg} \sum_{i=1}^{m} w_i \mathbf{H}_{ar_i} \boldsymbol{\delta}_i^c$
            if $\hat{l}(\boldsymbol{x}_{adv}) == l_{true}$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
            else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
        end
    end
    $\boldsymbol{\varepsilon}[c] = \varepsilon_{max}$
    $target = \arg\min \boldsymbol{\varepsilon}$, $\boldsymbol{\delta}_i = \boldsymbol{\varepsilon}[target] w_i \boldsymbol{\delta}_i^{target}$
end
```

for each antenna, the adversary can exploit the channel since each antenna chooses what is best for itself. The details are presented in Algorithm 5.

3.3.3 Inversely Proportional to Channel Gain (IPCG) Attack

In contrast to the PCG attack, the adversary allocates more power to weak channels to compensate for the loss over the weak channels, i.e., inversely proportional to the channel gain. The perturbations that are transmitted by each antenna are generated using the MRPP attack and the power for each antenna is determined to be inversely proportional to the channel gain. As in the PCG attack, the IPCG attack can be also crafted with a common target or independent targets for all antennas. The algorithm is the same as Algorithm 4 for common target and Algorithm 5 for the independent targets except that w_i changes to be inversely proportional to the channel, i.e.,

Algorithm 6: EMCG attack

```
Inputs: input $\boldsymbol{r}_t$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$ and model of
  the classifier $L(\boldsymbol{\theta}, \cdot, \cdot)$
Initialize: $\boldsymbol{\varepsilon} \leftarrow \mathbf{0}^{C\times 1}$, $k \leftarrow \mathbf{0}^{p\times 1}$, $\boldsymbol{\delta}_i \leftarrow \mathbf{0}^{p\times 1}$ for $\forall i$
for $i = 1$ to $p$ do
    $h_{vir,i} = \max\{\|h_{ar_1,i}\|_2, \cdots, \|h_{ar_m,i}\|_2\}$
    $k[i] = \arg\max\{\|h_{ar_1,i}\|_2, \cdots, \|h_{ar_m,i}\|_2\}$
end
Virtual channel : $\mathbf{H}_{vir} = \text{diag}\{h_{vir,1}, \cdots, h_{vir,p}\}$
for class-index $c$ in range($C$) do
    $\varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0$
    $\boldsymbol{\delta}^c = \frac{\mathbf{H}^*_{vir} \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^c)}{(\|\mathbf{H}^*_{vir} \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{tr}, \boldsymbol{y}^c)\|_2)}$
    while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
        $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
        $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x} - \varepsilon_{avg} \mathbf{H}_{vir} \boldsymbol{\delta}^c$
        if $\hat{l}(\boldsymbol{x}_{adv}) == l_{true}$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
        else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
    end
    $\boldsymbol{\varepsilon}[c] = \varepsilon_{max}$
end
$target = \arg\min \boldsymbol{\varepsilon}$, $\boldsymbol{\delta}^{vir} = \boldsymbol{\varepsilon}[target] \boldsymbol{\delta}^{target}$
for $i = 1$ to $p$ do
    $\boldsymbol{\delta}_{k[i]} = \boldsymbol{\delta}^{vir}[i]$
end
Transmit $\boldsymbol{\delta}_i$, $i = 1, \cdots, m$
```

$$w_i = \frac{1}{\|\mathbf{h}_{ar_i}\|_2 \left(\frac{1}{\sum_{j=1}^{m} \|\mathbf{h}_{ar_j}\|_2} \right)}, i = 1, \cdots, m.$$

3.3.4 Elementwise Maximum Channel Gain (EMCG) Attack

Unlike the previous attacks that considered the channel gain of the channel vector with dimension $p \times 1$ as a way to allocate power among antennas, the EMCG attack considers the channel gain of each element of the channel to fully utilize the channel diversity as shown in Fig. 3.2(b). First, the adversary compares the channel gains elementwise and selects one antenna that has the largest channel gain at each instance. Specifically, the adversary finds and transmits with the antenna

$j^* = \underset{j=1,\cdots,m}{\arg\max}\{\|h_{ar_j,t}\|_2\}$ that has the largest channel gain at instance t. Further, a virtual channel $h_{vir,t}$ at instance t is defined as the channel with the largest channel gain among antennas. Then, the adversary generates the perturbation $\boldsymbol{\delta}^{vir}$ with respect to $\boldsymbol{h}_{vir} = [h_{vir,1}, \cdots, h_{vir,p}]^T$ using the MRPP attack and transmits each element of $\boldsymbol{\delta}^{vir}$ with the antenna that has been selected previously. The details are in Algorithm 6.

3.4 Simulation Results

In this section, we compare the performances of the attacks introduced in Section 3.3 (along with the MRPP attack from Chapter 2 where the adversary has a single antenna) to investigate how the number of antennas at the adversary affects the attack performance. Also, multiple adversaries that are each equipped with a single antenna and located at different positions are considered to motivate the need to craft attacks for the adversary with multiple antennas.

To evaluate the performance, we use the VT-CNN2 classifier from [52] as the modulation classifier (also used in [59, 66]) where the classifier consists of two convolution layers and two fully connected layers, and train it with GNU radio ML dataset RML2016.10a [63]. The dataset contains 220,000 samples where half of the samples are used for training and the other half are used for testing. Each sample corresponds to one specific modulation type at a specific signal-to-noise ratio (SNR). There are 11 modulations which are BPSK, QPSK, 8PSK, QAM16, QAM64, CPFSK, GFSK, PAM4, WBFM, AM-SSB and AM-DSB. We follow the

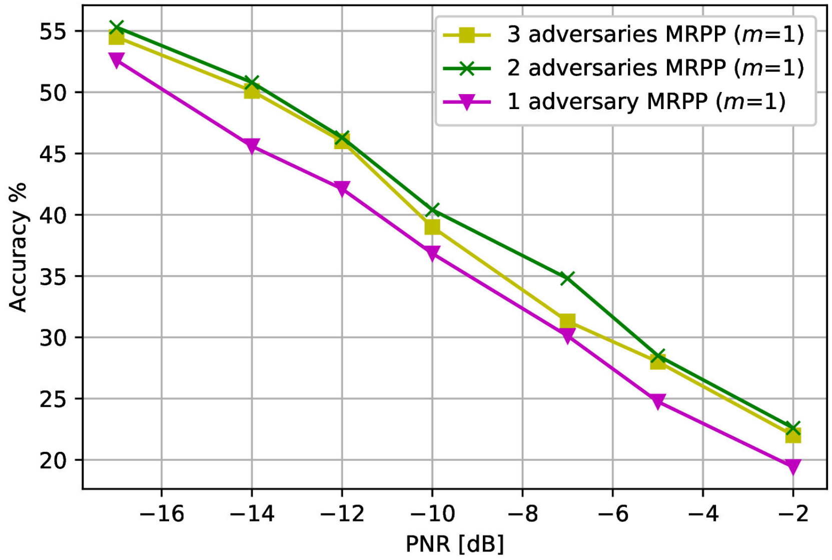


Figure 3.3: Classifier accuracy with respect to the number of adversaries with single antenna.

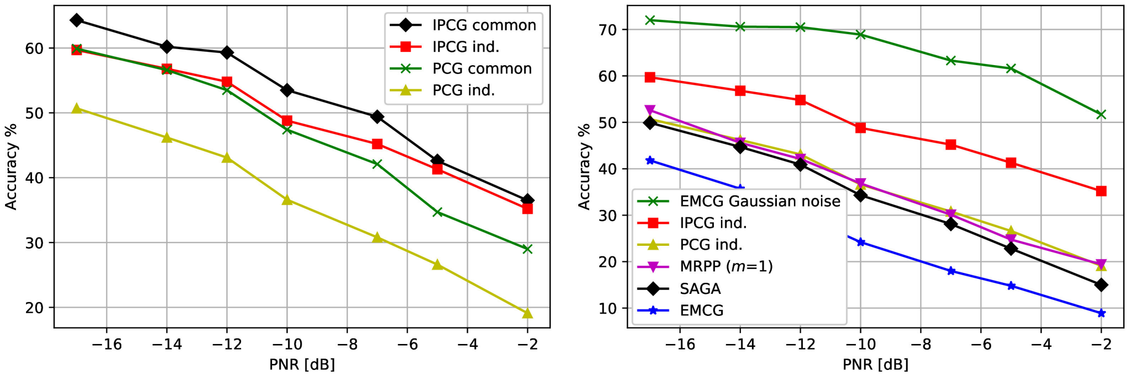
same setup of [52], using Keras with TensorFlow backend, where the input sample to the modulation classifier is 128 I/Q channel symbols.

In the simulations, we introduce the channel between the ith antenna at the adversary and the receiver as a Rayleigh fading channel with path-loss and shadowing, i.e., $h_{ar_i,j} = K(\frac{d_0}{d})^{\gamma}\psi h_{i,j}$ where $K = 1, d_0 = 1, d = 10, \gamma = 2.7, \psi \sim \text{Lognormal}(0, 8)$ and $h_{i,j} \sim \text{Rayleigh}(0, 1)$. We assume that channels between antennas are independent (except for Fig. 3.5) and fix SNR as 10dB. We evaluate the attack performance as a function of the perturbation-to-noise ratio (PNR) from [59]. The PNR represents the relative perturbation power with respect to the noise power. As the PNR increases, the power of the perturbation relatively increases compared to the noise power making the perturbation more likely to be detected by the receiver since it becomes more distinguishable from noise.

First, we compare the classifier accuracy of an adversary equipped with a single antenna using the MRPP attack to the case of multiple adversaries where each adversary has a single antenna using the MRPP attack. For a fair comparison, total power that is used among adversaries is kept the same as the power used by the single adversary and the power is equally divided among adversaries. Results are shown in Fig. 3.3. Note that for the case of two or more adversaries, adversaries are not synchronized and do not collaborate with each other as they are physically not co-located meaning that they attack with independent targets. We observe that the accuracy of the classifier does not drop although more adversaries are used to attack the classifier. This result suggests that dividing the power equally is not helpful and thus motivates the need for an adversary with multiple antennas to choose power allocation on antennas and exploit the channel diversity.

Adversarial attacks using two antennas with common target and independent targets are compared in Fig. 3.4(a). The PCG attack outperforms the IPCG attack regardless of whether the target is common or independent showing that the power allocation among antennas is important. Also, choosing an independent target at each antenna performs better than the common target case for both PCG and IPCG attacks suggesting that choosing the best target (determined by the channel realization) for each antenna is more effective.

Fig. 3.4(b) presents the classifier accuracy at the receiver when the adversary transmits an adversarial perturbation with $m = 2$ antennas using different attacks that are introduced in Section 3.3. The EMCG attack with Gaussian noise transmitted by the adversary with two antennas is compared with the adversarial perturbation with two antennas using the MRPP attack at each antenna. The use



(a)

(b)

Figure 3.4: (a) Classifier accuracy when adversarial attacks with common target and independent targets are transmitted at the adversary. (b) Classifier accuracy under different attack schemes.

of Gaussian noise as perturbation results in poor attack performance although the EMCG attack is used to determine the antenna to transmit supporting the use of the MRPP attack. Fig. 3.4(b) shows that although the adversary uses two antennas, the accuracy of the classifier is higher than the case under the MRPP attack of an adversary with single antenna when the IPCG attack with independent targets is used. Also, the performance of the PCG attack with independent targets is similar to the performance of the MRPP attack of the adversary with a single antenna although the adversary puts more power to the better channel. We observe that the SAGA attack slightly outperforms the MRPP attack of an adversary with a single antenna suggesting that the SAGA attack takes advantage of having two channels to choose from. Moreover, the EMCG attack significantly outperforms other attacks by fully utilizing the channel diversity.

So far, results have been obtained under the assumption that channels between the antennas are independent, which also yields zero covariance. Next, we consider

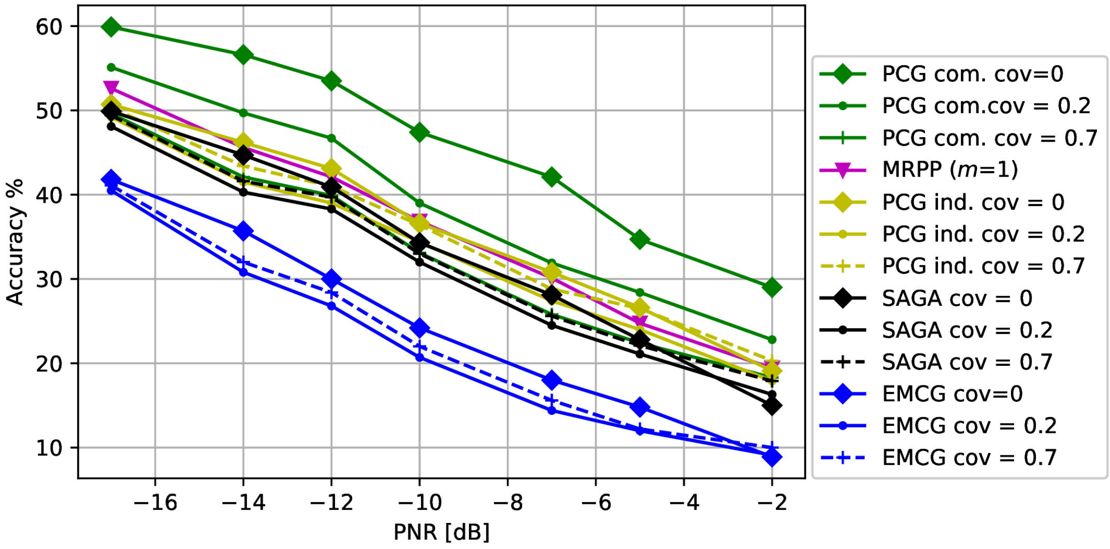
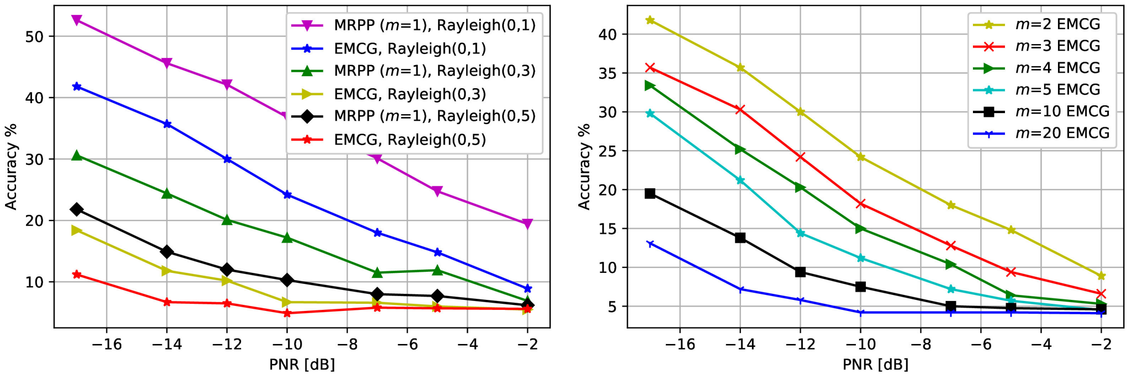


Figure 3.5: Classifier accuracy with respect to different covariances of channels between antennas.

correlation between the channels and investigate various attacks of an adversary with two antennas under different covariance levels. Results are shown in Fig. 3.5. We observe that as the covariance between the antennas increases, the performance of the PCG attack with common target increases significantly where it is comparable to the SAGA attack and even outperforms the PCG attack with independent targets. Note that the PCG attack with independent targets outperforms the PCG attack with common target when the channels are independent as shown in Fig. 3.5. In contrast, we see that other attack schemes are not significantly affected by the covariance. Further, we observe that even if the covariance is increased to 0.7, the attack performance slightly decreases compared to when the covariance is 0.2 in the EMCG attack, the PCG attack with independent targets, and the SAGA attack.

Assuming again independent channels from adversary antennas to the receiver, the classifier accuracy is shown in Fig. 3.6(a) when we vary the channel variance.



(a)

(b)

Figure 3.6: (a) Classifier accuracy with respect to different Rayleigh fading variances. (b) Classifier accuracy with different number of antennas at the adversary.

The classifier accuracy drops as the channel variance increases for all cases due to the increased uncertainty induced by the increased channel gain from the adversary to the receiver. Further, the performance ratio between MRPP and EMCG attacks increases as the channel variance increases. We also observe that as the PNR increases, the gap between MRPP and EMCG attacks decreases except for the case when the channel variance is 1.

Finally, we evaluate the attack performance of the adversary with different number of antennas m for the EMCG attack. Results are shown in Fig. 3.6(b) when the variance of channels is 1. The classifier accuracy decreases as m increases due to the increased channel diversity available to the adversary to exploit. Moreover, as the PNR increases, the performance gap between attacks launched with different m decreases suggesting that an increase of m in the high PNR region is not as effective as in the low PNR region.

3.5 Conclusion

We considered a wireless communication system where a DL-based signal classifier is used at the receiver to classify signals transmitted from the transmitter to their modulation types and showed that different methods to craft adversarial perturbations can be used to exploit multiple antennas at the adversary. We showed that just adding more antennas at the adversary does not always improve the attack. Thus, it is important to carefully allocate power among antennas, determine the adversarial perturbation for each antenna, and exploit channel diversity to select which antenna to transmit. In this context, the proposed EMCG attack significantly outperforms other attacks and effectively uses multiple antennas to evade the target classifier over-the-air. Next, we showed that the attack performance holds for different conditions of channels from the adversary antennas to the receiver and significantly improves by increasing the number antennas at the adversary.

CHAPTER 4

Adversarial Machine Learning for NextG Covert Communications Using Multiple Antennas

4.1 Introduction

In Chapter 2 and 3, we consider DL-based classifiers which are used at the receiver where the adversary tries to manipulate the target classifier by transmitting adversarial perturbations. However, there exists a possibility that adversaries also use DL-based classifiers for adversarial purposes such as eavesdropping. *Covert communications* has been studied to hide information in noise where the main goal has been to reduce the signal-to-noise ratio (SNR) at the eavesdropper [67,68]. A fundamental bound has been demonstrated on the total transmit power over a given number of channel users while maintaining covert communications, generally known as the square-root law [69]; see also [70] for related work. Motivated by this observation, in this chapter, we study covert communications from an *AML* point of view.

We consider an eavesdropper with a DL-based classifier to detect an ongoing transmission where this classifier achieves high accuracy for distinguishing the received signals from noise. We introduce a *CJ* that has been extensively used in the physical layer security literature [55–57]. In this chapter, the CJ transmits signals over the air at the same time as the transmitter with the purpose of fooling the eavesdropper's classifier for covert communications. These signals from CJ corresponds to an evasion attack (or adversarial attack) in AML where evasion attacks have been used to manipulate wireless signal classification (in particular, modulation classification) [20, 21, 23, 24, 27, 59, 66, 71–77], spectrum sensing [40], autoencoder communications [51], initial access [78], channel estimation [79], and power control [80]. In this chapter, the adversarial attack is used as a means of covert communications to prevent an eavesdropper from distinguishing an ongoing transmission from noise.

We use the CJ as the source of adversarial perturbation to manipulate the classifier at an eavesdropper into making classification errors. While a perturbation with a high power level transmitted by the CJ can easily fool the classifier, it would also increase the interference and the bit error rate (BER) at the intended receiver to an unacceptable level. Therefore, an upper bound on the perturbation strength is imposed. A special case of our setting has been considered in [81], where the transmitter with a single antenna adds perturbations to its own signals to fool an eavesdropper with a modulation classifier while aiming to maintain its own communication performance. In this chapter, our focus is on covert communications aided by a CJ, whose position can further boost the impact on the eavesdropper to classify received signal as noise while reducing the impact on the BER performance. Note that we only consider fooling a classifier into misclassifying a signal as noise since it is typically more demanding. Further, we extend the analysis to the use of *multiple antennas* at the CJ to generate multiple concurrent perturbations over different channel effects (subject to a total power budget) for better covert communications.

4.2 System Model

We consider a wireless system that consists of a transmitter, a receiver, a CJ, and an eavesdropper as shown in Fig. 4.1. The transmitter sends p complex symbols consecutively in time, $\boldsymbol{x} \in \mathbb{C}^p$, by mapping a binary input sequence $\boldsymbol{m} \in \{0,1\}^l$. Specifically, $\boldsymbol{x} = g_s(\boldsymbol{m})$, where $g_s : \{0,1\}^l \to \mathbb{C}^p$ and s represents the modulation type of the transmitter. Then, the transmitter's signal received at node j (either the receiver r or the eavesdropper e) is given by

$$\boldsymbol{r}_{tj} = \mathbf{H}_{tj} g_s(\boldsymbol{m}) + \boldsymbol{n}_{tj} = \mathbf{H}_{tj}\boldsymbol{x} + \boldsymbol{n}_{tj}, \quad j \in \{r, e\}, \tag{4.1}$$

where $\mathbf{H}_{tj} = \text{diag}\{h_{tj,1}, \cdots, h_{tj,p}\} \in \mathbb{C}^{p\times p}$ and $\boldsymbol{n}_{tj} \in \mathbb{C}^p$ are the channel and complex Gaussian noise from the transmitter to node j, respectively. Upon receiving the signal $\boldsymbol{r}_{tr}$, the receiver decodes the message with the BER given by

$$\mathrm{P}_e(\boldsymbol{m}, \boldsymbol{r}_{tr}) = \frac{1}{l}\sum_{i=1}^{l} \mathbb{I}\{m_i \neq \hat{m}_i\}, \tag{4.2}$$

where $\hat{m}_i$ is a decoded bit and $\mathbb{I}\{\cdot\}$ is an indicator function.

The eavesdropper tries to detect the existence of wireless transmission using a pre-trained DL-based classifier, namely a DNN, $f(.,\boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}^2$, where $\boldsymbol{\theta}$ is the set of DNN parameters and $\mathcal{X} \subset \mathbb{C}^p$. An input $\boldsymbol{x} \in \mathcal{X}$ is assigned a label $\hat{l}(\boldsymbol{x}, \boldsymbol{\theta}) = \arg\max_k f_k(\boldsymbol{x}, \boldsymbol{\theta})$, where $f_k(\boldsymbol{x}, \boldsymbol{\theta})$ is the output of a classifier f corresponding to the

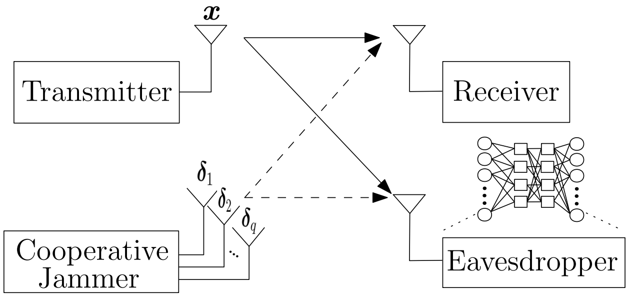


Figure 4.1: Wireless communication system with a transmitter, a receiver, a cooperative jammer, and an eavesdropper.

kth class.

To make communications between the transmitter and its receiver covert, the CJ with q antennas transmits perturbation signals $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \cdots, \boldsymbol{\delta}_q \in \mathbb{C}^p$, where the ith antenna transmits $\boldsymbol{\delta}_i$, to cause misclassification at the eavesdropper by changing the label of the received signal $\boldsymbol{r}_{te}$ from *signal* to *noise*. Thus, if the transmitter transmits $\boldsymbol{x}$, the received signal at node j is given by

$$\boldsymbol{r}'_{tj}(\boldsymbol{\delta}_1, \cdots, \boldsymbol{\delta}_q) = \mathbf{H}_{tj}\boldsymbol{x} + \sum_{i=1}^{q} \mathbf{H}_{c_ij}\boldsymbol{\delta}_i + \boldsymbol{n}_{tj}, \quad j \in \{r, e\}, \tag{4.3}$$

where $\mathbf{H}_{c_ij} = \text{diag}\{h_{c_ij,1}, \cdot\cdot\cdot, h_{c_ij,p}\} \in \mathbb{C}^{p \times p}$ is the channel from the ith antenna of the CJ to node j.

Since the perturbation signals from the CJ not only create interference at the eavesdropper, but also at the receiver, the CJ determines its signals $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \cdots, \boldsymbol{\delta}_q$ to cause misclassification at the eavesdropper using a fixed power budget P_{max} that also limits the BER at the receiver. Formally, the CJ first determines $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \cdots, \boldsymbol{\delta}_q$

by solving the following optimization problem:

$$\begin{aligned}\arg\min_{\boldsymbol{\delta}_i} \quad & \sum_{i=1}^{q} ||\boldsymbol{\delta}_i||_2^2 \\ s.t. \quad & \hat{l}(\boldsymbol{r}_{te}, \boldsymbol{\theta}) \neq \hat{l}(\boldsymbol{r}'_{te}(\boldsymbol{\delta}_1, \cdots, \boldsymbol{\delta}_q), \boldsymbol{\theta}) \\ & \sum_{i=1}^{q} ||\boldsymbol{\delta}_i||_2^2 \leq P_{max}. \end{aligned} \tag{4.4}$$

The solution $\boldsymbol{\delta}_i^*$ to (4.4) results in a BER, $\mathrm{P}_e(\boldsymbol{m}, \boldsymbol{r}'_{tr}(\boldsymbol{\delta}_i^*))$, at the receiver that can be bounded to a target level by selecting P_{max} accordingly. Since solving (4.4) is difficult, different methods have been proposed in computer vision to approximate the adversarial perturbations such as the fast gradient method (FGM) [58]. The FGM is computationally efficient for crafting adversarial attacks by linearizing the loss function, $L(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$, of the DNN classifier in a neighborhood of $\boldsymbol{x}$ where $\boldsymbol{y}$ is the label vector. This linearized function is used for optimization. In this chapter, we consider a *targeted attack*, where the perturbation of the CJ aims to decrease the loss function of the label *noise* and cause a specific misclassification, from *signal* to *noise*, at the eavesdropper even though there is an actual transmission. We approach the problem from an AML point of view and aim to fool a target classifier, which is equivalent to hiding communications in noise from a wireless communications perspective. While designing the attack, we constrain the BER at the receiver to stay below a certain level while satisfying the power constraint at the CJ, as stated in the constraints of the (4.4). We assume that the CJ collaborates with the transmitter and knows the transmitted signal from the transmitter.

4.3 Adversarial Perturbation for the CJ

In this section, we design the white-box perturbation for the CJ using a targeted FGM to solve (4.4). We first assume that the CJ has one antenna, $q = 1$. We will relax the assumption in Section 4.4. For the targeted attack, the CJ minimizes $L(\boldsymbol{\theta}, \boldsymbol{r}'_{te}(\boldsymbol{\delta}), \boldsymbol{y}^{target})$ with respect to $\boldsymbol{\delta}$ where $\boldsymbol{y}^{target}$ is the one-hot-encoded desired target class. We fix $\boldsymbol{y}^{target}$ as *noise* label since the CJ always tries to add perturbation to fool the eavesdropper into misclassifying a received signal as noise. We use FGM to linearize the loss function as $L(\boldsymbol{\theta}, \boldsymbol{r}'_{te}(\boldsymbol{\delta}), \boldsymbol{y}^{target}) \approx L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target}) + (\mathbf{H}_{ce}\boldsymbol{\delta})^T \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target})$ and then minimize it by setting $\mathbf{H}_{ce}\boldsymbol{\delta} = -\alpha \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target})$, where α is a scaling factor to constrain the adversarial perturbation power to P_{max}. The details of determining the CJ's perturbation signal are presented in Algorithm 7. After we obtain the $\boldsymbol{\delta}$ that causes misclassification at the eavesdropper and satisfies the power constraint, we check the BER at the receiver. The perturbation power can further be adjusted to meet a target BER level. Specifically, if the BER level at the receiver is more important than fooling the eavesdropper, we can decrease the adversarial perturbation power. On the other hand, if fooling the eavesdropper is the priority, we can increase the adversarial perturbation power.

Algorithm 7: Generating the perturbation of the CJ

```
Inputs: input $\boldsymbol{r}_{te}$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$, and $L(\boldsymbol{\theta}, \cdot, \cdot)$.
Initialize: $\varepsilon \leftarrow 0, \varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0$.
$\boldsymbol{\delta}_{norm} = \frac{\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target})}{(||\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target})||_2)}$.
while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
    $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
    $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{r}_{te} - \varepsilon_{avg}\boldsymbol{\delta}_{norm}$
    if $\hat{l}(\boldsymbol{x}_{adv}) == noise$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
    else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
end
$\varepsilon^* = \varepsilon_{max}$, $\boldsymbol{\delta}^{jam} = -\varepsilon^* \boldsymbol{\delta}_{norm}$
```

4.4 Adversarial Perturbations Using Multiple Antennas at the CJ

In this section, we present different methods to utilize q antennas at the CJ to improve the performance of the adversarial attack against the eavesdropper. Note that the adversary can allocate power differently to each antenna and increase the channel diversity by using multiple antennas. We apply the attacks from Section 3.3 with some modifications to fool the eavesdropper with multiple antennas.

4.4.1 Single-Antenna Genie-Aided (SAGA) Attack

We first begin with an attack where the CJ allocates all the power to only one antenna for the entire symbol block of an input to the classifier at the eavesdropper. In this attack, we assume that the CJ is aided by a genie and thus knows in advance the best antenna out of q antennas that causes a misclassification. Then, the genie-aided CJ puts all the power to that one specific antenna to transmit the adversarial perturbation against the eavesdropper.

Algorithm 8: PCG attack at the CJ

```
Inputs: input $\boldsymbol{r}_{te}$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$, and model
 of the classifier $L(\boldsymbol{\theta}, \cdot, \cdot)$.
Initialize: $w_i = \frac{\|\mathbf{h}_{c_ie}\|_2}{\sum_{j=1}^{q} \|\mathbf{h}_{c_je}\|_2}, i = 1, \cdots, q$.
$\varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0$.
for $i = 1$ to $q$ do
    $\boldsymbol{\delta}_i = \frac{\mathbf{H}^*_{c_ie} \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target})}{(\|\mathbf{H}^*_{c_ie} \nabla_{\boldsymbol{x}} L(\boldsymbol{\theta}, \boldsymbol{r}_{te}, \boldsymbol{y}^{target})\|_2)}$
end
while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
    $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
    $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x} - \varepsilon_{avg} \sum_{i=1}^{q} w_i \mathbf{H}_{c_ie} \boldsymbol{\delta}_i$
    if $\hat{l}(\boldsymbol{x}_{adv}) == noise$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
    else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
end
$\varepsilon^* = \varepsilon_{max}$
$\boldsymbol{\delta}_i = w_i \varepsilon^* \boldsymbol{\delta}_i$ for $\forall i$
```

4.4.2 Proportional to Channel Gain (PCG) Attack

To exploit the channel with the better channel gain, the CJ allocates more power to better channels. Specifically, the power allocation for the ith antenna is proportional to the channel gain $\|\mathbf{h}_{c_ie}\|_2$, where $\mathbf{h}_{c_ie} = [h_{c_ie,1}, \cdot \cdot \cdot, h_{c_ie,p}]^T$, using weight $w_i = \frac{\|\mathbf{h}_{c_ie}\|_2}{\sum_{j=1}^{q} \|\mathbf{h}_{c_je}\|_2}, i = 1, \cdots, q$. The adversarial perturbation that is transmitted by each antenna is generated using the MRPP attack as before and transmitted with the power allocated to each antenna. The detailed algorithm is presented in Algorithm 8.

4.4.3 Inversely Proportional to Channel Gain (IPCG) Attack

In contrast to the PCG attack, the CJ allocates more power to weak channels to compensate for the loss over the weak channels, i.e., inversely proportional to the channel gain. The perturbations that are transmitted by each antenna are generated using the MRPP attack and the power for each antenna is determined to be inversely proportional to the channel gain. The algorithm is the same as Algorithm 8 except that w_i changes to be inversely proportional to the channel, i.e., $w_i = \frac{1}{\|\mathbf{h}_{c_i e}\|_2} \frac{1}{\sum_{j=1}^{q} \frac{1}{\|\mathbf{h}_{c_j e}\|_2}}, i = 1, \cdots, q.$

4.4.4 Elementwise Maximum Channel Gain (EMCG) Attack

Unlike the previous attacks that considered the channel gain of the channel vector with dimension $p \times 1$ as a way to allocate power among antennas, the EMCG attack considers the channel gain for each time instance to fully utilize the channel diversity. First, the CJ compares the channel gains elementwise and selects one antenna that has the largest channel gain at each instance. Specifically, the CJ finds and transmits with the antenna $j^* = \arg \max\limits_{j=1,\cdots,q} \{\|h_{ar_j,t}\|_2\}$ that has the largest channel gain at instance t. Furthermore, a virtual channel $h_{vir,t}$ at instance t is defined as the channel with the largest channel gain among antennas which is $h_{ar_{j^*},t}$. Then, the adversary generates the perturbation $\boldsymbol{\delta}^{vir}$ with respect to $\boldsymbol{h}_{vir} = [h_{vir,1}, \cdots, h_{vir,p}]^T$ using the MRPP attack and transmits each element of $\boldsymbol{\delta}^{vir}$ with the antenna that has been selected previously. The details are provided in Algorithm 9.

4.5 Simulation Results

We analyzed the success of covertness achieved by CJ's perturbation at the eavesdropper and the corresponding effect on the BER at the receiver. We first assumed

Algorithm 9: EMCG attack at the CJ

```
Inputs: input $\boldsymbol{r}_{te}$, desired accuracy $\varepsilon_{acc}$, power constraint $P_{max}$, and model
  of the classifier $L(\boldsymbol{\theta}, \cdot, \cdot)$.
Initialize: $k \leftarrow \mathbf{0}^{p\times 1}$, $\boldsymbol{\delta}_i \leftarrow \mathbf{0}^{p\times 1}$ for $\forall i$.
for $i = 1$ to $p$ do
    $h_{vir,i} = \max\{\|h_{c_1 e,i}\|_2, \cdots, \|h_{c_m e,i}\|_2\}$
    $k[i] = \arg\max\{\|h_{c_1 e,i}\|_2, \cdots, \|h_{c_m e,i}\|_2\}$
end
Virtual channel : $\mathbf{H}_{vir} = \mathrm{diag}\{h_{vir,1}, \cdots, h_{vir,p}\}$
$\varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0$
$\boldsymbol{\delta} = \frac{\mathbf{H}^*_{vir}\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{r}_{te},\boldsymbol{y}^{target})}{(\|\mathbf{H}^*_{vir}\nabla_{\boldsymbol{x}} L(\boldsymbol{\theta},\boldsymbol{r}_{te},\boldsymbol{y}^{target})\|_2)}$
while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
    $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
    $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x} - \varepsilon_{avg}\mathbf{H}_{vir}\boldsymbol{\delta}$
    if $\hat{l}(\boldsymbol{x}_{adv}) == noise$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
    else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
end
$\varepsilon^* = \varepsilon_{max}$
$\boldsymbol{\delta}^{vir} = \varepsilon^*\boldsymbol{\delta}$
for $i = 1$ to $p$ do
    $\boldsymbol{\delta}_{k[i]} = \boldsymbol{\delta}^{vir}[i]$
end
Transmit $\boldsymbol{\delta}_i$, $i = 1, \cdots, q$
```

that the CJ only had one antenna to analyze the impact of the CJ on the eavesdropper. Then, we increased the number of antennas at the CJ to observe the performance when multiple antennas are used with different methods. We compared this perturbation with random Gaussian noise transmitted by the CJ. Furthermore, we changed the location of the CJ to investigate the effects of topology and channel.

4.5.1 Simulation Settings

We assumed that the binary source data were generated independently and uniformly at the receiver. The classifier at the eavesdropper was a convolutional neural network (CNN). The input to the CNN was of two dimensions $(2, 16)$ corresponding to 16 in-phase/quadrature (I/Q) data samples. The CNN consisted of a convolutional layer with kernel size $(1,3)$, a hidden layer with dropout rate 0.1, a rectified linear unit (ReLU) activation function at the convolutional and hidden layers and a softmax activation function at the output layer that provides the label *signal* or *noise*. We applied a backpropagation algorithm with the Adam optimizer to train the CNN using cross-entropy as the loss function. The CNN was implemented in Keras with the TensorFlow backend. We assumed that the eavesdropper already knew the signal type that was used at the transmitter. Thus, the classifier at the eavesdropper was only trained with two labels, *signal* and *noise*. For each signal type, we trained a separate classifier using different datasets, where 20,000 symbols were generated and split into blocks of 16 I/Q symbols. The channel between the nodes had path-loss effects and Rayleigh fading such that the channel gain from node i to node j was $h_{ij} = \left(\frac{d_0}{d_{ij}}\right)^{\gamma} h_{i,j}$, where d_{ij} is the distance from node i to j, d_0 is the reference distance, $h_{i,j}$ is Rayleigh fading between node i to j, and γ is the path loss exponent. We set $d_0 = 1$ and $\gamma = 2.8$ throughout the simulations. Note that there was only a path loss component in the channels for the simulations with CJ for the case of one antenna.

We used the perturbation-to-noise ratio (PNR) metric from [59] that captures the relative perturbation power at the CJ with respect to the noise and measured how the increase in the PNR affected the accuracy of the classifier at the eavesdropper. As the PNR increases, the perturbation generated by the CJ is more likely to be detected by the eavesdropper and increases the BER at the receiver.

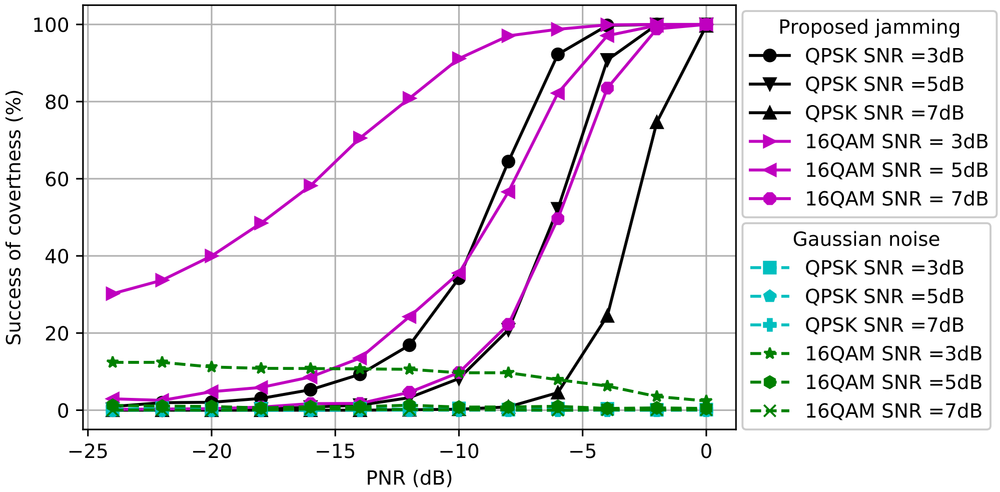


Figure 4.2: Success of covertness at the eavesdropper when $d_{ce} = d_{cr} = 1$.

4.5.2 Performance Evaluation of CJ With One Antenna for Signals With Different Modulations

We first assumed that the CJ only had one antenna, $q = 1$, and aimed to hide signals with a fixed modulation scheme, namely QPSK or 16QAM, used by the transmitter using Algorithm 7. Note that we used only Algorithm 7 since the CJ only had a single antenna. The first topology that we considered was $d_{cr} = d_{ce} = 1$. In Fig. 4.2, we show how the perturbation signal generated by the CJ affects the classifier at the eavesdropper. The x-axis is the PNR (measured in dB) and the y-axis is the success of covertness (measured in percentage) that indicates the success of making wireless communications covert, namely the likelihood that the eavesdropper classifies a signal plus perturbation as noise. We observe that as the SNR of the signal increases, the CJ needs more perturbation power to cause misclassification at the eavesdropper. Furthermore, the 16QAM-modulated signal

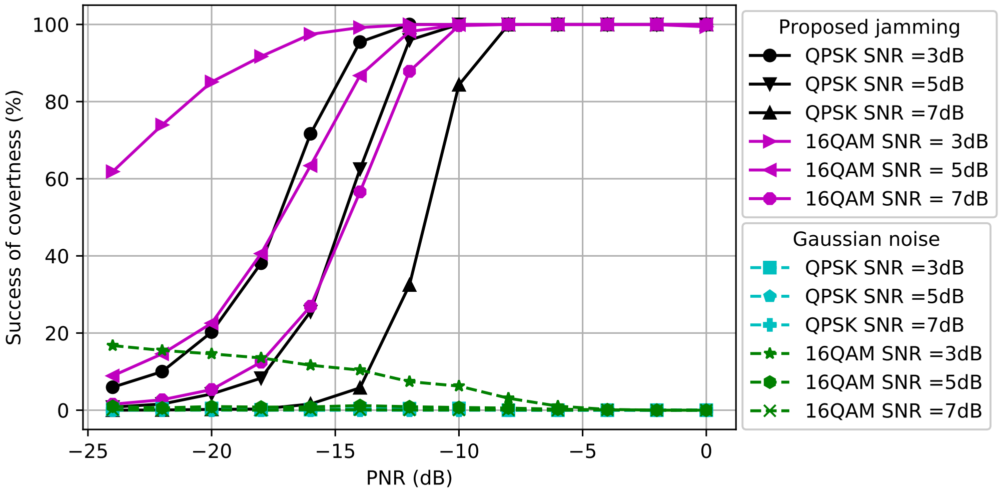


Figure 4.3: Success of covertness at the eavesdropper when $d_{ce} = 0.5$ and $d_{cr} = 1.5$.

is more susceptible to adversarial perturbation than the QPSK-modulated signal, since it is more difficult to distinguish the 16QAM-modulated signal from the noise for the same SNR. Furthermore, we observe that the success of covertness suddenly increases after some PNR value for both modulation types. On the contrary, the Gaussian noise based perturbation has a negligible effect on the classifier for all SNR values. We further observe that the Gaussian noise with more power decreases the success of covertness when the SNR of 16-QAM modulated signal is 3 dB. The reason is the Gaussian noise strengthens the noise which makes the received signal at the eavesdropper resemble the strength of the signal, thus the classifier at the eavesdropper classifies the received signal as signal.

In Fig. 4.3, we consider $d_{cr}, = 1.5$ and $d_{ce} = 0.5$ (namely, the distance between the CJ and the receiver is increased and the distance between the CJ and the eavesdropper is decreased compared to Fig. 4.2). As the SNR of the signal increases, the CJ requires more power to cause misclassification at the eavesdropper, as we

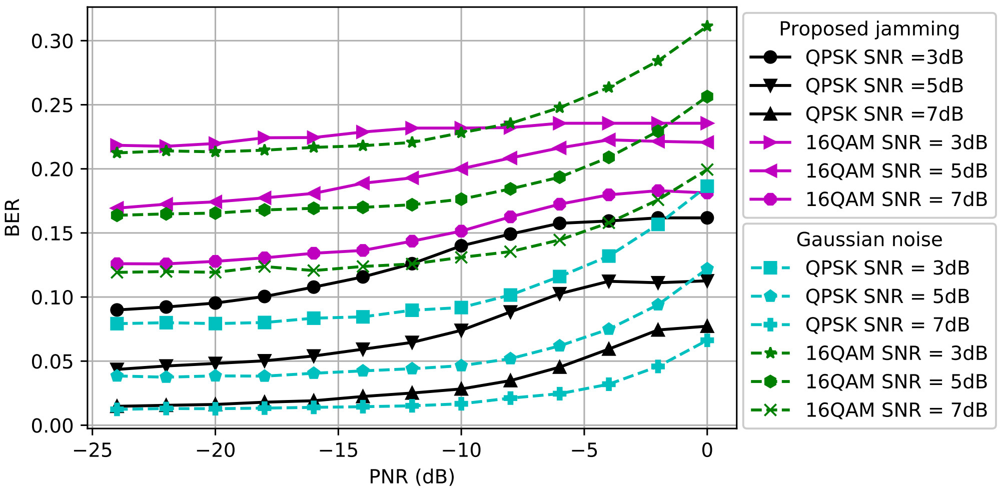


Figure 4.4: BER at the receiver when $d_{ce} = d_{cr} = 1$.

also observed in Fig. 4.2. Due to the reduced path loss effect between the CJ and the eavesdropper, less power is required to cause misclassification compared to Fig. 4.2. This result motivates the use of AML instead of conventional jamming (e.g., [82]) to attack an eavesdropper.

4.5.3 Reliability of Communications

The BER performance at the receiver for different modulation types and SNR values is compared in Fig. 4.4 when $d_{cr} = d_{ce} = 1$. We observe that the BER of the 16QAM-modulated signals is more susceptible to the adversarial perturbation signal than the BER of QPSK-modulated signals. The reason is that since the 16QAM transmits more bits than the QPSK per symbol, the distances between constellation points are smaller, which leads to a larger BER for a given SNR. Moreover, as the SNR increases, the average BER decreases as expected. For the CJ with the proposed adversarial perturbation, we observe that the BER curve saturates after

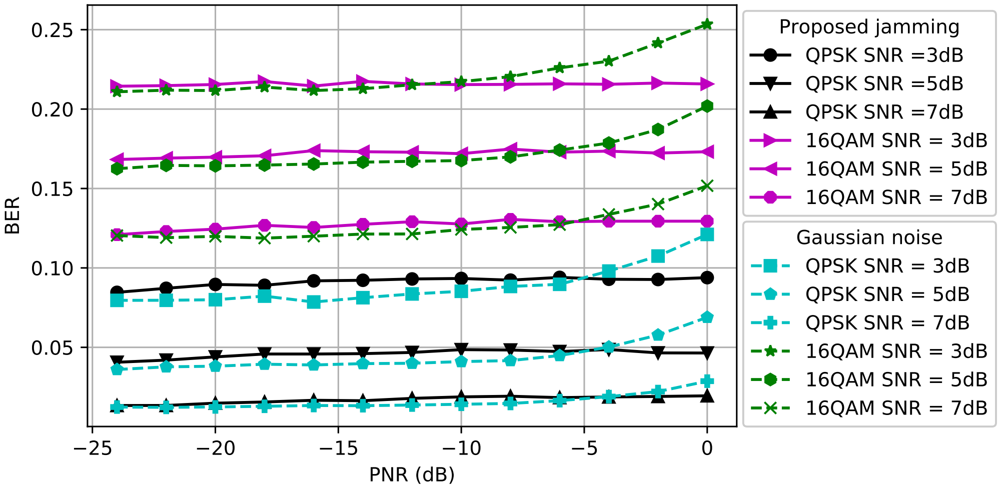


Figure 4.5: BER at the receiver when $d_{ce} = 0.5$ and $d_{cr} = 1.5$.

some PNR value because the successful perturbation signal can be generated using less power than the maximum power that the CJ can use. Fig. 4.4 can be used as a guideline to determine the maximum PNR to satisfy the BER requirement at the receiver. For example, to meet the target BER of 0.15 for a QPSK-modulated signal, the PNR is selected to be at most -8dB when the SNR is 3dB and the resulting success of covertness is 65%. Furthermore, we observe that the Gaussian noise based perturbation results in a lower BER than the adversarial perturbation in the low PNR regime. However, the BER gap between these two CJ schemes decreases when the PNR increases, and the adversarial perturbation results in a smaller BER in the high PNR region.

The BER performance at the receiver for different modulation types and SNR values is compared in Fig. 4.5 when $d_{cr} = 1.5$ and $d_{ce} = 0.5$. We observe that the BER gap between the Gaussian noise and adversarial perturbation for the same SNR value decreases due to the increased path loss effect between the CJ and the receiver. Thus, the CJ can create a perturbation signal that causes misclassification with higher success without increasing the BER further if the location of the CJ is closer to the eavesdropper. This result motivates the control of the CJ positions to fool a target classifier while protecting the BER performance of the intended receiver.

4.5.4 Performance Evaluation for 5G Communications

As a full-fledged waveform to hide, we considered the 5G physical layer communications where a 5G user equipment (UE) transmits a 5G uplink signal to a base station (gNodeB) in the presence of the perturbation from the CJ. MATLAB's 5G toolbox was used to generate 5G signals that included the transport (uplink shared channel, UL-SCH) and physical channel. The transport block was segmented after the cyclic redundancy check (CRC) addition and low-density parity-check (LDPC) coding was used as forward error correction. The output codewords were QPSK-modulated as an example. Next, the generated resource grid was OFDM-modulated with inverse fast Fourier transform and cyclic prefix (CP) addition operations where the subcarrier spacing was 15 kHz. The target code rate was set to $\frac{820}{1024}$ and the output I/Q samples were stored after the signal passed through the channel. The eavesdropper attempted to distinguish the received signals from noise, whereas the receiver attempted to decode the received signals by removing the CP and performing FFT, channel equalization, QPSK demodulation, LDPC, and CRC decoding operations.

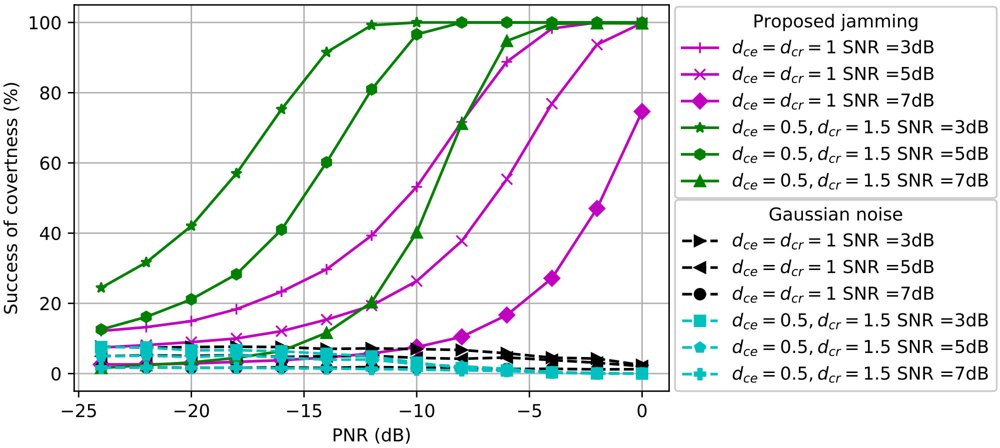


Figure 4.6: 5G communications covertness performance at the eavesdropper.

4.5.4.1 Covertness of Communications

The success of covertness for 5G communications is considered in Fig. 4.6. As in the previous figures for QPSK-modulated signals and 16QAM-modulated signals, the proposed perturbation outperforms the Gaussian noise significantly in the high-PNR region for 5G signals. Furthermore, we observe that more power is needed for the CJ to fool the classifier at the eavesdropper when the distance between the CJ and the eavesdropper increases.

4.5.4.2 Reliability of Communications

The BER for 5G communications is shown in Fig. 4.7. When $d_{ce} = d_{cr} = 1$ and the SNR is 5 dB, the Gaussian noise based perturbation has a higher BER performance compared to the proposed perturbation and a similar result is also observed for other SNR values. Note that the adversarial perturbation by the CJ not only increases the success of covertness, but also has less effect on the BER performance of the receiver compared to the Gaussian noise based perturbation for 5G communication signals.

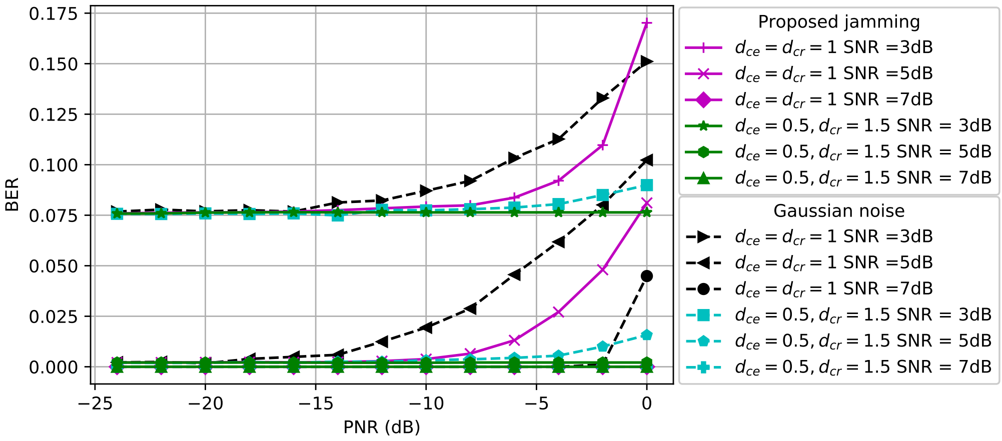
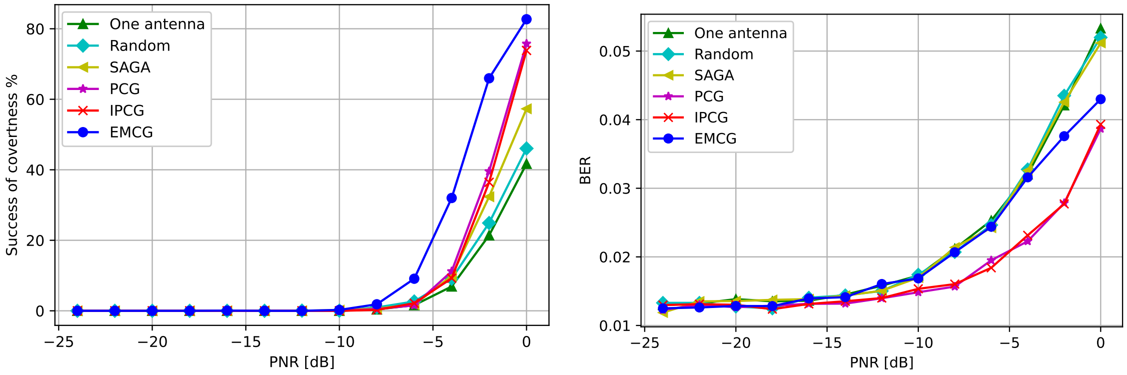


Figure 4.7: 5G communications BER performance at the receiver.

We further observe that the Gaussian noise based perturbation results in a higher BER than the proposed adversarial perturbation when $d_{ce} = 0.5$ and $d_{cr} = 1.5$.

4.5.5 Performance Evaluation of CJ With Multiple Antennas

Next, we analyzed the performance of the CJ with multiple antennas when a QPSK-modulated signal was used at the transmitter. Note that the channel between the CJ and the receiver and the channel between the CJ and the eavesdropper had Rayleigh fading. Note that $h_{i,j} \sim \text{Rayleigh}(0, 1)$ if specified otherwise. Fig. 4.8(a) presents the success of covertness when the CJ transmits an adversarial perturbation with $q = 2$ antennas using the different attack methods introduced in Section 4.4. We observe that all different methods using multiple antennas outperform the attack generated by the CJ with one antenna. Furthermore, randomly selecting one antenna at the CJ performs worst among attacks using multiple antennas and the performance of the IPCG attack is similar to the performance of the PCG attack. Moreover, the EMCG attack outperforms other attacks by fully utilizing the channel diversity.



(a)

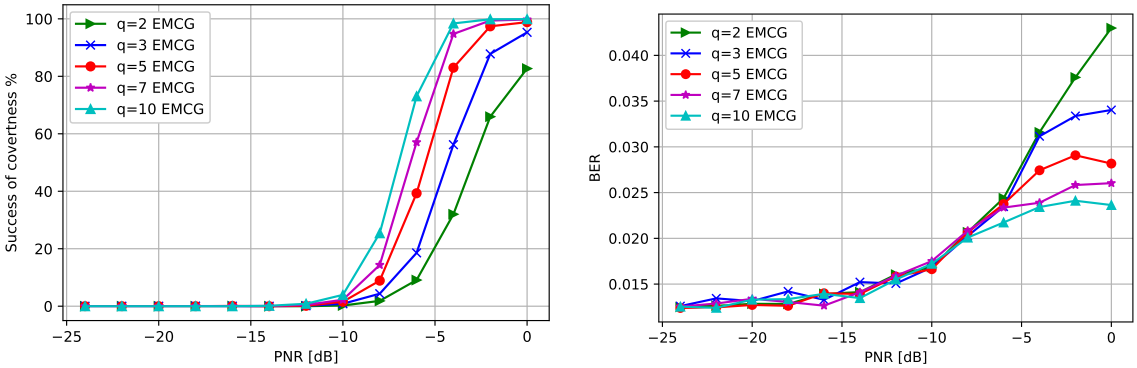
(b)

Figure 4.8: Performance when CJ has $q = 2$ antennas: (a) success of covertness and (b) BER at the receiver.

Fig. 4.8(b) presents the BER performance of different attack methods. We observe that the CJ using one antenna gives the largest BER whereas the PCG and IPCG attacks give the smallest BER. Furthermore, the EMCG attack gives a moderate BER increase while successfully making communications covert.

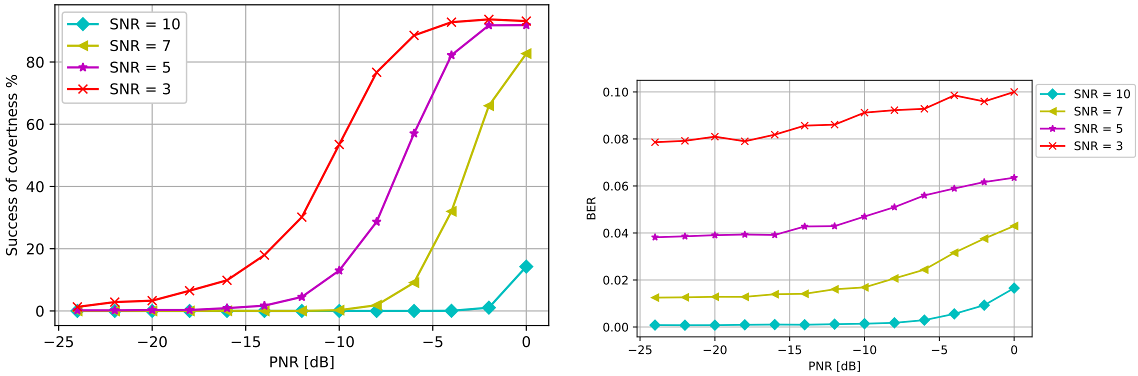
The performance of the CJ with different number of antennas is presented in Fig. 4.9(a). As the number of the antennas at the CJ increases, the success of covertness also increases suggesting that using more antenna at the CJ helps the covertness of communications. Furthermore, the BER decreases when more antennas are used at the CJ as we can see from Fig. 4.9(b). Therefore, using more antennas at the CJ is always beneficial for communications in terms of covertness and BER when the EMCG attack is used at the CJ.

Next, we varied the SNR levels to analyze how the SNR affected the covertness and the BER in Fig. 4.10. As expected, the CJ needs a higher PNR to fool the eavesdropper when the SNR is high. Furthermore, we observe that the BER slightly increases when the PNR increases and the BER is higher for a lower SNR.



(a) (b)

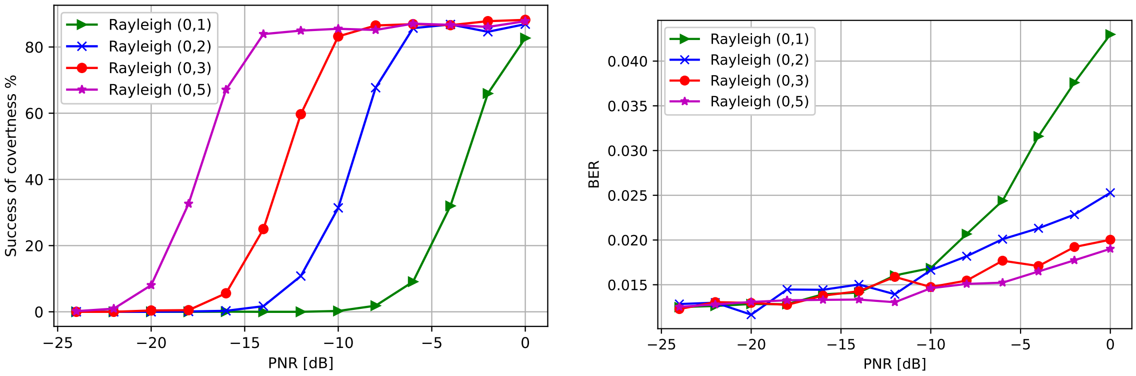
Figure 4.9: Performance with different number of antennas at the CJ: (a) success of covertness and (b) BER at the receiver.



(a) (b)

Figure 4.10: Performance with respect to different SNR levels: (a) success of covertness and (b) BER at the receiver.

Finally, we increased the variance of the Rayleigh fading between the CJ and the eavesdropper to analyze the effect of the channel on the covertness of communications. In Fig. 4.11(a), we observe that a lower PNR is needed to fool the eavesdropper when the variance of the Rayleigh fading is high. Furthermore, as a consequence of using a lower PNR at the CJ, a higher variance of Rayleigh fading results in a lower BER at the receiver.



(a)

(b)

Figure 4.11: Performance with respect to different Rayleigh fading variances: (a) success of covertness and (b) BER at the receiver.

4.6 Conclusion

We considered a wireless communications system in which a CJ with multiple antennas transmits perturbation signals to fool a DL-based classifier at the eavesdropper into classifying the ongoing transmissions as noise. Following the AML approach, the CJ was designed to generate the perturbation signal with different methods. For both basic modulated signals and sophisticated 5G signals, we showed that the CJ could generate a perturbation signal that caused misclassification at the eavesdropper (from *signal* to *noise*) with high success, while the BER at the receiver was only slightly affected. Furthermore, we showed that by adding more antennas at the CJ always improved the attack performance and lowered the BER when the EMCG attack was used. These results demonstrate that wireless communications can be successfully kept covert when multiple antennas are used at the CJ by allocating the transmit power efficiently.

CHAPTER 5

Channel Effects on Surrogate Models of Adversarial Attacks against Wireless Signal Classifiers

5.1 Introduction

For adversarial attacks on wireless signal classification, as we have seen in Chapters 2, 3, and 4, a common assumption has been made that the inputs to the target model and the surrogate model have the same distributions such that the adversarial attacks can be readily transferred from the surrogate model to the target model. However, practical wireless deployments require that the wireless adversary builds the surrogate model by observing the spectrum and collecting the training data over-the-air. Therefore, the surrogate model and the target model may not have the same input distributions due to the discrepancies in channels experienced by the adversary and the target receiver. Motivated by this observation, we study the performance of the adversarial perturbation using the surrogate model.

In this chapter, we consider a wireless communications system with a background emitter, a transmitter, and an adversary where the transmitter collects I/Q data to train its own DNN classifier to detect ongoing transmission of a background transmitter for channel access decision. In the meanwhile, the adversary builds a surrogate model of the transmitter's DNN classifier by observing the over-the-air signal. To build a surrogate model, the adversary collects the I/Q data, signals received from the background signal source, by exploiting the broadcast nature of transmissions and obtains the labels by listening to the potential signals from the transmitter.

Using the surrogate model, the adversary generates adversarial perturbations to enforce the missclassification at the transmitter's classifier. This setting may model a system where the background emitter is a primary, the transmitter is a secondary, and the adversary is trying to fool the secondary to transmit even though the channel is used by the primary. Compared to previous works, we make the realistic assumption that the adversarial attacks are generated using the surrogate model that is trained with different distribution of the training dataset relative to the transmitter's target model since the signals received by the adversary and the transmitter are different due to channel discrepancies. Different topologies of the adversary are considered to investigate how the difference in the distribution of the training datasets at the adversary changes the performance of the adversarial attack on the transmitter's classifier. Further, different ways to determine the transmit power at the adversary using the surrogate model are considered and compared. Finally, we relax the assumption about knowing the exact input at the transmitter and use the input at the adversary instead. Results show that the performance of adversarial attacks against a wireless signal classifier heavily relies on the reliability of surrogate model which depends on the difference in channels experienced by the

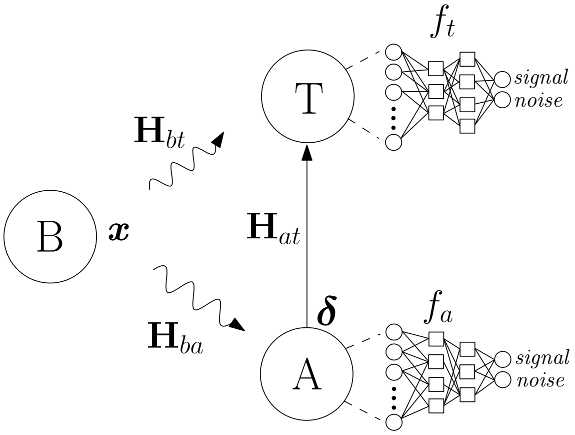


Figure 5.1: System model. ‘B’ stands for the background emitter, ‘T’ stands for the transmitter, and ‘A’ stands for the adversary.

adversary and the transmitter.

5.2 System Model

We consider a wireless communication system that consists of a background emitter, a transmitter, and an adversary as shown in Fig. 5.1. The background emitter transmits k complex symbols, $\boldsymbol{x} \in \mathbb{C}^k$, and node j (either the transmitter t or the adversary a) receives

$$\boldsymbol{r}_{bj} = \mathbf{H}_{bj}\boldsymbol{x} + \boldsymbol{n}_{bj}, \quad j \in \{t, a\}, \tag{5.1}$$

where $\mathbf{H}_{bj} = \text{diag}\{h_{bj,1}, \cdots, h_{bj,k}\} \in \mathbb{C}^{k \times k}$ and $\boldsymbol{n}_{bj} \in \mathbb{C}^k$ are the channel gains and complex Gaussian noise from the background emitter to node j, respectively.

Using $\boldsymbol{r}_{bt}$, the transmitter trains its DNN classifier, $f_t(.;\boldsymbol{\theta}_t) : \mathcal{X} \rightarrow \mathbb{R}^2$, to determine the existence of ongoing background transmission to utilize the idle bands, where $\boldsymbol{\theta}_t$ is the set of transmitter's DNN parameters and $\mathcal{X} \subset \mathbb{C}^k$. The input $\boldsymbol{r}_{bt}$

is assigned to the label $\hat{l}_t(\boldsymbol{r}_{bt}, \boldsymbol{\theta}_t) = \arg\max_q f_t^{(q)}(\boldsymbol{r}_{bt}, \boldsymbol{\theta}_t)$, where $f_t^{(q)}(\boldsymbol{r}_{bt}, \boldsymbol{\theta}_t)$ is the output of classifier f_t corresponding to the qth class.

Concurrently, the adversary tries to detect the background transmission based on $\boldsymbol{r}_{ba}$ using its own DNN classifier as the *surrogate model*, $f_a(.;\boldsymbol{\theta}_a) : \mathcal{X} \rightarrow \mathbb{R}^2$, where $\boldsymbol{\theta}_a$ is the set of the adversary's DNN parameters. The label corresponding to the input $\boldsymbol{r}_{ba}$ is defined as $\hat{l}_a(\boldsymbol{r}_{ba}, \boldsymbol{\theta}_a) = \arg\max_q f_a^{(q)}(\boldsymbol{r}_{ba}, \boldsymbol{\theta}_a)$ where $f_a^{(q)}(\boldsymbol{r}_{ba}, \boldsymbol{\theta}_a)$ is the output of classifier f_a corresponding to the qth class. If the label of the classifier is 'signal', the adversary transmits a perturbation $\boldsymbol{\delta} \in \mathbb{C}^k$ to change the label of the DNN classifier at the transmitter to 'noise'. Note that fooling the transmitter's classifier to perturb the label from 'noise' to 'signal' is much easier than 'signal' to 'noise', thus we only consider the latter case where the adversary aims to perturb the label from 'signal' to 'noise'.

If the background emitter transmits $\boldsymbol{x}$ and the adversary transmits $\boldsymbol{\delta}$, the received signal at the transmitter is

$$\boldsymbol{r}'_{bt}(\boldsymbol{\delta}) = \mathbf{H}_{bt}\boldsymbol{x} + \mathbf{H}_{at}\boldsymbol{\delta} + \boldsymbol{n}_{bt}, \tag{5.2}$$

where $\mathbf{H}_{at} = \text{diag}\{h_{at,1}, \cdots, h_{at,k}\} \in \mathbb{C}^{k \times k}$ is the channel gain from the adversary to the transmitter.

The adversary generates the adversarial perturbation $\boldsymbol{\delta}$ to cause misclassification at the transmitter for the input $\boldsymbol{r}_{bt}$ while satisfying the power budget P_{max} for the stealthiness of the attack. Thus, the adversary determines $\boldsymbol{\delta}$ by solving the following optimization problem:

$$\begin{aligned} \arg\min_{\boldsymbol{\delta}} \quad & ||\boldsymbol{\delta}||_2 \\ s.t. \quad & \hat{l}_t(\boldsymbol{r}_{bt}, \boldsymbol{\theta}_t) \neq \hat{l}_t(\boldsymbol{r}'_{bt}(\boldsymbol{\delta}), \boldsymbol{\theta}_t) \\ & ||\boldsymbol{\delta}||_2^2 \leq P_{max}. \end{aligned} \tag{5.3}$$

However, in reality, the adversary has no information about the classifier at the transmitter, f_t, and it is not possible to check whether an attack generated at the adversary satisfies the constraint $\hat{l}_t(\boldsymbol{r}_{bt}, \boldsymbol{\theta}_t) \neq \hat{l}_t(\boldsymbol{r}'_{bt}(\boldsymbol{\delta}), \boldsymbol{\theta}_t)$, or not. Therefore, we change this constraint by using the DNN classifier at the adversary, f_a, which can be thought of as a *surrogate model* for the transmitter's classifier. Now, the optimization problem to generate the adversarial perturbation at the adversary is

$$\begin{aligned} \arg\min_{\boldsymbol{\delta}} \quad & ||\boldsymbol{\delta}||_2 \\ s.t. \quad & \hat{l}_a(\boldsymbol{r}_{bt}, \boldsymbol{\theta}_a) \neq \hat{l}_a(\boldsymbol{r}'_{bt}(\boldsymbol{\delta}), \boldsymbol{\theta}_a) \\ & ||\boldsymbol{\delta}||_2^2 \leq P_{max}. \end{aligned} \tag{5.4}$$

As the non-linearity of the DNNs makes the solution $\boldsymbol{\delta}^*$ to (5.4) difficult to obtain, the adversarial perturbation is approximated. Fast gradient method (FGM) [58] is an efficient method to craft adversarial attacks by linearizing the loss function, $L_j(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$, of the jth node's DNN classifier in a neighborhood of input $\boldsymbol{x}$, where $\boldsymbol{y}$ is the label vector, and uses linearized loss function for the optimization. In this chapter, we consider a targeted attack, specifically the maximum received perturbation power (MRPP) attack in Chapter 2, where the adversary's perturbation aims to decrease the loss function of the class 'noise' and cause a specific misclassification, from 'signal' to 'noise', at the transmitter in the presence of background transmission.

5.2.1 Adversarial Attack Built Upon the Surrogate Model

In this section, we introduce how to craft an adversarial perturbation using the surrogate model of the adversary, f_a, to flip the label of the transmitter's classifier. To do so, the adversary first tries to detect the background transmission based on $\boldsymbol{r}_{ba}$ using its own DNN classifier, f_a. Since the adversary tries to change the label from 'signal' to 'noise', the adversary generates an attack if the label $\hat{l}_a(\boldsymbol{r}_{ba}, \boldsymbol{\theta}_a) = \arg\max_q f_a^{(q)}(\boldsymbol{r}_{ba}, \boldsymbol{\theta}_a) =$ 'signal'. Then, the adversary crafts the adversarial perturbation using the MRPP attack. Note that we assume that the adversary knows the channel $\mathbf{H}_{at}$ and the exact input at the transmitter, $\boldsymbol{r}_{bt}$, when creating the adversarial perturbation. We relax the assumption regarding knowing the exact input at the transmitter by using the input at the adversary, $\boldsymbol{r}_{ba}$, instead of $\boldsymbol{r}_{bt}$ later in this chapter.

Since the adversary has no information about the transmitter's classifier, it is hard for the adversary to determine the transmit power that is needed to change the label at the transmitter. Therefore, we consider two different ways to decide the transmit power at the adversary.

5.2.2 Adversarial Perturbation Using the Maximum Power P_{max}

Without knowing the transmitter's classifier, one simple option for the adversary is to use the maximum power for the adversarial perturbation. Therefore, the adversary simply transmits $\boldsymbol{\delta}^* = -P_{max} \frac{\nabla_{\boldsymbol{x}} \mathbf{H}_{at}^* L_a(\boldsymbol{\theta}_a, \boldsymbol{r}_{bt}, \boldsymbol{y}^{target})}{(||\nabla_{\boldsymbol{x}} \mathbf{H}_{at}^* L_a(\boldsymbol{\theta}_a, \boldsymbol{r}_{bt}, \boldsymbol{y}^{target})||_2)}$, where $\boldsymbol{y}^{target}$ is 'noise'.

5.2.3 Adversarial Perturbation Using the Surrogate Model of the Adversary

The transmit power at the adversary should be carefully determined to fool the transmitter's classifier. This has been done in [51, 66] by using the transmitter's classifier. However, the adversary has no information about the transmitter's classifier. One option for the adversary is to use its own classifier as a surrogate model for the transmitter's classifier and determine the transmit power based on its own classifier. The details of how to generate the adversarial perturbation are presented in Algorithm 10.

5.2.4 Adversarial Perturbation Using r_{ba}

Previously, we assumed that the adversary knows the exact input at the transmitter, $\boldsymbol{r}_{bt}$. However, it is impractical to assume that the adversary knows $\boldsymbol{r}_{bt}$. Thus, the adversary uses the received signal at the adversary, $\boldsymbol{r}_{ba}$, instead of $\boldsymbol{r}_{bt}$ to craft the adversarial perturbation. The same procedure used in Section 5.2.2 and Section 5.2.3 can be used to generate the adversarial perturbation without the knowledge of the

Algorithm 10: Adversarial perturbation using $\boldsymbol{r}_{bt}$

```
Inputs: $\boldsymbol{r}_{bt}$, desired accuracy $\varepsilon_{acc}$ and $P_{max}$
Initialize: $\varepsilon \leftarrow 0, \varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0, \boldsymbol{y}^{target} \leftarrow$ ‘noise’
$\boldsymbol{\delta}_{norm} = \frac{\nabla_{\boldsymbol{x}} \mathbf{H}^*_{at} L_a(\boldsymbol{\theta}_a, \boldsymbol{r}_{bt}, \boldsymbol{y}^{target})}{(||\nabla_{\boldsymbol{x}} \mathbf{H}^*_{at} L_a(\boldsymbol{\theta}_a, \boldsymbol{r}_{bt}, \boldsymbol{y}^{target})||_2)}$
if $\hat{l}_a(\boldsymbol{r}_{ba}) ==$ ‘noise’ then
    while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
        $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
        $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{r}_{bt} - \varepsilon_{avg} \boldsymbol{\delta}_{norm}$
        if $\hat{l}_m(\boldsymbol{x}_{adv}) ==$ ‘noise’ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
        else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
    end
end
$\varepsilon = \varepsilon_{max}$, $\boldsymbol{\delta}^* = -\varepsilon \boldsymbol{\delta}_{norm}$
```

input at the transmitter by changing $\boldsymbol{r}_{bt}$ to $\boldsymbol{r}_{ba}$ in conjunction with using the surrogate model of the adversary.

5.3 Topologies to Characterize the Effects of Surrogate Models on Adversarial Attacks

We consider different topologies to investigate the effect of surrogate models on the performance of adversarial attacks generated at the adversary in different locations. We assume that the channel between node i and node j is Rayleigh fading with path-loss and shadowing, i.e., $h_{ij,k} = K(\frac{d_0}{d_{ij}})^{\gamma} \psi h^{ray}_{ij,k}$, where $K = 1, d_0 = 1, \gamma = 2.7, \psi \sim$ Lognormal$(0, 8)$, $h^{ray}_{ij,k} \sim$ Rayleigh$(0, 1)$ and d_{ij} is the distance between node i and node j. Thus, if the location of the adversary changes with respect to the background emitter, the classifier at the adversary is trained with a different input distribution that changes with respect to the distance from the background emitter. Also, if the location of the adversary changes with respect to the transmitter, the usage of

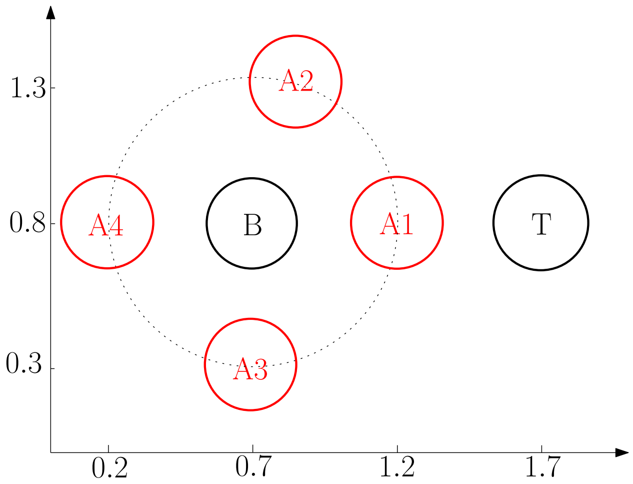


Figure 5.2: Fix the distance from the background emitter 'B' to the adversary (A1-A4 correspond to different locations of the adversary).

the power needed at the adversary changes, i.e., the adversary needs more power if the distance between the transmitter and adversary increases. To understand the performance of the classifiers at the adversary when the location changes, we consider the following adversary locations while we fix the distance between the background emitter and the transmitter as 1 in all topologies.

5.3.1 Fixed Distance Between the Background Emitter and the Adversary

We first consider topologies where the distance between the background emitter and the adversary is fixed for all cases and the only difference among adversary locations is the distance to the transmitter as in Fig. 5.2. In other words, the adversary uses the same surrogate model for these topologies. Specifically, the distance between the background emitter and the adversary, namely d_{ba} is 0.5 for adversary A1-A4 in Fig. 5.2 and the distance between the transmitter and the adversary, namely d_{ta}, is $0.5, 1, \sqrt{1.25}, 1.5$ for adversary A1-A4 in Fig. 5.2, respectively. Note that all the classifiers of the adversary at different locations are the same for these locations since the distributions of the received signals from the background emitter are the same. The only difference among the adversary locations is the distance to the transmitter.

5.3.2 Fixed Distance Between the Transmitter and the Adversary

Now, we consider the topology in Fig. 5.3 where the distance between the adversary and the transmitter is fixed for all cases and the difference among the adversary locations is their distance to the background emitter. In other words, the surrogate models of the adversary are different, whereas the channel effects on the adversarial perturbations have the same distributions. Specifically, the distance between the transmitter and the adversary, namely d_{ta} is 0.5 for adversary A1, A5, A6, and A7 in Fig. 5.3 and the distance between the transmitter and the adversary, namely d_{ba}, is $0.5, 1, \sqrt{1.25}, 1.5$ for adversary A1, A5, A6, and A7 in Fig. 5.3, respectively. Since the locations of the adversary are different with respect to the background emitter, the classifiers of the adversary at different locations are trained with different distribution of inputs. Note that the classifier at the transmitter is trained with the same distribution of inputs as adversary at position 5 (A5) and differently from other locations.

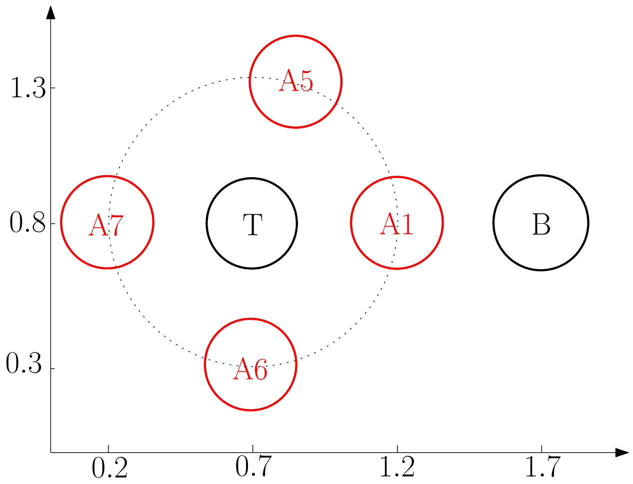


Figure 5.3: Fix the distance from the transmitter 'T' to the adversary (A1, A5-A7 correspond to different locations of the adversary).

5.4 Performance Evaluation

In this section, we investigate how different topologies described in Section 5.3 and the corresponding channels affect the performance of the adversarial perturbation built upon the adversary's surrogate model. We assume that there is QPSK signal transmission in the background and the classifiers at the transmitter and the adversary are a convolutional neural network (CNN) where the input to the CNN is of two dimensions (2,16) corresponding to 16 in-phase/quadrature (I/Q) data samples. We investigate the adversarial perturbation performance when the classifier at the adversary is the same as or different from the classifier at the transmitter. In all cases, the adversary trains a surrogate model that is different from the model of the transmitter's classifier.

The default CNN structure used in this chapter consists of a convolutional layer with kernel size $(1, 3)$, a hidden layer with dropout rate 0.1, ReLu activation

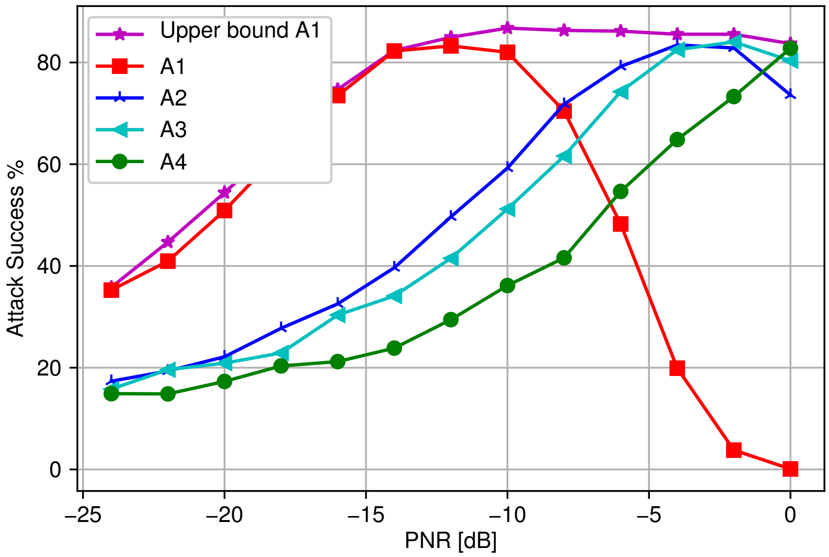


Figure 5.4: Attack performance of the adversary using maximum power when d_{ba} is fixed.

function at convolutional and hidden layers and softmax activation function at the output layer that provides the label 'signal' or 'noise'. We apply the backpropagation algorithm with Adam optimizer to train the CNN using cross-entropy as the loss function. The CNN is implemented in Keras with TensorFlow backend. Note that even though the architecture of the classifier at the transmitter and the adversary are the same, each classifier is trained with a different input data distribution due to the different channel between the background emitter and each node. We use the perturbation-to-noise ratio (PNR) metric that is also used in [51, 66] to represent the relative perturbation power with respect to the noise. Note that as the PNR increases, the probability of an attack to be detected also increases.

In Fig. 5.4, we first consider the topology depicted in Fig.5.2, where the distance between the background emitter and each adversary is the same. Note that we use the same classifier for the adversary at all locations in this topology

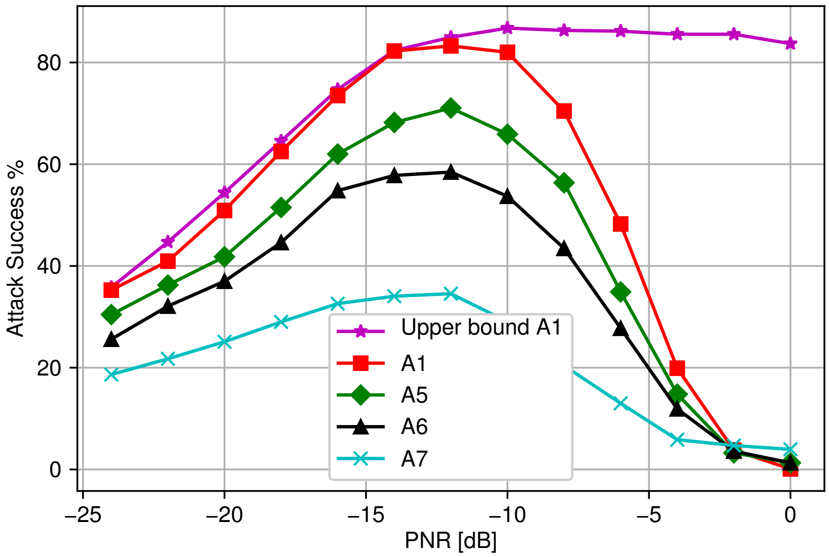


Figure 5.5: Attack performance of the adversary using maximum power when d_{ta} is fixed.

since the input distributions for the adversary at all these locations are the same and the only difference is its distance to the transmitter. Further, the case where the classifier of the adversary is the same as the transmitter's classifier is considered as an upper-bound where the power is determined using Algorithm 10. As the distance between the adversary and the transmitter increases, the peak of the curve shifts to the right meaning that it needs more power to fool the classifier at the transmitter. Moreover, it is observed that the attack success first increases as the PNR increases and then decreases after exceeding some PNR value. This is because the adversary transmits the adversarial perturbation with the maximum power that fails to fool the classifier at the transmitter into classifying the received signal as 'noise' when too much power is used at the adversary. Therefore, the adversary should select the right amount of power to transmit the adversarial perturbation. We observe that when the surrogate model at the adversary is the same as the transmitter's classifier, the attack success increases up to some PNR value and then saturates after exceeding it.

Now, we consider a different topology where the distance between the transmitter and each adversary is the same as shown in Fig. 5.3. Since the distance between the background emitter and the adversary at each location is different, the distributions of the received signal at different locations of the adversary are different. Thus, we train the classifiers for the adversary at different locations differently and use these classifiers in the simulations. In Fig. 5.5, we observe that as the distance between the background emitter and the adversary increases, the peak of the attack success decreases. In addition, even though the adversary at location 5 is trained with the same distribution as the transmitter's classifier, the classifier of the adversary at location 1 that is closer to the background emitter performs better. This observation suggests that the adversary should be located closer to the background emitter when the distance to the transmitter is fixed to increase the attack performance.

The performance of the adversarial attack using maximum power, using determined power, and using $\boldsymbol{r}_{ba}$ is compared in Fig. 5.6. It is seen that even though the adversary precisely decides the transmit power based on its own classifier, it is performing poorly compared to the case when the adversary uses maximum power. Further, when we relax the assumption of knowing $\boldsymbol{r}_{bt}$ and instead use $\boldsymbol{r}_{ba}$ to generate the adversarial attack at the adversary, the overall performance is worse compared to other methods and decreases when PNR increases.

Finally, we apply DNN architecture for the surrogate model at the adversary

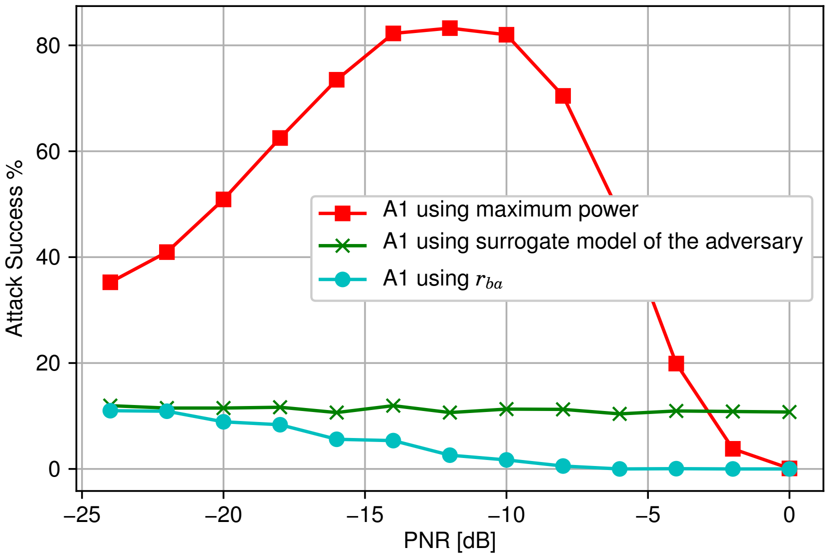


Figure 5.6: Attack performance of adversary using maximum power, surrogate model of the adversary and $\boldsymbol{r}_{ba}$.

that is different from the transmitter's classifier. The results are shown in Fig. 5.7. We observe that although the number of hidden layers increases compared to the CNN with one hidden layer, the impact on the attack success is negligible. The transferability [64] property holds since the performance has not changed although the architecture changes at the adversary. However, the surrogate models may be significantly different from the target model as the channels experienced by the transmitter and the adversary may differ. There may be other channel differences, such as multipath, intersymbol interference, and mobility (Doppler), present in practice. These differences will increase the difference between the target model and the surrogate model at the adversary, and reduce the attack success. Therefore, transferability argument is not readily applicable in practical wireless applications of over-the-air adversarial attacks.

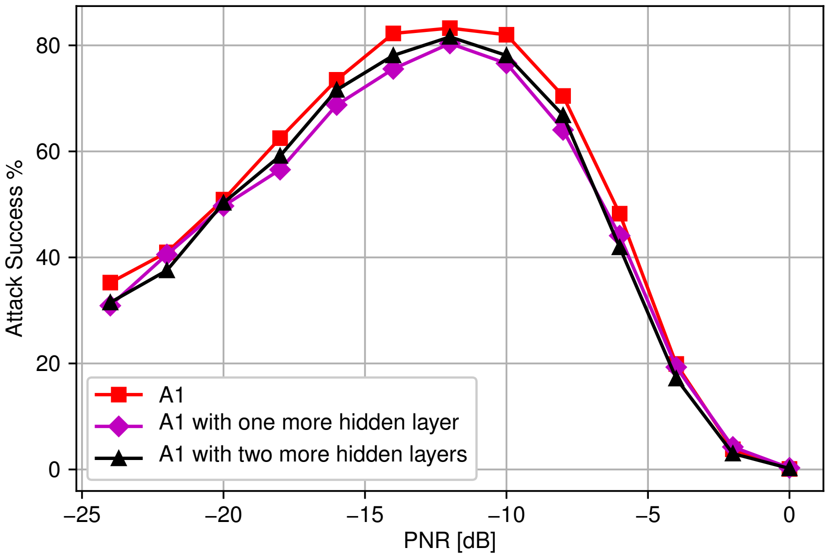


Figure 5.7: Attack performance of adversary with different classifier architecture.

5.5 Conclusion

We considered a wireless communication system where an adversary transmits a perturbation signal to fool the DNN classifier at a transmitter into classifying the ongoing background transmission as noise. The adversary trains its surrogate model by observing the spectrum and uses this model to design the adversarial attack. Through different topologies, we showed how the adversary's location significantly affects the attack performance as the surrogate model may differ from the target model due to channel discrepancies. In particular, the attack success against the transmitter drops when the adversary moves away from the background emitter since the surrogate model becomes less reliable although the adversary does not necessarily move closer to the transmitter.

CHAPTER 6

Adversarial Attacks on Deep Learning Based mmWave Beam Prediction in 5G and Beyond

6.1 Introduction

One particular application of DL is in the domain of initial access (IA) [83–86], where user equipments (UEs) need to establish their initial connection to an access point or base station when they attempt to join the communication network for the first time [87]. As 5G and beyond communication systems rely on mmWave and higher frequency bands to sustain high data rates over large available bandwidths, transmissions become more directional using narrow beams. This paradigm makes the beam alignment in the IA process more difficult as many narrow beams need to be swept to find the most suitable beam for each UE [87].

In the IA, the transmitter such as the 5G base station (gNodeB) sequentially transmits pilot signals over different narrow beams. The UE calculates the received signal strength (RSS) for each beam, determines the beam that provides the highest RSS, and informs the gNodeB of this beam selection. Since the time for the IA is limited, it is essential to reduce the number of beams swept as checking each beam consumes time and delays the UE to gain access to time-sensitive services such as ultra-reliable low-latency communications (URLLC) in 5G. To overcome this issue, a DL-based approach has been proposed in [88, 89] to reduce the number of beams that need to be swept before making the IA decision, compared to the conventional beam sweeping approach that needs to sweep all possible beams. In this DL-based approach, the UE predicts the best beam from a large set of narrow beams by using only the RSSs for a subset of these possible beams.

In this chapter, motivated by the need for DL-based IA, we investigate the vulnerability of a DNN that is used for mmWave beam prediction as part of the IA process in 5G and beyond communications. We first generate a non-targeted attack using the fast gradient method (FGM) [58] to cause any misclassification at the DNN classifier at the receiver (independent of the wrong labels for beams). Then, we introduce the k-worst beam attack that not only causes a misclassification but also enforces the DNN classifier to select one of the k worst beams to further reduce the IA performance. We compare the non-targeted FGM attack and k-worst beam attack with benchmark attacks with Gaussian and uniform noise added across RSSs from input beams. We show that the beam prediction of the IA process is highly vulnerable to adversarial attacks that can significantly reduce the beam prediction accuracy. The effect of this attack translates to notable reduction in communication performance in terms of data rate since the UEs connect to the gNodeB using beams that are not well aligned and the corresponding RSSs drop significantly.

6.2 System Model

We consider a mmWave network that consists of a directional transmitter (e.g., the gNodeB in 5G), an omnidirectional receiver (e.g., a UE) during the IA, and an adversary. A pre-trained DL-based classifier is applied to the RSS values measured at the receiver to select the best beam without a need to sweep all beams, as proposed in [88, 89]. Concurrently, the adversary attempts to cause misclassification at the receiver by launching an over-the-air adversarial attack.

In the conventional IA process with exhaustive beam sweeping, the transmitter transmits pilot signals over all narrow beams and the receiver uses the RSS values from all $\mathcal{N}$ possible beams where $N = |\mathcal{N}|$ to select the best beam. The transmitter transmits the pilot signal over each beam at a separate time slot. DL reduces the number of beams swept such that the DNN classifier at the receiver only uses the RSS values from $\mathcal{M}$ subset of beams, where $\mathcal{M} \subseteq \mathcal{N}$ and $M = |\mathcal{M}|$, to select the best beam. For that purpose, the transmitter transmits the pilot signals over a smaller set of beams, each at a separate time slot. Overall, the total time needed for the IA is reduced. In the DL-based approach, the DNN classifier at the receiver is denoted by $f(.;\boldsymbol{\theta}) : \mathcal{X}_M \rightarrow \mathbb{R}^N$, where $\mathcal{X}_M \subset \mathbb{R}^M$ is the input to the DNN which corresponds to the RSS values of M beams and $\boldsymbol{\theta}$ is the set of the receiver's DNN parameters. The classifier f assigns the best beam $\hat{l}(\boldsymbol{x}_M, \boldsymbol{\theta}) = \arg\max_j f_j(\boldsymbol{x}_M, \boldsymbol{\theta})$ to every input $\boldsymbol{x}_M \in \mathcal{X}_M$. In this formulation, $f_j(\boldsymbol{x}_M, \boldsymbol{\theta})$ is the output of classifier f corresponding to the jth beam.

As shown in Fig. 6.1, the adversary generates an adversarial attack by jam-

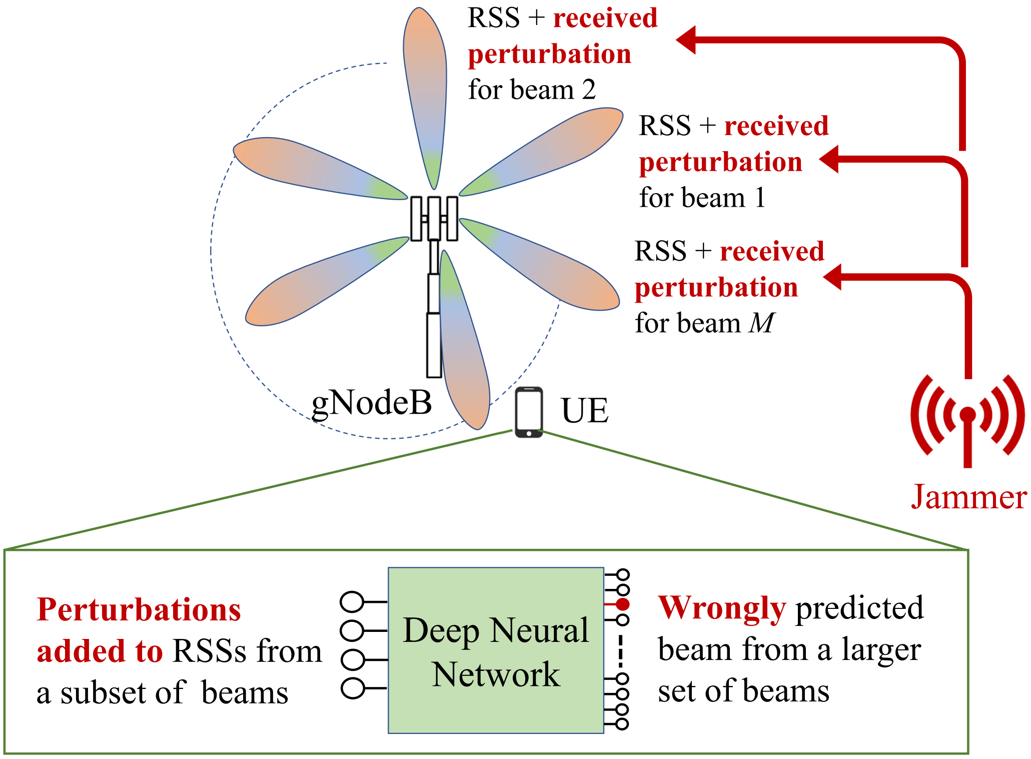

RSS + received perturbation for beam 2
RSS + received perturbation for beam 1
RSS + received perturbation for beam M
gNodeB
UE
Jammer
Perturbations added to RSSs from a subset of beams
Deep Neural Network
Wrongly predicted beam from a larger set of beams

Figure 6.1: Adversarial attack on DL-based beam prediction.

ming the spectrum with adversarial perturbation, $\boldsymbol{\delta} = [\delta_1, \delta_2, \cdots, \delta_M]$, under some suitable power constraint P_{max} with respect to the original RSS values, $\boldsymbol{x} \in \mathbb{R}^N$. The adversarial perturbation $\boldsymbol{\delta}$ is determined by solving the following optimization problem:

$$\begin{aligned} \arg\min_{\boldsymbol{\delta}} \quad & \sum_{i=1}^{M} \delta_i \\ s.t. \quad & \hat{l}(\boldsymbol{x}_M, \boldsymbol{\theta}) \neq \hat{l}(\boldsymbol{x}_M + \boldsymbol{\delta}, \boldsymbol{\theta}), \ \sum_{i=1}^{M} \delta_i \leq P_{max}. \end{aligned} \tag{6.1}$$

Note that $\boldsymbol{x}_M \in \mathbb{R}^M$ is the M elements from $\boldsymbol{x}$ which correspond to the RSS values at the receiver for all beams.

Solving (6.1) is hard due to the inherent structure of the DNN. Thus, different methods have been proposed to approximate the adversarial attack such as FGM. FGM is a computationally efficient way of creating adversarial attacks by linearizing the loss function of the DNN. We denote the loss function of the DNN classifier by $L(\boldsymbol{\delta}, \boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{y} \in \{0,1\}^N$ is the one-hot encoded class vector. Then, FGM linearizes this loss function in a neighborhood of $\boldsymbol{x}$ and uses this linearized function for optimization.

We assume that the adversary knows the architecture ($\boldsymbol{\theta}$ and $L(\cdot)$) of the DNN classifier at the receiver, as well as the exact RSS values for all N beams that are received at the receiver. Later in the chapterr, we relax the assumption about knowing all N RSS values to knowing only M RSS values, and compare the results under these two assumptions. In the next section, we describe the two adversarial attacks considered in this chapter, namely, the non-targeted attack and the k-worst attack.

6.3 Adversarial Attacks on Beam Prediction

6.3.1 Non-Targeted FGM Attack

First, we consider a non-targeted FGM attack where the adversary searches for a perturbation that causes any misclassification at the receiver's DNN classifier. The adversary designs a perturbation that maximizes the loss function $L(\boldsymbol{\delta}, \boldsymbol{x}_M, \boldsymbol{y}^{true})$, where $\boldsymbol{y}^{true}$ is the true label of $\boldsymbol{x}_M$. FGM is used to linearize the loss function as $L(\boldsymbol{\theta}, \boldsymbol{x}_M + \boldsymbol{\delta}, \boldsymbol{y}^{true}) \approx L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{true}) + \boldsymbol{\delta}^T \nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{true})$ that is maximized

Algorithm 11: Non-targeted FGM attack

```
Inputs: RSS values $\boldsymbol{x}_M$, true label $\boldsymbol{y}^{true}$, desired accuracy $\varepsilon_{acc}$, power
  constraint $P_{max}$, and model of the classifier
Initialize: $\varepsilon \leftarrow 0, \varepsilon_{max} \leftarrow P_{max}, \varepsilon_{min} \leftarrow 0$
$\boldsymbol{\delta}_{norm} = \frac{\nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta},\boldsymbol{x}_M,\boldsymbol{y}^{true})}{(||\nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta},\boldsymbol{x}_M,\boldsymbol{y}^{true})||_1)}$
if $\hat{l}(\boldsymbol{x}_M, \boldsymbol{\theta}) == \boldsymbol{y}^{true}$ then
    while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
        $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
        $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x}_M + \varepsilon_{avg}\boldsymbol{\delta}_{norm}$
        if $\hat{l}_m(\boldsymbol{x}_{adv}) == \boldsymbol{y}^{true}$ then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
        else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
    end
end
$\varepsilon = \varepsilon_{max}$, $\boldsymbol{\delta}^* = \varepsilon\boldsymbol{\delta}_{norm}$
```

by setting $\boldsymbol{\delta} = \alpha\nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{true})$, where α is a scaling factor to constrain the adversarial perturbation power to P_{max}. The details of selecting α and generating the non-targeted FGM attack are presented in Algorithm 11.

6.3.2 k-worst Beam Attack

Next, we present an adversarial attack where the adversary not only causes a misclassification at the receiver's DNN classifier but also tries to change the beam to one of the worst k beams. Unlike attacks on some signal classification tasks (such as in the computer vision or wireless domains), where changing the label from 'signal 1' to 'signal 2' and from 'signal 1' to 'signal 3' may have the same effect on the classifier meaning that both correspond to an error in classification, changing the label from the best beam to second worst beam and from the best beam to the worst beam have totally different effects on the communication performance (as the signal quality on a beam strongly affects the achieved rate following the IA). Thus, the adversary first tries to change the label to the worst beam and if the adversary is not able to make this change, the adversary tries to change the label to the second worst beam. The adversary continues to do so until it tries for the kth worst beam.

Here, we consider the targeted FGM attack where the adversary aims to fool the DNN to a target label and tries to minimize the loss function $L(\boldsymbol{\delta}, \boldsymbol{x}_M, \boldsymbol{y}^{target})$, where $\boldsymbol{y}^{target}$ is one of the k worst beams. FGM is used to linearize the loss function as $L(\boldsymbol{\theta}, \boldsymbol{x}_M + \boldsymbol{\delta}, \boldsymbol{y}^{target}) \approx L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{target}) + \boldsymbol{\delta}^T \nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{target})$ that is minimized by setting $\boldsymbol{\delta} = -\alpha \nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{target})$, where α is a scaling factor to constrain the adversarial perturbation power to P_{max}. First, we assume that the adversary knows the order of beam indices based on the real RSS values. Then, we relax this assumption by getting the order from the DNN's output. Since the DNN is only trained to find the best beam, there exists a discrepancy between the real order of beams and the order of beams obtained from the DNN output. The details of the algorithm are presented in Algorithm 12.

6.4 Performance Evaluation

This section describes the DNN classifier used at the receiver and the dataset that is used to train it, and then compares the performances of the two attacks that are introduced in this chapter with two benchmark jamming attacks that inject Gaussian or uniform noise on different beams.

```
Algorithm 12: $k$-worst beam attack
  Inputs: RSS values $\boldsymbol{x}_M$, true label $\boldsymbol{y}^{true}$, $k$ worst beam indices, desired
    accuracy $\varepsilon_{acc}$, power constraint $P_{max}$, and model of the classifier
  if $\hat{l}(\boldsymbol{x}_M, \boldsymbol{\theta}) == \boldsymbol{y}^{true}$ then
      for $i$ in $k$ worst beam indices do
          Initialize: $\varepsilon \leftarrow 0, \varepsilon_{max} \leftarrow P_{max}, \varepsilon_{min} \leftarrow 0, \boldsymbol{y}^{target} \leftarrow i$
          $\boldsymbol{\delta}_{norm} = \frac{\nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{target})}{(||\nabla_{\boldsymbol{x}_M} L(\boldsymbol{\theta}, \boldsymbol{x}_M, \boldsymbol{y}^{target})||_1)}$
          while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
              $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
              $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x}_M - \varepsilon_{avg}\boldsymbol{\delta}_{norm}$
              if $\hat{l}_m(\boldsymbol{x}_{adv}) == \boldsymbol{y}^{true}$ then  $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
              else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
          end
          $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{x}_M - \varepsilon_{avg}\boldsymbol{\delta}_{norm}$
          if $\hat{l}_m(\boldsymbol{x}_{adv}) == i$ then break
      end
  end
  $\boldsymbol{\delta}^* = -\varepsilon_{avg}\boldsymbol{\delta}_{norm}$
```

6.4.1 Deep Learning Framework for IA

We use the DNN structure of seven layers including the input and the output layers where five hidden layers have 32, 64, 126, 64, 32 neurons respectively. The input layer has $M = 12$ neurons and the output layer is a Softmax layer that has $N = 24$ neurons where each neuron represents a likelihood score for each beam. Each dense layer uses RELU as an activation function and the output is batch normalized before sending it to the next layer. Cross-entropy is used as the loss function. Adam optimizer is used with initial and final learning rates 10^{-3} and 0.1, respectively. We follow the simulation setup of [88] and apply the attacks in a two-dimensional mmWave network where line-of-sight (LoS) mmWave channels with pathloss and shadowing effects are considered. A 10×10 antenna array is used at the transmitter to generate a beam width of about 15° using the standard planar array formulation. The location of the transmitter is fixed at $(0, 0)$ and the receiver's x and y positions are each uniformly randomly distributed between -25 m and 25 m. This bounds the simulation cell to an area of 50×50 m. The transmit power is set to 20 dBm. A total of 10^6 receiver positions are generated, which act as data samples. The adversary is also uniformly distributed in this area and its perturbations are reflected on the data samples.

6.4.2 Attack Performance Results

We compare the two attack schemes that we have described earlier with the Gaussian attack and the uniform attack that generate perturbations with Gaussian and equal power distribution for M beams, respectively. In the simulations, we use the perturbation-to-signal ratio (PSR) metric that shows the relative perturbation power with respect to the received signal power (namely, the RSS at the receiver). As the PSR increases, the attack becomes more likely to be detected. Fig. 6.2 presents the accuracy of the classifier at the receiver under the non-targeted FGM attack and compares it with the Gaussian attack and the uniform attack. The non-targeted FGM attack significantly impacts the accuracy of the classifier for beam selection even for signal strength fluctuations that cannot be resolved with typical hardware. The Gaussian attack and the uniform attack do not perform well compared to the non-targeted attack in [-40dB,-20dB] region, while they are comparable in the high PSR region.

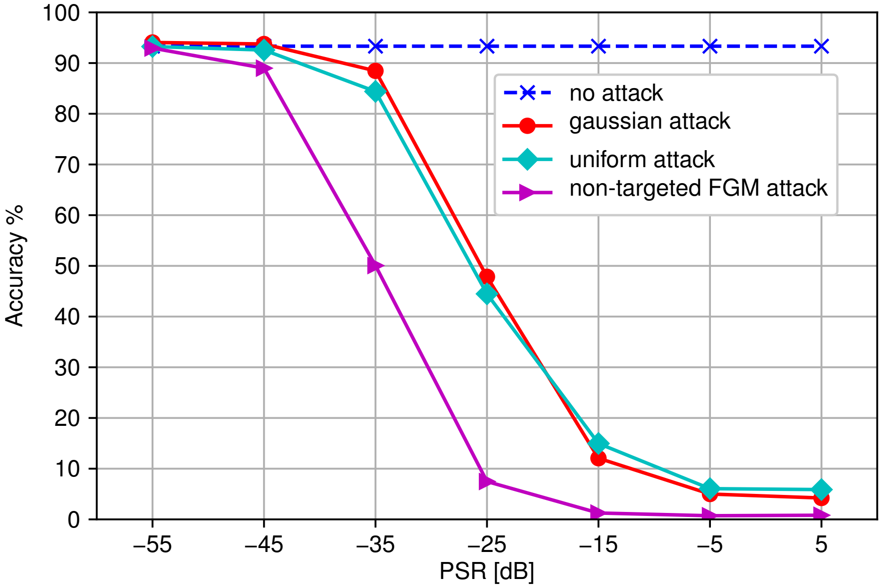


Figure 6.2: Classifier accuracy under the non-targeted attack.

In Fig. 6.3, we investigate the performance of the k-worst beam attack, where k is set as 4, 8 and 12, and compare it with the Gaussian attack and the uniform attack. Note that the accuracy definition in Fig. 6.3 is different from Fig. 6.2, where the accuracy is defined as the percentage that the label obtained from the DNN classifier is in the $N - k$ best beams since the attack is successful only if the k-worst beam attack fools the DNN classifier into choosing one of the worst k beams. As k increases for the k-worst beam attack, the DNN classifier accuracy decreases meaning that it is harder to enforce the beam selection to the worst beams. Also, the k-worst beam attack using the real order of RSSs outperforms the k-worst beam attack using the DNN order as there is a discrepancy between the real order and the DNN-predicted order of RSSs. Furthermore, the Gaussian attack and the uniform attack both saturate around 50% meaning that under both attacks the beams are misclassified to the best group or the worst group 50% of the time.

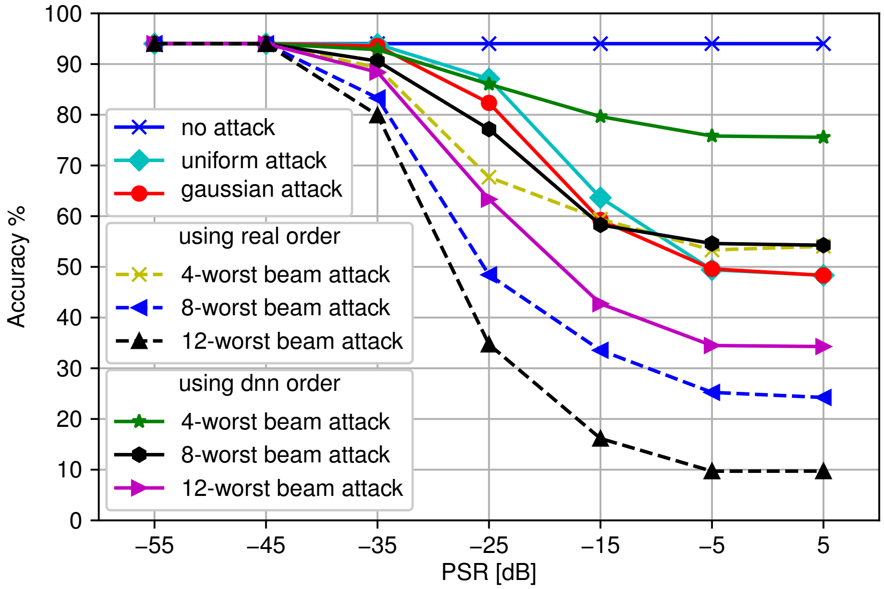


Figure 6.3: Classifier accuracy under the k-worst beam attack.

6.5 Conclusion

We presented an adversarial attack to fool the beam selection process of the IA based on a DNN classifier that uses only a subset of beams to predict the beam that is best oriented to the receiver. We investigated two different attacks, namely, the non-targeted FGM attack that only aims to fool the DNN classifier with misclassification to any other beam label, and the k-worst beam attack that not only fools the DNN classifier but also enforces the label that is chosen at the DNN to be in one of the k-worst beams. We showed that the adversarial attack can significantly decrease the accuracy of the DNN and fool the DNN into selecting the worst beam. We conclude that as DL finds applications for beam prediction in mmWave communication for 5G and beyond, the IA becomes vulnerable to adversarial attacks that can significantly reduce the beam selection performance.

CHAPTER 7

Adversarial Attacks against Deep Learning Based Power Control in Wireless Communications

7.1 Introduction

In this chapter, we use a regression-based DNN, unlike previous chapters where we use a DNN classifier, at the BS to allocate the power to orthogonal subcarriers and serve multiple users. Power allocation problem using DL have been studied in [90–92] and adversarial attacks on the MIMO power control have been considered in [93] with the goal of preventing the underlying DNN (that is trained to maximize the product of signal-to-noise-ratios (SNRs) by taking the UE positions as the input) from finding a feasible solution. In this chapter, we formulate the power allocation under the attack to rely on a robust and practical DNN solution that always finds a feasible solution for any set of channel estimates given as the input. To launch an attack on this DNN, we consider an adversary that manipulates the input (channel gains) to the DNN in test time to minimize the minimum rate among all UEs. The adversary can be modeled in two ways: (i) the adversary is an external transmitter that aims to manipulate the inputs to the DNN over-the-air by interfering with the pilot signals that are transmitted to estimate the channel gain, or (ii) the adversary is a rogue UE that transmits fabricated channel estimates back to the BS.

We design an adversarial attack to change the DNN's input to manipulate the minimum rate over all UEs subject to the condition that the perturbations to the inputs of the DNN are bounded. In particular, the adversary generates the adversarial attacks to manipulate the minimum rate by crafting the perturbations to the DNN input based on the gradient of the minimum rate. For that purpose, we consider two approaches to compute the gradient of the rate with respect to the inputs to the DNN when crafting the perturbation: (i) the adversary obtains the DNN power allocation outputs from its surrogate model, computes the minimum rate based on these outputs using analytical means, and then computes the gradient of the rate with respect to the changes to the DNN input, and (ii) the adversary aims to attack the DNN by calculating the gradient of the DNN's loss function using the fast gradient method (FGM), where we define the DNN's loss function as the error with respect to the minimum rate.

We consider the attacks targeted on a single UE or all UEs, i.e., the adversary aims to manipulate the channel gain estimates of a single UE or all UEs, respectively. We compare these attacks with a benchmark attack, namely, a scaling down attack, where the adversary scales down the input to the DNN. Our results show that the adversarial attacks are much more effective than the benchmark attack in terms of reducing the rate of communications even if a small perturbation is used. We also show that the adversarial attacks can be effectively launched even under two types of uncertainty at the adversary, (i) the knowledge of channel gains at the adversary is

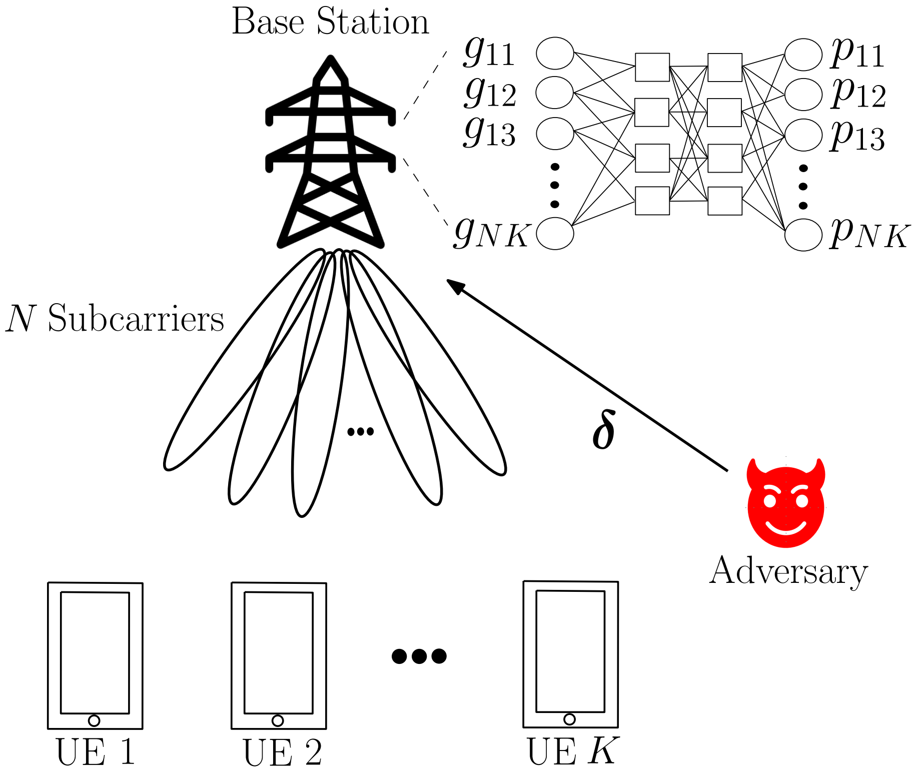


Figure 7.1: Wireless communication system with a base station, K UEs, and an adversary.

erroneous, and (ii) the adversary cannot generate the exact planned perturbation in the channel gain estimate of the UE. Overall, these results show that the adversarial attacks pose a serious threat to power allocation solutions that rely on DL.

7.2 Victim Model: DL for Power Allocation

We consider the power allocation problem for downlink communications from the BS using N different orthogonal subcarriers to communicate with K UEs, where the downlink signal transmitted by the ith subcarrier of the BS to the jth UE is x_{ij} and the corresponding power is $|x_{ij}|^2 = p_{ij}$. The channel between the ith subcarrier of the BS to the jth UE is h_{ij} and the corresponding channel gain is $g_{ij} = |h_{ij}|^2$. For channel estimation, the BS transmits pilot signals from each of its subcarriers one by one, the UE estimates the channel gains, and reports them back to the BS. Based on these channel estimates, the BS allocates power to its subcarriers to serve each of the UEs. In particular, power p_{ij} for UE j's data at subcarrier i is an optimization variable to be determined by the BS, where $\sum_i \sum_j p_{ij} \leq p$. Then the received signal at the jth UE for subcarrier i is

$$s_{j,i} = h_{ij}x_{ij} + \sum_{k \neq j} h_{ij}x_{ik} + n_i, \tag{7.1}$$

where n_i is the noise with power σ_i^2. The rate of UE j is given as in [94] by

$$r_j(\boldsymbol{p}) = \sum_{i=1}^{N} \log_2 \left(1 + \frac{g_{ij}p_{ij}}{\sigma_i^2 + \sum_{k \neq j} g_{ij}p_{ik}} \right), \tag{7.2}$$

where $\boldsymbol{p} = [p_{11}, p_{12}, \cdots, p_{NK}]$.

The achievable rates for the UEs are considered by the BS to allocate the transmit power where the objective of the BS can be maximizing the minimum rate of all UEs, namely maximizing $r_{\min}$, where $r_{\min} \leq r_j(\boldsymbol{p})$ for all j, by allocating the transmit power to subcarriers for UEs. Therefore, we have the following optimization problem:

$$\begin{aligned} \max_{\boldsymbol{p}} \quad & r_{\min} \\ \text{s.t.} \quad & r_{\min} \leq r_j(\boldsymbol{p}), \quad 1 \leq j \leq K \\ & \sum_{i=1}^{N} \sum_{j=1}^{K} p_{ij} \leq p \end{aligned}$$

$$p_{ij} \geq 0, \quad 1 \leq i \leq N, 1 \leq j \leq K. \tag{7.3}$$

This is a nonlinear optimization problem. Although some methods such as interior point and trust region can be applied to solve such nonlinear optimization problems, the complexity could be high for online power allocation under dynamic channel gains. Thus, the BS can build a DL algorithm, namely train a DNN, to solve (7.3). The input for this multi-output regression problem is the set of channel gains g_{ij} (note that σ_i^2 and p are constants) and the output is the power allocation p_{ij}, where the training data (input and output) samples are obtained by solving (7.3) offline. Specifically, we define the dataset, $\{\boldsymbol{x}(n), \boldsymbol{y}(n)\}_{n=1}^{N_t}$, where the input $\boldsymbol{x}(n)$ is the channel gain, the output $\boldsymbol{y}(n)$ is the power allocation, and N_t is the size of the training dataset. We denote the regression-based DNN at the BS as $f(\boldsymbol{x}(n); \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the set of DNN parameters, the loss function of the DNN at the BS is $L_g(\boldsymbol{\theta}, \boldsymbol{x}(n), \boldsymbol{y}(n))$, and the predicted power allocation $f(\boldsymbol{x}(n), \boldsymbol{\theta})$ is $\hat{\boldsymbol{p}}$.

While the BS determines the power allocation variable $\hat{\boldsymbol{p}}$ from the estimated channel gains using the pre-trained regression-based DNN, there exists an adversary that aims to decrease the minimum rate among UEs by manipulating the channel information at the BS so that the BS makes wrong decisions. While doing so, the adversary may attack one UE (and change its channel gains) or attack all UEs. We assume that there is a budget on these changes, which is measured as the percentage of the total original channel gains. For example, suppose the budget is 1% (or, 0.01). Then, the total change can be no more than $0.01 \sum_i g_{ij}$, if the adversary attacks

UE j, or $0.01 \sum_i \sum_j g_{ij}$, if the adversary attacks all UEs. The adversary can train its own DNN and use this DNN as a surrogate model for the DNN of the BS. We introduce two approaches to generate an adversarial attack at the adversary, based on this surrogate model.

7.3 Adversarial Attacks on Power Control

7.3.1 Simplified Analytical Gradient-Based Attack

To maximize the impact of changes at the BS, the adversary needs to carefully spend the budget of changes on channel gains. The first approach that we consider for this purpose is based on the analysis of each channel gain's gradient using (7.2) and the adversary DNN's power outputs. Denote the gradient for the ith subcarrier and UE j as η_{ij}, which can be determined by channel gains and power values as

$$\eta_{ij} = \frac{p_{ij}\sigma_i^2}{(\sigma_i^2 + g_{ij}\sum_{k\neq j} p_{ik})(\sigma_i^2 + g_{ij}\sum_{k=1}^{N} p_{ik})\log_e 2}, \tag{7.4}$$

where we simplify the problem by ignoring the dependency between p_{ij} and g_{ij}. In Section 7.3.2, we will use the gradient through the loss function defined for the DNN to consider this dependency. Since $\eta_{ij} > 0$, the channel gain from the ith subcarrier to UE j should be decreased so that the rate can be smaller. Moreover, it is more effective if the adversary changes a channel gain with a larger value η_{ij}. Thus, the adversary first selects the channel gain g_{ij} with the largest η_{ij} and tries to decrease this channel gain. The details to attack a specified UE j are presented

Algorithm 13: Simplified analytical gradient-based attack algorithm for changing channel gains.

```
Input: the target UE $j$, channel gain $g_{ik}$ for each subcarrier $i$ and UE $k$,
  budget for total change $B_g$, a small threshold $\varepsilon$ for minimum channel gain
Calculate: gradient $\eta_{ij}$ for each subcarrier $i$ by (7.4)
Sort: $\eta_{ij}$ in a list $A$ based on the non-increasing order
for $i \in A$ do
    if $g_{ij} \geq B_g + \varepsilon$ then $\delta_{ij} = -B_g$ break
    else $B_g = B_g - g_{ij} + \varepsilon$ and $\delta_{ij} = \varepsilon - g_{ij}$
end
Output: $\delta_{ij}$ for $i = 1, \cdots, N$
```

in Algorithm 13.

If the adversary can attack all UEs, it first selects the UE j with the largest $\sum_i \eta_{ij}$. If the budget for change permits, the adversary decreases $g_{ij}, i = 1, \cdots, N$, to 0 and then selects the next UE to attack. Otherwise, the adversary applies Algorithm 13 to attack this UE.

Once the channel gains are changed by the adversary, the BS makes its decision on power allocation based on the incorrect channel gains and determines the transmitted data rate for each UE based on allocated powers and incorrect channel gains. On the other hand, the maximum link rate for each UE is determined by the allocated powers and real channel gains. If the transmitted rate is no more than the maximum link rate, the achieved rate is the transmitted rate, otherwise the achieved rate is zero since the transmitted data cannot be decoded by an UE.

7.3.2 DNN Gradient-Based Attack

The second approach for the adversary to craft the adversarial perturbation $\boldsymbol{\delta}$ solves the following optimization problem:

$$\begin{aligned}
\min_{\boldsymbol{\delta}} \max_{\boldsymbol{p}} \quad & r_{\min} \\
\text{s.t.} \quad & r_{\min} \le r'_j(\boldsymbol{\delta}), \quad 1 \le j \le K \\
& \sum_{i=1}^{N} \sum_{j=1}^{K} |\delta_{ij}| \le B_g \\
& \sum_{i=1}^{N} \sum_{j=1}^{K} p_{ij} \le p \\
& p_{ij} \ge 0, \quad 1 \le i \le N, 1 \le j \le K,
\end{aligned} \tag{7.5}$$

where

$$r'_j(\boldsymbol{\delta}) = \sum_{i=1}^{N} \log_2 \left(1 + \frac{(g_{ij} + \delta_{ij}) p_{ij}}{\sigma_i^2 + ((g_{ij} + \delta_{ij}) \sum_{k \neq j} p_{ik})} \right) \tag{7.6}$$

and B_g is the budget for total change at the adversary. However, solving (7.5) is hard due to nonlinearity. Thus, we use the FGM [58] to linearize the loss function $L_g(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$ of the adversary's DNN in a neighborhood of $\boldsymbol{x}$ and use this linearized function to generate an adversarial attack. Since the goal of the adversary is to minimize $r_{\min}$, the adversary defines a loss function $L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$ that calculates $r_{\min}$. Note that the loss function $L_g(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$ is used for training and the loss function

Algorithm 14: DNN gradient-based attack algorithm for changing channel gains.

```
Input: channel gain $\boldsymbol{x}$, budget for total change $B_g$, and architecture of the DNN
Loss function: use $L_a$ to generate perturbation that minimizes the rate among all UEs
Calculate: $\nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$
if $\min\{\nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})\} < 0$ then
    $\boldsymbol{\eta} = \nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y}) - \min\{\nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})\}\mathbf{1}$
end
else $\boldsymbol{\eta} = \nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$
Transmit: $\boldsymbol{\delta} = B_g \frac{\boldsymbol{\eta}}{||\boldsymbol{\eta}||_1}$
```

$L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})$ is used to create an attack. Therefore, the adversary uses

$$\boldsymbol{\delta} = B_g \frac{\nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})}{(||\nabla_{\boldsymbol{x}} L_a(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{y})||_1)} \tag{7.7}$$

to attack all UEs. Then, the BS receives $g_{ij} + \delta_{ij}$ for $i = 1, \cdots, N$ and $j = 1, \cdots, K$. Note that $g_{ij} + \delta_{ij}$ can be negative or greater than 1 depending on the δ_{ij}. For this case, the negative value is changed to zero and the value greater than 1 is changed to 1, since a channel gain is always in $[0, 1]$. Thus, the budget B_g is not fully used. We can fully utilize the budget B_g in (7.7) by shifting the perturbation so that $g_{ij} + \delta_{ij}$ stays in $[0, 1]$. The perturbation $\boldsymbol{\delta}$ to attack a single UE can be determined similarly by changing $\delta_{ik} = 0$ for $k \neq j$. The details are presented in Algorithm 14.

7.4 Performance Evaluation

We consider a feedforward neural network (FNN) trained as a multi-output regression model for power control. As N is the number of subcarriers and K is the number of UEs, both the input layer and the output layer have size $N \times K$, which

Table 7.1: The DNN architecture for the power allocation.

Layers	Number of neurons	Activation function
Input	$N \times K$	-
Dense 1	1024	ReLu
Dense 2	1024	ReLu
Dense 3	1024	ReLu
Dense 4	512	ReLu
Output	$N \times K$	Softmax

corresponds to both the total number of channels (the input of the FNN) and the total number of powers to be allocated (the output of the FNN). To ensure the output power constraint (i.e., the sum of power outputs is less than or equal to p), the activation function for the output layer is set as softmax (such that the DNN outputs are summed up to 1) and the output values are multiplied by p to get power values. To collect training data, we generate 50000 random instances of channel gains for $N = 4$, $N = 10$, and $N = 20$, where the number of UEs, K, is fixed as 3, and then solve (7.3) by the interior point method in MATLAB to obtain the corresponding power allocation and achieved objective value $r_{\min}$. We train three different DNNs for different number of subcarriers and use half of the generated dataset to train the DNN. The noise power σ_i^2 is $1/N$ and the total power is $p = 10$ during the simulations. The DNN structure for the power allocation is given in Table 7.1.

We use different loss functions for the DNN at the BS. The mean absolute error (MAE) loss function aims to minimize the average absolute error between powers p_{ij} of the training data and powers $\hat{p}_{ij}$ obtained by the DNN, i.e.,

$$l_{\mathrm{MAE}} = \frac{1}{NK} \sum_i \sum_j |p_{ij} - \hat{p}_{ij}|. \tag{7.8}$$

The mean absolute percentage error (MAPE) loss function aims to minimize

$$l_{\text{MAPE}} = \frac{1}{NK}\sum_{i}\sum_{j}\frac{|(p_{ij}+c)-(\hat{p}_{ij}+c)|}{p_{ij}+c}, \tag{7.9}$$

where a constant $c = 10$ is added to all powers to avoid the divided-by-zero issue. The mean squared logarithmic error (MSLE) loss function aims to minimize

$$l_{\text{MSLE}} = \frac{1}{NK}\sum_{i}\sum_{j}(\log(p_{ij}+1)-\log(\hat{p}_{ij}+1))^2. \tag{7.10}$$

We apply the ADAM optimizer and find that DL can always achieve small loss values for all loss functions. However, if we calculate the ratio between the achieved minimum rate using the DNN's output with the minimum rate achieved by training data, these loss functions translate to different performance. For $N = 4$ and $K = 3$, the average ratio between these two rates is 86.37% for MSLE, 85.86% for MAE, and 84.11% for MAPE. Since the aim of the BS is to maximize the minimum rate among all UEs, we define our custom loss function that aims to minimize

$$l_{\text{custom}} = (\min_{j} r_j(\boldsymbol{p}) - \min_{j} r_j(\hat{\boldsymbol{p}}))^2, \tag{7.11}$$

which achieves 94.45% as the ratio between the achieved minimum rate using the DNN's output and the minimum rate achieved by training data when $N = 4$ and $K = 3$. Thus, we adopt the custom loss function as our loss function during the simulations. Note that this is also the loss function that is used to create the adversarial perturbation at the adversary. Throughout the simulations, we use this

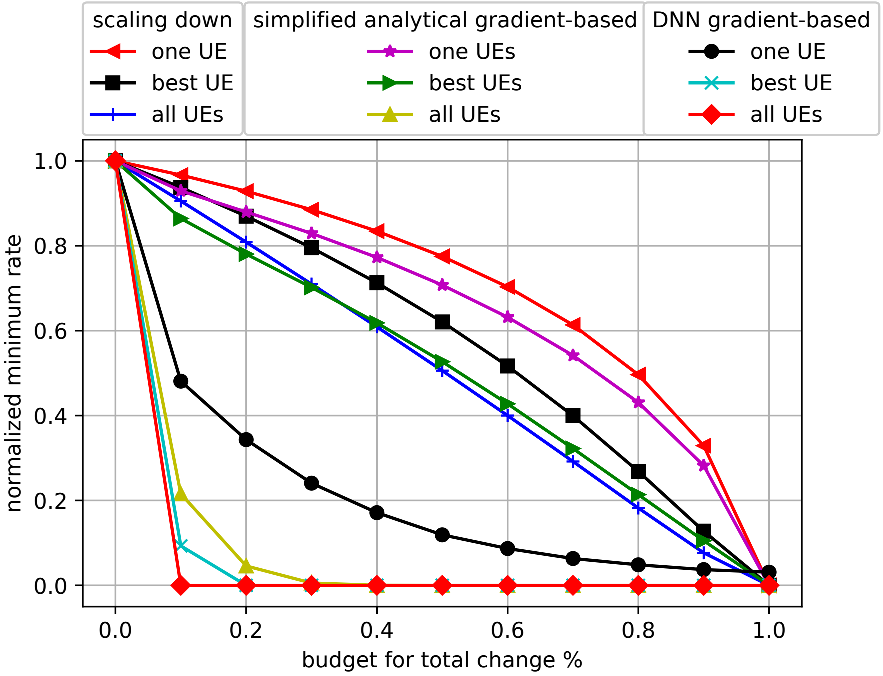


Figure 7.2: The normalized minimum rate when $N = 4$ and $K = 3$.

normalized rate ratio.

In Fig. 7.2, we compare the two gradient-based attacks when $N = 4$ and $K = 3$. The scaling down attack is also compared as a benchmark attack that enforces the input at the DNN to scale down by $1 - \rho$. For the one UE case, the adversary always attacks UE 1. We also consider a hypothetical scheme that the adversary can always find the best UE to attack. The scaling down attack has poor performance compared to the other attacks. Attacking all three UEs outperforms the cases where the adversary attacks one fixed UE or the best UE. The DNN gradient-based attack on all UEs outperforms other attack schemes, while the simplified analytical gradient-based attack on all UEs has comparable attack performance.

Next, we compare different attacks in Fig. 7.3 when $N = 10$ and $K = 3$. Without any attack, the regression-based DNN at the BS reaches 90.90% ratio

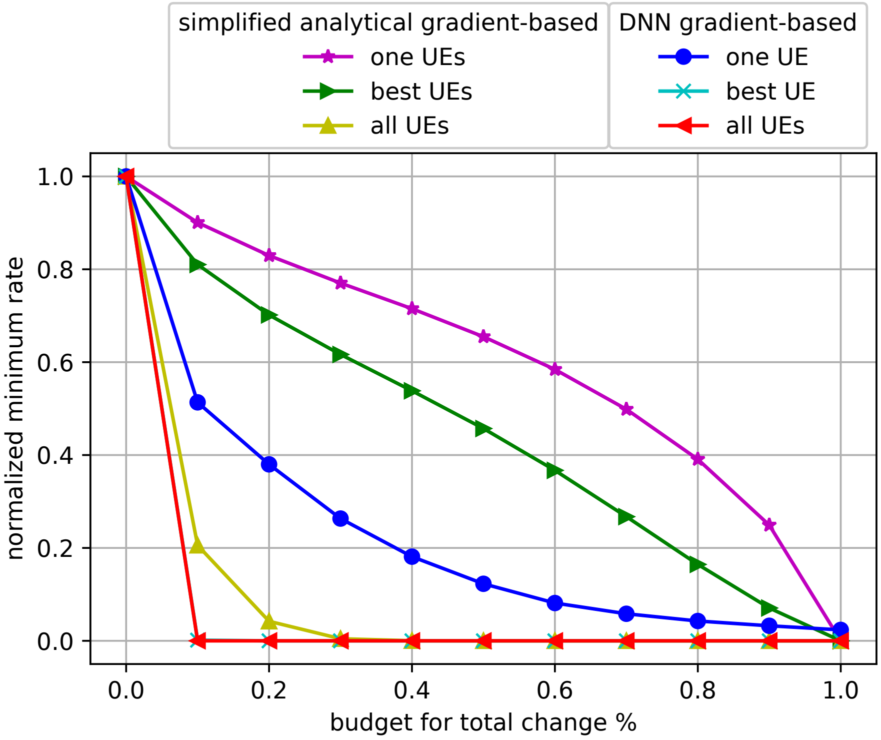


Figure 7.3: The normalized minimum rate when $N = 10$ and $K = 3$.

between the achieved minimum rate using the DNN's output and the minimum rate achieved by training data. Attacking all UEs simultaneously has more effect on the minimum rate among UEs compared to attacking only one UE for all attack schemes. Moreover, the DNN gradient-based attack outperforms the analytical gradient-based attack when all UEs are under attack. It is also observed that when the DNN gradient-based attack is used for attacking the best UE has the same effect as attacking all UEs simultaneously. Similar results are also obtained in Fig. 7.4 when $N = 20$ and $K = 3$. Without any attack, the DNN at the BS reaches 83.73% ratio between the achieved minimum rate using the DNN's output and the minimum rate achieved by training data.

We also consider the impact of uncertainty on adversarial attacks. The first

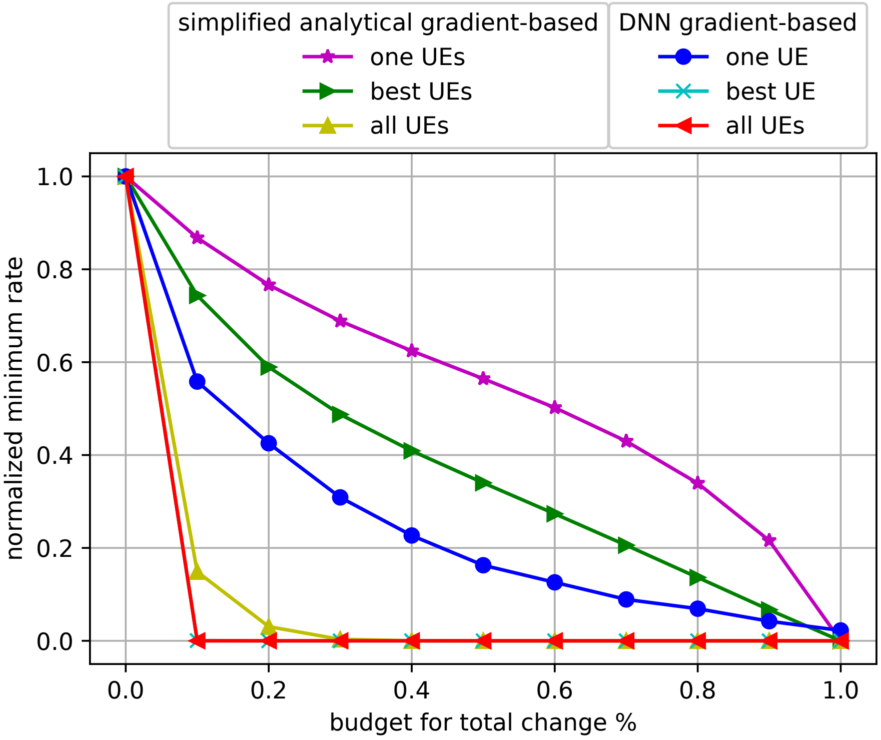


Figure 7.4: The normalized minimum rate when $N = 20$ and $K = 3$.

uncertainty is that the adversary may not have accurate knowledge on channel gains, i.e., if the real gain is g, the adversary may have an estimate within $[(1-e)g, (1+e)g]$, where e is the error ratio. The second uncertainty is that the adversary may not be able to launch the attack exactly as specified, i.e., if the adversary aims to change a channel gain by amount c, the actual change may take a value in $[(1-e)c, (1+e)c]$. This uncertainty may be due to the erroneous knowledge of the adversary about the channel gain from itself to the UE such that it exercises the wrong perturbation power. For the analytical gradient-based attack, Table 7.2 shows the normalized minimum rate under uncertainty when the budget to change UE 1's channel gains is up to 10% when 4 subcarriers are used. Note that when there is no error, the attack leads to 87.77% of the rate ratio. Results in Table 7.2 show that the errors

Table 7.2: The impact of uncertainty on attack performance.

Error ratio	5%	10%	15%	20%
Error on channel gain	87.77%	87.79%	87.81%	87.84%
Error on channel change	88.02%	88.25%	88.47%	88.67%

on channel gains and channel changes will reduce the effect of attack (the rate under attack is higher when there are such errors). However, such effect is small, i.e., the attack is robust to the two types of uncertainty at the adversary.

7.5 Conclusion

We considered the power control problem at the BS that uses a DNN to maximize the minimum rate among all UEs while the adversary launches adversarial attacks to minimize the minimum rate among all UEs. We considered various methods such as analytical and DNN gradient-based attacks to craft adversarial perturbations on channel estimates targeting one or multiple UEs subject to the budget on adversarial perturbations. Our results showed that the adversarial attacks on power control can significantly reduce the minimum rate among all UEs by slightly manipulating the channel estimate inputs to the DNN of the BS. These attacks remain effective when we vary the number of subcarriers at the BS and when the adversary is subject to errors regarding channel gains and channel changes.

CHAPTER 8

Covert Communications via Adversarial Machine Learning and Reconfigurable Intelligent Surfaces

8.1 Introduction

Reconfigurable intelligent surfaces (RISs) have emerged as novel tools for software-defined wireless communications to increase the coverage and the spectral efficiency of 5G and beyond wireless communication systems [95–97]. RISs correspond to large number of reflecting antennas that can be controlled to interact with incident signals. Specifically, the phase shifts of the RISs can be controlled without the need of any computing or energy source for decoding, encoding, or transmission. For that purpose, it is necessary to select the best reflection beamforming or interaction vector at the RIS to focus the incident beam towards the receiver. However, this is a complex task as reflection properties (as in phase shifts) need to be optimized for a large number of antenna elements.

DL has been effectively applied to solve the complex task of optimizing the RIS-aided communications [98–102]. The interaction vector at the RIS was designed in [103] by using the channel information as the input to the DNN. Reinforcement learning (RL) was applied in [104] to predict the interaction vector at the RIS without the need for an external source to determine it. A recurrent neural network (RNN) was used in [105] to predict whether to use a direct link or the RIS, and in the latter case to predict the best RIS beam. For indoor communications, DL was used in [106] for the RISs to improve the focus of transmitted signals to receiver positions. Joint design of transmit beamforming at the base station (BS) and phase shift at the RIS was studied in [107] to maximize the sum rate of multiuser downlink MIMO systems with deep RL. A convolutional neural network (CNN) was used in [108] to identify the interfering users from the incident signal at the RIS. The RIS was integrated with autoencoder communications in [109] by training the DNNs for the RIS, the encoder at the transmitter, and the decoder at the receiver.

In this chapter, we consider RIS-aided wireless communications, where a receiver uses its DNN classifier to detect the transmitter's signal that is reflected by the RIS. Concurrently, there exists an eavesdropper with another DNN classifier to identify an ongoing transmission for adversarial purposes. The transmitter adds adversarial attacks to its signals to fool the eavesdropper and reduce its detection accuracy. Minimum power is used for these adversarial perturbations to minimize the effect on the receiver's detection performance. Simultaneously, the RIS interaction vector is designed so that the RIS reflects the signal towards the intended receiver while keeping the reflection away from the eavesdropper.

Note that the prior work on the RISs has typically considered improving the performance (such as the signal-to-noise-ratio (SNR) at the receiver) by optimizing the RIS interaction vector only for the receiver. First, we show that this approach does not guarantee covertness of the signals at the eavesdropper. In particular, while the SNR of the receiver is highly correlated with the receiver's detection accuracy and can effectively guide the optimization of the RIS interaction vector to maximize the receiver performance, it does not reliably tell how to design the RIS to reduce the eavesdropper's detection accuracy. Then, we show how to reduce the eavesdropper's performance by adding adversarial perturbations to the transmitter signals that are reflected by the RIS. For that purpose, we consider different topologies and analyze how the design of the RIS interaction vector for covert communications adapts to different locations of the receiver and the eavesdropper. Our results show that the beam selection of the RIS is the crucial component for covert communications when the transmitter has a low power budget for the adversarial attack. However, when there is enough power budget, the adversarial perturbation becomes the dominant factor to improve covert communications.

8.2 System Model

We consider a communication system where a transmitter is transmitting the signal x while the intended receiver uses a pretrained DNN classifier to detect the ongoing signal that is reflected by the RIS equipped with N reconfigurable antenna elements. The transmitter and the intended receiver have a single antenna each. Concurrently, there exists an eavesdropper with a single antenna that also aims to detect the ongoing signal using another pretrained DNN classifier. To defend against eavesdropping, the transmitter adds a perturbation δ to its signal, which corresponds to an adversarial attack to the eavesdropper. We describe in Section 8.3 how to craft this perturbation for covert communications. In addition, we design the RIS interaction vector $\boldsymbol{\psi}$ so that the adversarial attack becomes most effective on the eavesdropper while minimizing the effect on the classifier at the intended receiver. In other words, the designs of the adversarial perturbation and the RIS interaction vector are coupled, and should be jointly performed. We assume that the RIS interaction vector $\boldsymbol{\psi}$ is selected from a predefined codebook $\mathcal{S}$.

When the transmitter transmits x, the input to the RIS (namely, the incident signal for the RIS) is given by

$$\boldsymbol{x}_{ris}(x) = \boldsymbol{h}_{tr} x, \tag{8.1}$$

where $\boldsymbol{h}_{tr} \in \mathbb{C}^{N\times 1}$ is the channel between the transmitter and the RIS. We assume that the phase shift of the RIS element is quantized and represented with 1 bit where each RIS element introduces either 0° or 180° phase shift and κ loss to the signal. Thus, the signal at the output of RIS is given by

$$[\boldsymbol{y}_{ris}(x)]_i = c_i[\kappa \boldsymbol{x}_{ris}(x)]_i, \quad i = 1, \cdots, N, \tag{8.2}$$

where $c_i \in \{-1, 1\}$ or $c_i = e^{j\theta_i}$ and θ_i corresponds to the phase shifts (e.g., $\theta_i \in \{0, \pi\}$). No noise is added at the RIS (in accordance with previous RIS studies) since it is a passive device. The received signal at the intended receiver is

$$y_r(x) = \boldsymbol{h}_{ri}^T \boldsymbol{y}_{ris}(x) + n_r, \tag{8.3}$$

where n_r is the noise at the intended receiver and $\boldsymbol{h}_{ri} \in \mathbb{C}^{N \times 1}$ is the channel between the RIS and the intended receiver. This channel formulation takes the channel gain and the phase shift between the RIS and the intended receiver into account. Simultaneously, the eavesdropper receives the signal

$$y_{eve}(x) = \boldsymbol{h}_{re}^T \boldsymbol{y}_{ris}(x) + n_e, \tag{8.4}$$

where n_e is the noise at the eavesdropper and $\boldsymbol{h}_{re} \in \mathbb{C}^{N \times 1}$ is the channel between the RIS and the eavesdropper (taking the channel gain and the phase shift between the RIS and the eavesdropper into account). When the transmitter transmits $x + \delta$, the input to the RIS expression of (8.1) changes to

$$\boldsymbol{x}_{ris}(x + \delta) = \boldsymbol{h}_{tr} x + \boldsymbol{h}_{tr} \delta, \tag{8.5}$$

and (8.2), (8.3), and (8.4) change accordingly.

We define the pretrained classifier at the intended receiver as $f_r(.; \boldsymbol{\theta}_r) : \mathcal{X} \to \mathbb{R}^2$, to determine the existence of ongoing background transmission to utilize the idle bands, where $\boldsymbol{\theta}_r$ is the set of transmitter's DNN parameters and $\mathcal{X} \subset \mathbb{C}^M$ is the complex-valued inputs that can be also represented by concatenation of two real-valued inputs. Note that the input to the DNN is defined as $\boldsymbol{y}_r(\boldsymbol{x}) =$

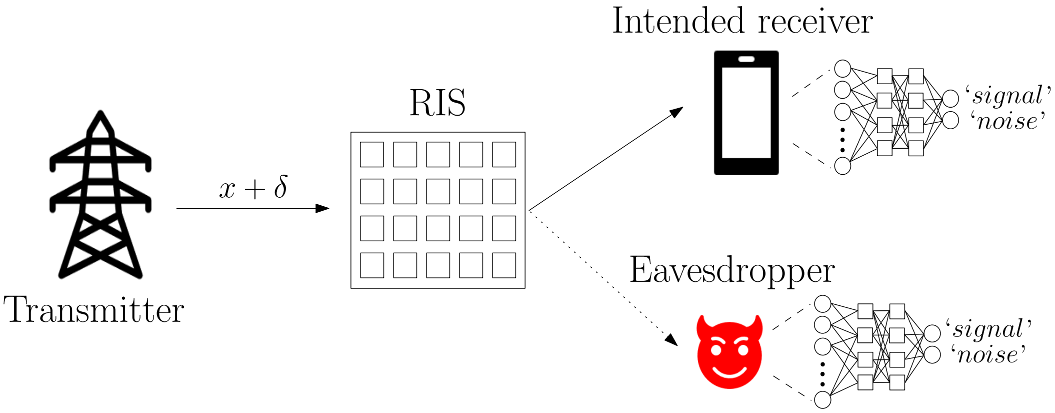


Figure 8.1: RIS-aided communications in the presence of an eavesdropper.

$[y_r(x_1), y_r(x_2), \cdots, y_r(x_M)] \in \mathcal{X}$. The input $\boldsymbol{y}_r(\boldsymbol{x})$ is assigned to the label $\hat{l}_r(\boldsymbol{y}_r(\boldsymbol{x}), \boldsymbol{\theta}_r) = \arg\max_q f_r^{(q)}(\boldsymbol{y}_r(\boldsymbol{x}), \boldsymbol{\theta}_r)$, where $f_r^{(q)}(\boldsymbol{y}_r(\boldsymbol{x}), \boldsymbol{\theta}_r)$ is the output of classifier $f_r^{(q)}$ corresponding to the qth class. Concurrently, the eavesdropper tries to detect the background transmission based on the $\boldsymbol{y}_{eve}(\boldsymbol{x}) = [y_{eve}(x_1), y_{eve}(x_2), \cdots, y_{eve}(x_M)] \in \mathcal{X}$ using its own DNN classifier. We define the classifier of the eavesdropper as $f_{eve}(.; \boldsymbol{\theta}_{eve}) : \mathcal{X} \to \mathbb{R}^2$. The input $\boldsymbol{y}_{eve}(\boldsymbol{x})$ is assigned to the label $\hat{l}_{eve}(\boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{\theta}_{eve}) = \arg\max_q f_{eve}^{(q)}(\boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{\theta}_{eve})$, where $f_{eve}^{(q)}(\boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{\theta}_{eve})$ is the output of classifier $f_{eve}^{(q)}$ corresponding to the qth class.

8.3 Adversarial Attack Against Eavesdropping

In this section, we introduce how to design an adversarial attack against the eavesdropper to cause misclassification. Since the adversarial perturbation that is transmitted at the transmitter is reflected by the RIS before it is received at the eavesdropper, we need to take the RIS interaction vector into account while generating the adversarial attack. If the transmitter transmits $x + \delta$, then the eavesdropper receives

$$y_{eve}(x+\delta) = \boldsymbol{h}_{re}^T \boldsymbol{\Phi} \boldsymbol{h}_{tr}(x+\delta) + n_e, \tag{8.6}$$

where $\boldsymbol{\Phi} = \text{diag}[\phi_1, \phi_2, \cdots, \phi_N] \in \mathbb{C}^{N \times N}$ and $\phi_k = c_k \kappa$.

The transmitter designs the adversarial perturbation vector $\boldsymbol{\delta} \subset \mathbb{C}^M$ to cause misclassification at the eavesdropper while limiting its effect on the intended receiver by designing the RIS interaction vector simultaneously. Thus, the transmitter determines $\boldsymbol{\delta}$ by solving the following optimization problem:

$$\begin{aligned} \arg\min_{\boldsymbol{\delta}} \quad & ||\boldsymbol{\delta}||_2 \\ \text{s.t.} \quad & \hat{l}_{eve}(\boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{\theta}_{eve}) \neq \hat{l}_{eve}(\boldsymbol{y}_{eve}(\boldsymbol{x}+\boldsymbol{\delta}), \boldsymbol{\theta}_{eve}) \\ & ||\boldsymbol{\delta}||_2^2 \leq P_{max}. \end{aligned} \tag{8.7}$$

However, (8.7) is hard to solve due to the nonconvexity of the DNN structure. Therefore, we use fast gradient method (FGM) [110] to linearize the loss function, $L_{eve}(\boldsymbol{\theta}_{eve}, \boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{l})$, of the DNN in the neighborhood of input $\boldsymbol{y}_{eve}(\boldsymbol{x})$, where $\boldsymbol{l}$ is the label vector, and use the linearized loss function for the optimization. In this chapter, we consider a targeted attack against the eavesdropper such that the transmitter designs the perturbation that decreases the loss function of the class 'noise' to enforce a specific misclassification, from label 'signal' to label 'noise', at the eavesdropper by transmitting the perturbation in the opposite direction of the gradient of the loss function $-\nabla_{\boldsymbol{y}_{eve}(\boldsymbol{x})} L_{eve}(\boldsymbol{\theta}_{eve}, \boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{l}^{target})$, where $\boldsymbol{l}^{target}$ is 'noise' class. However, the channels between the nodes change the direction of the attack

Algorithm 15: Crafting the adversarial attack at the transmitter against the eavesdropper.

```
Inputs: $\boldsymbol{y}_{eve}(\boldsymbol{x})$, desired accuracy $\varepsilon_{acc}$, power budget $P_{max}$ and
  eavesdropper's DNN architecture
Initialize: $\varepsilon_{max} \leftarrow \sqrt{P_{max}}, \varepsilon_{min} \leftarrow 0, \boldsymbol{l}^{target} \leftarrow$ 'noise'
$\boldsymbol{\delta}_{norm} = \frac{(\boldsymbol{h}_{re}^{T}\boldsymbol{\Phi}\boldsymbol{h}_{tr})^{*}\nabla_{\boldsymbol{y}_{eve}(\boldsymbol{x})}L_{eve}(\boldsymbol{\theta}_{eve},\boldsymbol{y}_{eve}(\boldsymbol{x}),\boldsymbol{l}^{target})}{(||(\boldsymbol{h}_{re}^{T}\boldsymbol{\Phi}\boldsymbol{h}_{tr})^{*}\nabla_{\boldsymbol{y}_{eve}(\boldsymbol{x})}L_{eve}(\boldsymbol{\theta}_{eve},\boldsymbol{y}_{eve}(\boldsymbol{x}),\boldsymbol{l}^{target})||_{2})}$
if $\hat{l}_{eve}(\boldsymbol{y}_{eve}(\boldsymbol{x}), \boldsymbol{\theta}_{eve}) ==$ 'signal' then
    while $\varepsilon_{max} - \varepsilon_{min} > \varepsilon_{acc}$ do
        $\varepsilon_{avg} \leftarrow (\varepsilon_{max} + \varepsilon_{min})/2$
        $\boldsymbol{x}_{adv} \leftarrow \boldsymbol{y}_{eve}(\boldsymbol{x}) - \varepsilon_{avg}\boldsymbol{h}_{re}^{T}\boldsymbol{\Phi}\boldsymbol{h}_{tr}\boldsymbol{\delta}_{norm}$
        if $\hat{l}_{eve}(\boldsymbol{x}_{adv}, \boldsymbol{\theta}_{eve}) ==$ 'noise' then $\varepsilon_{min} \leftarrow \varepsilon_{avg}$
        else $\varepsilon_{max} \leftarrow \varepsilon_{avg}$
    end
end
$\varepsilon = \varepsilon_{max}$, $\boldsymbol{\delta}^{*} = -\varepsilon\boldsymbol{\delta}_{norm}$
```

$\boldsymbol{\delta}$ that is first sent at the transmitter. Thus, the transmitter takes the effect of the channels and the RIS interaction vector into account by multiplying $(\boldsymbol{h}_{re}^{T}\boldsymbol{\Phi}\boldsymbol{h}_{tr})^{*}$ with the gradient of the loss function as it has been done similarly in [66]. During the adversarial attack generation process, we assume that the transmitter has the information about all channels and the RIS interaction vector. Knowing the channel between the RIS and the eavesdropper is difficult for real systems. This assumption can be relaxed as in [71] to know the channel distribution between the RIS and the eavesdropper instead of the channel instance. The detailed algorithm is presented in Algorithm 15.

The RIS interaction vector is determined from the predefined codebook $\mathcal{S}$ that maximizes the classifier accuracy at the intended receiver while minimizing the classifier accuracy at the eavesdropper. Denote the accuracy of the intended receiver's classifier as $P_{acc,i}(\boldsymbol{x})$ and the accuracy of the eavesdropper's classifier as $P_{acc,e}(\boldsymbol{x})$ when the transmitted signal is $\boldsymbol{x}$. Then, the RIS interaction vector is

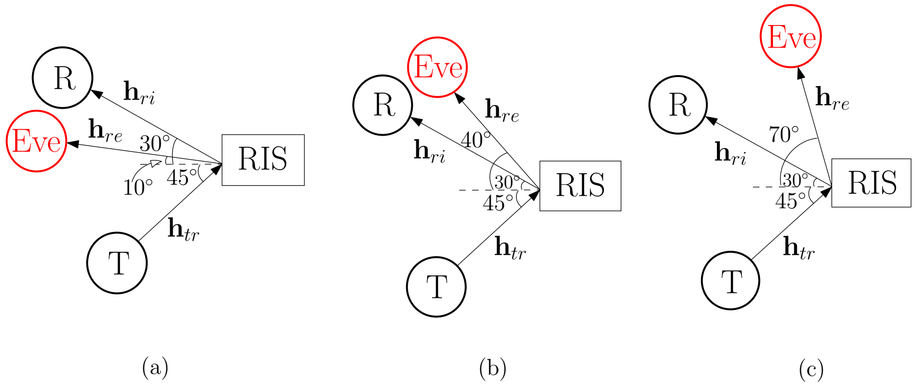


Figure 8.2: Different locations of the eavesdropper 'Eve' while the locations of transmitter 'T', receiver 'R', and the RIS are fixed.

selected as

$$\boldsymbol{\psi}^* = \arg\max_{\boldsymbol{\psi} \in \mathcal{S}} P_{acc,i}(\boldsymbol{x} + \boldsymbol{\delta}) - P_{acc,e}(\boldsymbol{x} + \boldsymbol{\delta}). \tag{8.8}$$

8.4 Performance Evaluation

8.4.1 Topology and Channel Models

We consider different topologies to study the effect of the RIS on the classifier performance at the eavesdropper and assess how the location of the eavesdropper with respect to the location of the intended receiver affects the RIS interaction vector selection according to (8.8). We fix the location of the intended receiver while changing the location of the eavesdropper to analyze how the selection of the RIS interaction vector changes. We define the incident angle of the transmitted signal from the transmitter to the RIS as θ_{tr}, the reflected angle from the RIS to the receiver as θ_{ri}, and the reflected angle from the RIS to the eavesdropper as θ_{re}. We set the angles $\theta_{tr} = 45°$ and $\theta_{ri} = 30°$, and change the location of the eavesdropper by changing the reflected angle from the RIS to the eavesdropper as $\theta_{re} = 10°, 40°, 70°$, as shown in Fig. 8.2. We define channels $\boldsymbol{h}_{tr}, \boldsymbol{h}_{ri}$, and $\boldsymbol{h}_{re}$ according to the wideband geometric channel model adopted in [103], where $\boldsymbol{h}_{tr}$ is given by

$$\boldsymbol{h}_{tr} = \sqrt{\rho_{tr} N} \boldsymbol{a}(\theta_{tr}), \tag{8.9}$$

where the ρ_{tr} is the path loss and $\boldsymbol{a}(\theta_{tr})$ is the array response vector of the RIS at the angles of arrival θ_{tr}, which is defined as $\boldsymbol{a}(\theta_{tr}) = \sqrt{\frac{1}{N}}[1, e^{jd\cos(\theta_{tr})}, \cdots, e^{jd(N-1)\cos(\theta_{tr})}]^T$. The channels $\boldsymbol{h}_{ri}$ and $\boldsymbol{h}_{re}$ are defined similarly. For performance evaluation, we set $N = 16$ and the spacing between reconfigurable antenna elements to the half of the wavelength. The predefined codebook adopts a discrete Fourier transform (DFT) codebook used in [103], where the ith codebook is defined as $\boldsymbol{\psi}_i = [1, e^{j2\pi i/N}, \cdots, e^{j2\pi(N-1)i/N}]^T$.

8.4.2 Deep Learning Classifiers

We assume that the transmitter transmits QPSK signals to the RIS. The classifiers at the receiver and the eavesdropper are modeled as two (different) CNNs, where the input to each CNN is of two dimensions (2,16) corresponding to 16 in-phase/quadrature (I/Q) data samples. The classifier architecture used in the simulations consists of a convolutional layer with kernel size $(1, 3)$, two hidden layers with dropout rate 0.1 with 128 and 64 nodes, ReLu activation function at convolutional and hidden layers, and softmax activation function at the output layer that provides the label ‘signal’ or ‘noise’. We use cross-entropy as the loss function of the CNN that is implemented in Keras with TensorFlow backend. To collect the dataset to train the classifier, we let the transmitter transmit the signals with power 30dBm that are reflected by all possible $K = 16$ RIS interaction vectors, e.g., RIS 1, RIS 2, $\cdots$, RIS 16, and three different SNR levels, e.g., 3dB, 5dB and 7dB, to train the receiver at a specific location. We collect 5000 samples for each RIS interaction vector and SNR level, generating 240000 signal samples. In addition, we generate 240000 noise samples and obtain 480000 samples in total to train and validate the classifier. We use half of the samples for training and the other half for validating the classifier.

8.4.3 Covert Communications Performance

Once we train the classifiers for different locations of the eavesdropper and the receiver, we test the performance of the classifiers at the receiver and the eavesdropper when the transmitter transmits the signal with the adversarial perturbation added to fool the eavesdropper. During the test time, we fix the transmit power of the signal at the transmitter as 30dBm and set the noise power that results in an average of 5dB SNR at the receiver and the eavesdropper. Note that the power for adversarial perturbation at the transmitter is used separately from the transmit power of the signal. We first motivate the need to design the RIS interaction vector differently when the eavesdropper is present. For that purpose, we investigate the correlation between the SNR and the probability of detection at the receiver, and the correlation between the SNR at the receiver and the probability of detection at

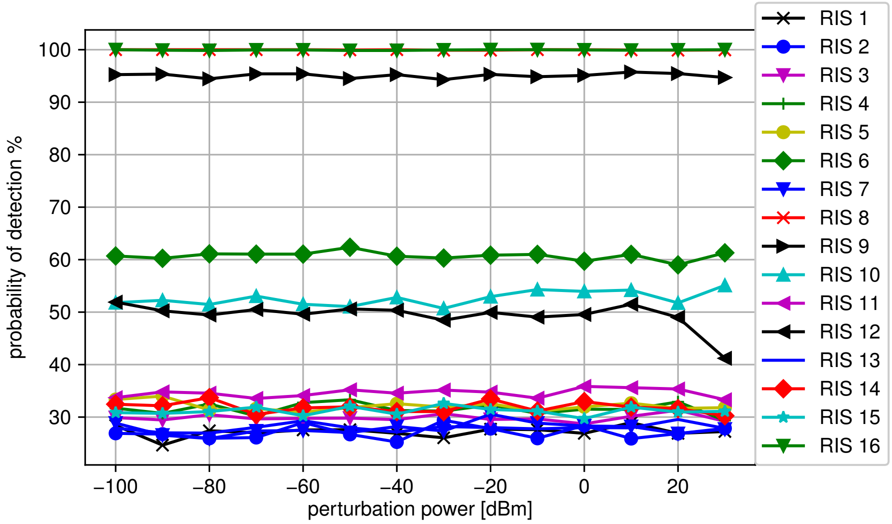


Figure 8.3: Probability of detection for the classifier of the receiver when $\theta_{tr} = 45°$, $\theta_{ri} = 30°$ and $\theta_{re} = 10°$.

the eavesdropper. We measure the correlation by the Pearson correlation coefficient. Without the eavesdropper, the RIS interaction vector is typically selected as the one that maximizes the SNR at the receiver. This selection is expected to yield a high probability of detection at the receiver. As an example, the correlation between the SNR and the probability of detection at the receiver is 0.94 for the topology shown in Fig. 8.2(c). This means that as expected, the SNR is a good measure to design the RIS for signal detection at the receiver. However, the correlation between the SNR at the receiver and the probability of detection at the eavesdropper is 0.69. This means that designing the RIS based on the SNR may be also good for the eavesdropper. Especially for high SNR at the receiver, the eavesdropper maintains moderate probability of detection. Therefore, we need a better criterion to select the RIS interaction vector than just selecting the one with highest SNR at the receiver, and additional means such as adversarial perturbation is needed to enable covert

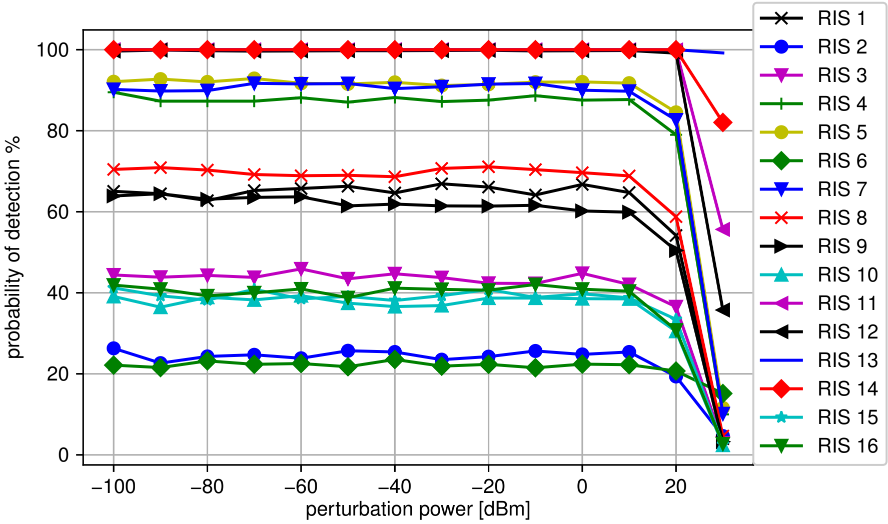


Figure 8.4: Probability of detection for the classifier of the eavesdropper when $\theta_{tr} = 45°$, $\theta_{ri} = 30°$ and $\theta_{re} = 10°$.

communications.

In this section, we investigate how different locations of the eavesdropper described in Section 8.4.1 affect the adversarial perturbation performance and the RIS interaction vector selection. First, we assess the performance of the classifier at the intended receiver with location given in Fig. 8.2(a). Fig. 8.3 shows that the classifier at the receiver is not affected by the adversarial perturbation even when its power is increased. Also, the probability of detection at the receiver differs significantly for different RIS interaction vectors. The probability of detection using RIS 16 and RIS 14 is 100% and 30%, respectively. From Fig. 8.3, the best RIS interaction vectors to select for the receiver are RIS 8 and RIS 16 leading to 100% probability of detection at the receiver.

The performance of the classifier at the eavesdropper with location from Fig. 8.2(a) is presented in Fig. 8.4. The probability of detection at the eavesdropper also

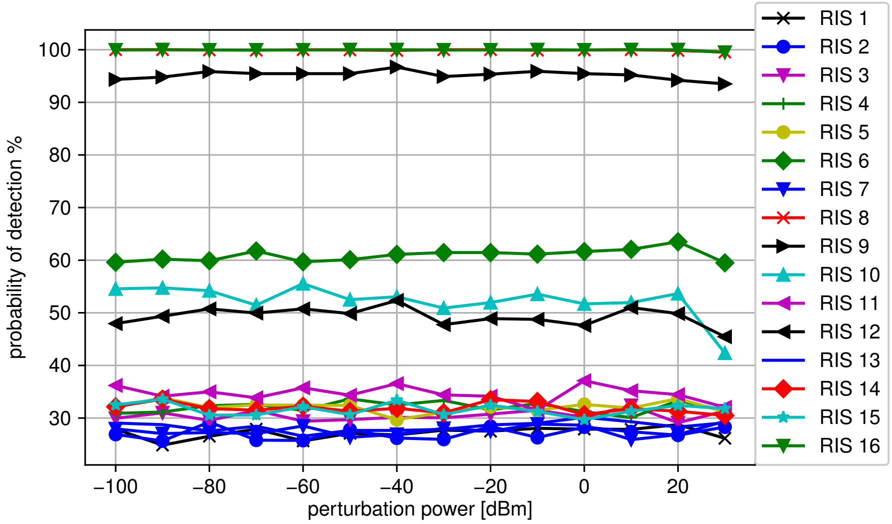


Figure 8.5: Probability of detection for the classifier of the receiver when $\theta_{tr} = 45°$, $\theta_{ri} = 30°$ and $\theta_{re} = 40°$.

differs considerably with respect to different RIS interaction vectors. The adversarial perturbation reduces the probability of detection at the eavesdropper and causes misclassifications very likely when using more than 20dBm perturbation power at the transmitter. For different RIS interaction vectors, the adversarial perturbation has different degrees of effect on the eavesdropper's classifier. In particular, the adversarial perturbation through RIS 12 has more effect on the classifier than the adversarial perturbation through RIS 14. To determine the RIS interaction vector that induces the best probability of detection at the receiver while causing the worst performance at the eavesdropper, we select the RIS interaction vector based on (8.8) by analyzing Fig. 8.3 and Fig. 8.4. For this topology, the best RIS interaction vector to use is RIS 16 and the probability of detection at the eavesdropper can drop to almost zero by jointly designing the RIS interaction vector and the adversarial perturbation.

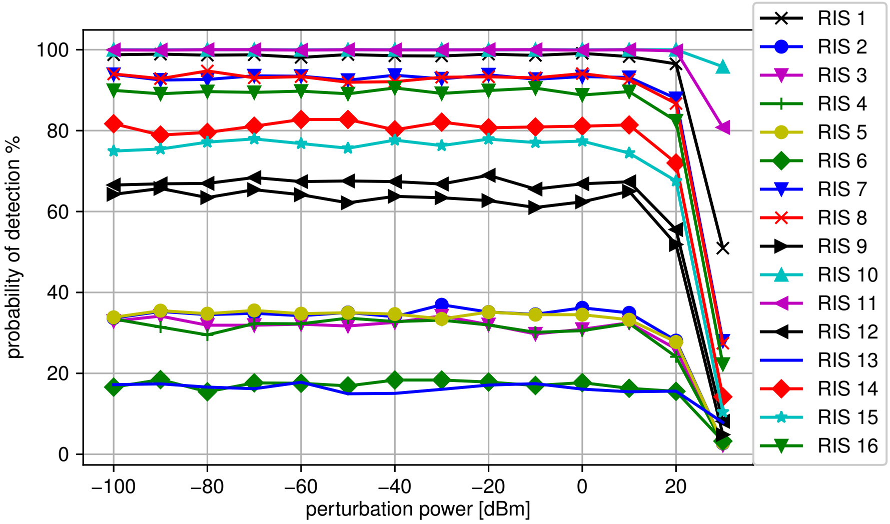


Figure 8.6: Probability of detection for the classifier of the eavesdropper when $\theta_{tr} = 45°$, $\theta_{ri} = 30°$ and $\theta_{re} = 40°$.

Next, we assess the performance for the topology given in Fig. 8.2(b). The performance of the classifier at the receiver in Fig. 8.5 is very similar to the result for the receiver in Fig. 8.3, since the location of the receiver is exactly the same. However, due to the change in location for the eavesdropper, it is observed in Fig. 8.6 that the order of the RIS interaction vector from the highest probability of detection to the lowest has changed compared to Fig. 8.4. Again, to determine the best RIS interaction vector, we analyze Fig. 8.5 and Fig. 8.6, and select the RIS interaction vector that provides the maximum value for the probability of detection difference between the receiver and the eavesdropper. For this topology, the best RIS interaction vectors for the receiver are RIS 8 and RIS 16 without considering the performance of the eavesdropper's classifier. However, RIS 8 and RIS 16 are also good for the eavesdropper, since the probability of detection at the eavesdropper for those RIS interaction vectors is around 90%, yielding around 10% difference between

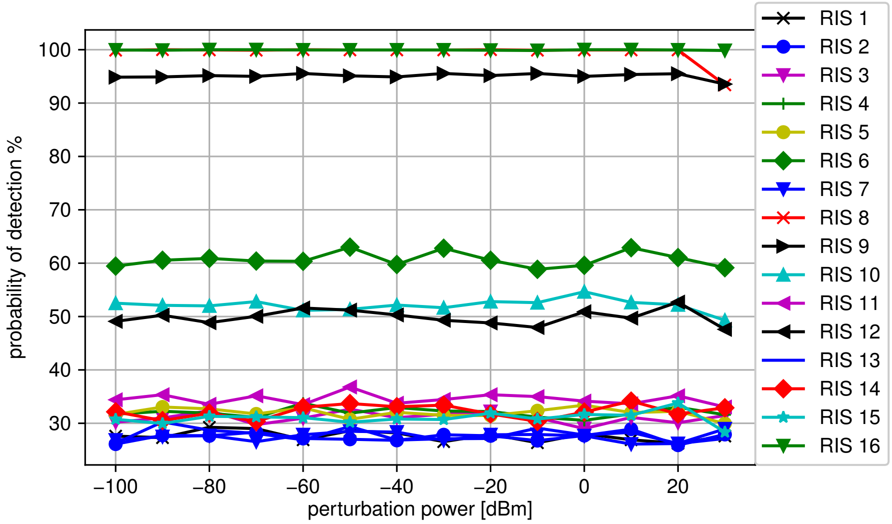


Figure 8.7: Probability of detection for the classifier of the receiver when $\theta_{tr} = 45°$, $\theta_{ri} = 30°$ and $\theta_{re} = 70°$.

the probability of detection at the receiver and eavesdropper. Instead, for covert communications, we need to select RIS 9, which reduces the detection probability at the receiver to 95% compared to 100% for RIS 8 and RIS 16, but RIS 9 enforces the probability of detection at the eavesdropper to drop to 65% without any adversarial perturbation. Furthermore, the probability of detection at the eavesdropper for RIS 9 can be reduced to 10% by adding an adversarial perturbation at the transmitter.

Finally, we assess the performance for the topology given in Fig. 8.2(c). The performance of the classifier at the receiver is similar to the performance in other topologies except that the probability of the detection for RIS 8 decreases when the adversarial perturbation is added with higher power. Fig. 8.8 shows that the order of the RIS interaction vector from the highest probability of detection to the lowest has changed again compared to the order from other topologies. The best RIS interaction vectors at the receiver are RIS 8 and RIS 16 leading to 100%

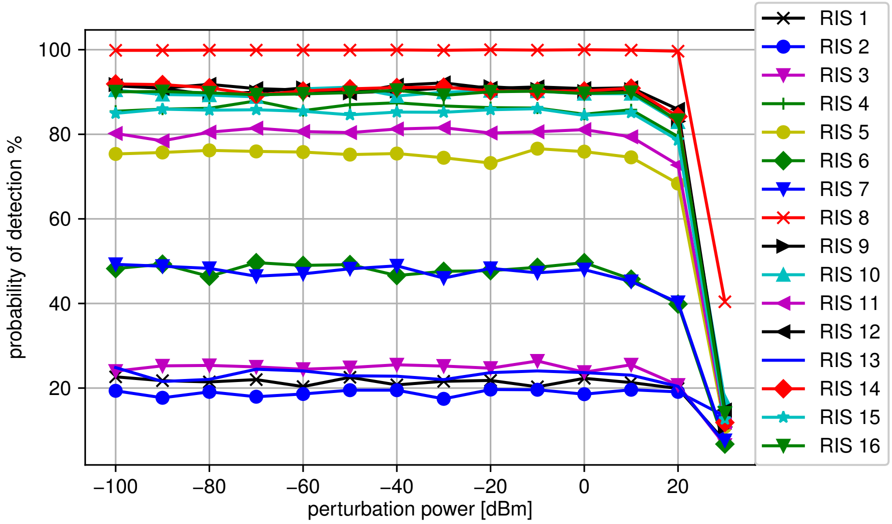


Figure 8.8: Probability of detection for the classifier of the eavesdropper when $\theta_{tr} = 45°$, $\theta_{ri} = 30°$ and $\theta_{re} = 70°$.

probability of detection, but the probability of detection at the eavesdropper using RIS 16 and RIS 8 is 100% and 90%, respectively, without a perturbation. Thus, RIS 16 is chosen over RIS 8, but the eavesdropper still can detect the signal with 90% accuracy. Adversarial perturbation is needed for covertness since the best RIS interaction vector for the receiver is also the best one for the eavesdropper. When RIS 16 is used, the probability of detection at the eavesdropper drops to 10% when the transmitter uses 25dBm power for adversarial perturbation, while the probability of detection at the receiver remains 100%.

8.5 Conclusion

We considered RIS-aided wireless communications where the receiver uses its DNN classifier to detect the ongoing transmission reflected by the RIS. Concurrently, there exists an eavesdropper that also aims to detect the ongoing transmission for its adversarial purposes. To make the communications covert, the transmitter crafts the adversarial perturbation to cause misclassifications at the eavesdropper. In addition, the RIS interaction vector that determines the direction of the reflected signal is designed so that the reflected signal is focused to the receiver while keeping it away from the eavesdropper. Through different topologies, we showed that the design of the RIS interaction vector for covert communications changes with respect to the location of not only the receiver but also the eavesdropper. Moreover, the adversarial perturbation that is generated at the transmitter further improves the covertness of communications and has only a negligible effect on the receiver performance.

CHAPTER 9

Conclusions

In this dissertation, we studied adversarial machine learning in wireless communications systems.

In Chapter 2, we presented over-the-air adversarial attacks against deep learning-based modulation classifiers by accounting for realistic channel and broadcast transmission effects. Specifically, we considered targeted, non-targeted and UAP attacks with different levels of uncertainty regarding channels, transmitter inputs, and DNN classifier models. We showed that these channel-aware adversarial attacks can successfully fool a modulation classifier over the air. Then, we introduced broadcast adversarial attacks to simultaneously fool multiple classifiers at different receivers with a single perturbation transmission. Finally, we presented a certified defense method using randomized smoothing, and showed that it is effective in reducing the impact of adversarial attacks on the modulation classifier performance.

In Chapter 3, we considered a wireless communication system where a DL-based signal classifier is used at the receiver to classify signals transmitted from the transmitter to their modulation types and showed that different methods to craft adversarial perturbations can be used to exploit multiple antennas at the adversary. We showed that just adding more antennas at the adversary does not always improve the attack. Thus, it is important to carefully allocate power among antennas, determine the adversarial perturbation for each antenna, and exploit channel diversity to select which antenna to transmit. In this context, the proposed EMCG attack significantly outperforms other attacks and effectively uses multiple antennas to evade the target classifier over the air.

In Chapter 4, we considered a wireless communications system in which a CJ with multiple antennas transmits perturbation signals to fool a DL-based classifier at the eavesdropper into classifying the ongoing transmissions as noise. Following the AML approach, the CJ was designed to generate the perturbation signal with different methods. For both basic modulated signals and sophisticated 5G signals, we showed that the CJ could generate a perturbation signal that caused misclassification at the eavesdropper (from *signal* to *noise*) with high success, while the BER at the receiver was only slightly affected. Furthermore, we showed that adding more antennas at the CJ always improved the attack performance and lowered the BER when the EMCG attack was used.

In Chapter 5, we considered a wireless communication system where an adversary transmits a perturbation signal to fool the DNN classifier at a transmitter into classifying the ongoing background transmission as noise. The adversary trains its surrogate model by observing the spectrum and uses this model to design the adversarial attack. Through different topologies, we showed how the adversary's location significantly affects the attack performance as the surrogate model may differ from the target model due to channel discrepancies. In particular, the attack success against the transmitter drops when the adversary moves away from the background emitter (and the surrogate model becomes less reliable) although the adversary does not necessarily move closer to the transmitter.

In Chapter 6, we presented an adversarial attack to fool the beam selection process of the IA based on a DNN classifier that uses only a subset of beams to predict the beam that is best oriented to the receiver. We investigated two different attacks, namely, the non-targeted FGM attack that only aims to fool the DNN classifier with misclassification to any other beam label, and the k-worst beam attack that not only fools the DNN classifier but also enforces the label that is chosen at the DNN to be in one of the k-worst beams. We showed that the adversarial attack can significantly decrease the accuracy of the DNN and fool the DNN into selecting the worst beam.

In Chapter 7, we considered the power control problem at the BS that uses a DNN to maximize the minimum rate among all UEs while the adversary launches adversarial attacks to minimize the minimum rate among all UEs. We considered various methods such as analytical and DNN gradient-based attacks to craft adversarial perturbations on channel estimates targeting one or multiple UEs subject to the budget on adversarial perturbations. Our results showed that the adversarial attacks on power control can significantly reduce the minimum rate among all UEs by slightly manipulating the channel estimate inputs to the DNN of the BS.

In Chapter 8, we considered RIS-aided wireless communications where the receiver uses its DNN classifier to detect the ongoing transmission reflected by the

RIS. Concurrently, there exists an eavesdropper that also tries to detect the ongoing transmission for its adversarial purposes. To make the communications covert, the transmitter crafts the adversarial perturbation to cause misclassifications at the eavesdropper. In addition, the RIS interaction vector that determines the direction of the reflected signal is designed so that the reflected signal is focused to the receiver while keeping it away from the eavesdropper. Through different topologies, we showed that the design of the RIS interaction vector for covert communications changes with respect to the location of not only the receiver but also the eavesdropper. Moreover, the adversarial perturbation that is generated at the transmitter further improves the covertness of communications and has only a negligible effect on the receiver performance.

The contents of Chapter 2 are published in [66,71], Chapter 3 in [72], Chapter 4 in [111,112], Chapter 5 in [73], Chapter 6 in [78], Chapter 7 in [80], Chapter 8 in [113].

Bibliography

[1] I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. MIT press, 2016.

[2] T. Erpek, T.J. O'Shea, Y. E. Sagduyu, Y. Shi, and T. C. Clancy. Deep learning for wireless communications. in development and analysis of deep learning architectures. pages 223–266. Springer, 2020.

[3] K. Davaslioglu and Y. E. Sagduyu. Generative adversarial learning for spectrum sensing. In *IEEE International Conference on Communications (ICC)*, 2018.

[4] X. Song, J. Wang, J. Wang, G. Gui, T. Ohtsuki, H. Gacanin, and H. Sari. Saldr: Joint self-attention learning and dense refine for massive MIMO CSI feedback with multiple compression ratio. *IEEE Wireless Communications Letters*, 2021.

[5] J. Wang, G. Gui, T. Ohtsuki, B. Adebisi, H. Gacanin, and H. Sari. Compressive sampled CSI feedback method based on deep learning for FDD massive MIMO systems. *IEEE Transactions on Communications*, 2021.

[6] Y. Jin, J. Zhang, S. Jin, and B. Ai. Channel estimation for cell-free mmWave massive MIMO through deep learning. *IEEE Transactions on Vehicular Technology*, 68(10):10325–10329, 2019.

[7] T. J. O'Shea, J. Corgan, and T. C. Clancy. Convolutional radio modulation recognition networks. In *Int. Conf. on Engineering Applications of Neural Networks*, 2016.

[8] T. J. O'Shea, T. Roy, and T. C. Clancy. Over-the-air deep learning based radio signal classification. *IEEE J. Sel. Topics Signal Process.*, 12(1):168–179, January 2018.

[9] Y. Shi, K. Davaslioglu, Y. E. Sagduyu, W. C. Headley, M. Fowler, and G. Green. Deep learning for signal classification in unknown and dynamic spectrum environments. In *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2019.

[10] S. Soltani, Y. E. Sagduyu, R. Hasan, Davaslioglu K, H. Deng, and T. Erpek. Real-time and embedded deep learning on FPGA for RF signal classification. In *IEEE Military Communications Conference (MILCOM)*, 2019.

[11] K. Davaslioglu, S. Soltani, T. Erpek, and Y. E. Sagduyu. DeepWiFi: Cognitive WiFi with deep learning. *IEEE Transactions on Mobile Computing*, 2019.

[12] Y. Wang, G. Gui, T. Ohtsuki, and F. Adachi. Multi-task learning for generalized automatic modulation classification under non-Gaussian noise with varying SNR conditions. *IEEE Transactions on Wireless Communications*, 20(6):3587–3596, 2021.

[13] Y. Wang, J. Gui, Y. Yin, J. Wang, J. Sun, G. Gui, H. Gacanin, H. Sari, and F. Adachi. Automatic modulation classification for MIMO systems via deep learning and zero-forcing equalization. *IEEE Transactions on Vehicular Technology*, 69(5):5688–5692, 2020.

[14] Y. Lin, Y. Tu, and Z. Dou. An improved neural network pruning technology for automatic modulation classification in edge devices. *IEEE Transactions on Vehicular Technology*, 69(5):5703–5706, 2020.

[15] Y. Vorobeychik and M. Kantarcioglu. Adversarial machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–169, December 2017.

[16] C. Szegedy, W. Zaremba, I Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. 2013. Available on arXiv: 1312.6199.

[17] Q. Huang, I. Katsman, H. He, Z. Gu, S. Belongie, and S. Lim. Enhancing adversarial example transferability with an intermediate level attack. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4733–4742, 2019.

[18] B. Luo, Y. Liu, L. Wei, and Q. Xu. Towards imperceptible and robust adversarial example attacks against neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[19] G. Elsayed, S. Shankar, B. Cheung, N. Papernot, A. Kurakin, I. Goodfellow, and J. Sohl-Dickstein. Adversarial examples that fool both computer vision and time-limited humans. *Advances in neural information processing systems*, 31, 2018.

[20] B. Flowers, R. M. Buehrer, and W. C. Headley. Evaluating adversarial evasion attacks in the context of wireless communications. 2019. Available on arXiv:1903.01563.

[21] S. Bair, M. Delvecchio, B. Flowers, A. J. Michaels, and W. C. Headley. On the limitations of targeted adversarial evasion attacks against deep learning enabled modulation recognition. In *ACM WiSec Workshop on Wireless Security and Machine Learning (WiseML)*, 2019.

[22] S. Kokalj-Filipovic and R. Miller. Adversarial examples in RF deep learning: detection of the attack and its physical robustness. 2019. Available on arXiv:1902.06044.

[23] S. Kokalj-Filipovic and R. Miller. Targeted adversarial examples against RF deep classifiers. In *ACM WiSec Workshop on Wireless Security and Machine Learning (WiseML)*, 2019.

[24] S. Kokalj-Filipovic, R. Miller, and G. M. Vanhoy. Adversarial examples in RF deep learning: Detection and physical robustness. In *GlobalSIP*, 2019.

[25] S. Kokalj-Filipovic, R. Miller, N. Chang, and C. L. Lau. Mitigation of adversarial examples in RF deep classifiers utilizing autoencoder pre-training. In *2019 International Conference on Military Communications and Information Systems (ICMCIS)*, pages 1–6, 2019.

[26] D. Ke, Z. Huang, X. Wang, and L. Sun. Application of adversarial examples in communication modulation classification. In *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 877–882, 2019.

[27] Y. Lin, H. Zhao, Y. Tu, S. Mao, and Z. Dou. Threats of adversarial attacks in DNN-based modulation recognition. In *International Conference on Computer Communications (INFOCOM)*, 2020.

[28] Y. Lin, H. Zhao, X. Ma, Y. Tu, and M. Wang. Adversarial attacks in modulation recognition with convolutional neural networks. *IEEE Transactions on Reliability*, 2020.

[29] M. Z. Hameed, A. Gyorgy, and D. Gunduz. The best defense is a good offense: Adversarial attacks to avoid modulation detection. 2019. Available on arXiv: 1902.10674.

[30] M. Z. Hameed, A. Gyorgy, and D. Gunduz. Communication without interception: Defense against modulation detection. In *GlobalSIP*, 2019.

[31] Q. Liu, J. Guo, C. Wen, and S. Jin. Adversarial attack on dl-based massive mimo csi feedback. *Journal of Communications and Networks*, 22(3):230–235, 2020.

[32] S. Bairl, M. DelVecchio, B. Flowersand A. J. Michaels, and W. C. Headley. 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*, pages 25–30, 2019.

[33] M. Usama, I. Ilahi, J. Qadir, R. N. Mitra, and M. K. Marina. Examining machine learning for 5g and beyond through an adversarial lens. *IEEE Internet Computing*, 25(2):26–34, 2021.

[34] M. Usama, M. Asim, J. Qadir, A. Al-Fuqaha, and M. A. Imran. Adversarial machine learning attack on modulation classification. In *2019 UK/China Emerging Technologies (UCET)*, pages 1–4. IEEE, 2019.

[35] Y. E. Sagduyu, Y. Shi, T. Erpek, W. Headley, B. Flowers, G. Stantchev, and Z. Lu. When wireless security meets machine learning: Motivation, challenges, and research directions. 2020. Available on arXiv:2001.08883.

[36] D. Adesina, C. Hsieh, Y. E. Sagduyu, and L. Qian. Adversarial machine learning in wireless communications using RF data: A review. *arXiv preprint arXiv:2012.14392*, 2020.

[37] T. Erpek, Y. E. Sagduyu, and Y. Shi. Deep learning for launching and mitigating wireless jamming attacks. *IEEE Transactions on Cognitive Communications and Networking*, 5(1):2–14, March 2019.

[38] Y. Shi, Y. E Sagduyu, T. Erpek, K. Davaslioglu, Z. Lu, and J. Li. Adversarial deep learning for cognitive radio security: Jamming attack and defense strategies. In *IEEE International Communications Conference (ICC) Workshop on Machine Learning in Wireless Communications*, 2018.

[39] E. Yang, S. Fang, I. Markwood, Y. Liu, S. Zhao, Z. Lu, and H. Zhu. Wireless training-free keystroke inference attack and defense. *IEEE/ACM Transactions on Networking*, 2022.

[40] Y. Shi, T. Erpek, Y. E Sagduyu, and J. Li. Spectrum data poisoning with adversarial deep learning. In *IEEE Military Communications Conference (MILCOM)*, 2018.

[41] Y. E. Sagduyu, T. Erpek, and Y. Shi. Adversarial deep learning for over-the-air spectrum poisoning attacks. *IEEE Transactions on Mobile Computing*, (1):2–14, Feb. 2021.

[42] T. Zheng and B. Li. Poisoning attacks on deep learning based wireless traffic prediction. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 660–669. IEEE, 2022.

[43] K. Davaslioglu and Y. E. Sagduyu. Trojan attacks on wireless signal classification with adversarial machine learning. In *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN) Workshop on Data-Driven Dynamic Spectrum Sharing*, 2019.

[44] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu. Over-the-air membership inference attacks as privacy threats for deep learning-based wireless signal classifiers. In *ACM WiSec Workshop on Wireless Security and Machine Learning (WiseML)*, 2020.

[45] Y. Shi and Y. Sagduyu. Membership inference attack and defense for wireless signal classifiers with deep learning. *IEEE Transactions on Mobile Computing*, 2022.

[46] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu. Generative adversarial network for wireless signal spoofing. In *ACM Workshop on Wireless Security and Machine Learning (WiseML)*, 2019.

[47] Y. Shi, K. Davaslioglu, and Y. E. Sagduyu. Generative adversarial network in the air: Deep adversarial learning for wireless signal spoofing. *IEEE Transactions on Cognitive Communications and Networking*, 7(1):294–303, March 2021.

[48] Y. E. Sagduyu, T. Erpek, and Y. Shi. Adversarial machine learning for 5g communications security. *Game Theory and Machine Learning for Cyber Security*, pages 270–288, 2021.

[49] Y. Shi, Y. E. Sagduyu, T. Erpek, and C. Gursoy. How to attack and defend 5G radio access network slicing with reinforcement learning. *arXiv preprint arXiv:2101.05768*, 2021.

[50] Y. Shi and Y. E. Sagduyu. Adversarial machine learning for flooding attacks on 5G radio access network slicing. In *IEEE International Conference on Communications (ICC) Workshops)*, 2021.

[51] M. Sadeghi and E. G. Larsson. Physical adversarial attacks against end-to-end autoencoder communication systems. *IEEE Commun. Lett.*, 23(5):847–850, May 2019.

[52] T. J. O'Shea and J. Hoydis. An introduction to deep learning for the physical layer. *IEEE Transactions on Cogn. Comm. and Netw.*, 3(4):563–575, December 2017.

[53] P. F. De Araujo-Filho, G. Kaddoum, M. Naili, E. T. Fapi, and Z. Zhu. Multi-objective gan-based adversarial attack technique for modulation classifiers. *IEEE Communications Letters*, 2022.

[54] J. Yi and A. E. Gamal. Gradient-based adversarial deep modulation classification with data-driven subsampling. *arXiv preprint arXiv:2104.06375*, 2021.

[55] E. Tekin and A. Yener. The general Gaussian multiple-access and two-way wiretap channels: Achievable rates and cooperative jamming. *IEEE Transactions on Information Theory*, 54(6):2735–2751, 2008.

[56] J. Xie and S. Ulukus. Secure degrees of freedom of multiuser networks: One-time-pads in the air via alignment. *Proceedings of the IEEE*, 103(10):1857–1873, Oct 2015.

[57] R. Bassily, E. Ekrem, X. He, E. Tekin, J. Xie, M. R. Bloch, S. Ulukus, and A. Yener. Cooperative security at the physical layer: A summary of recent advances. *IEEE Signal Processing Magazine*, 30(5):16–28, Sep. 2013.

[58] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *ICLR*, 2017.

[59] M. Sadeghi and E. G. Larsson. Adversarial attacks on deep-learning based radio signal classification. *IEEE Commun. Lett.*, 8(1):213–216, February 2019.

[60] J. M. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320, 2019.

[61] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

[62] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin. On evaluating adversarial robustness. 2019. Available on arXiv:1902.06705.

[63] T. J. O'Shea and N. West. Radio machine learning dataset generation with GNU radio. In *Proc. of the 6th GNU Radio Conf.*, 2016.

[64] N. Papernot, P. McDaniel, and I. Goodfellow. Transferability in machine learning: From phenomena to blackbox attacks using adversarial samples. 2016. Available on arXiv: 1605.07277.

[65] J.-Y. Franceschi, A. Fawzi, and O. Fawzi. Robustness of classifiers to uniform ℓ_p and Gaussian noise. In *21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, Feb. 2018.

[66] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus. Over-the-air adversarial attacks on deep learning based modulation classifier over wireless channels. In *Conference on Information Sciences and Systems (CISS)*, 2020.

[67] M. R. Bloch. Covert communication over noisy channels: A resolvability perspective. *IEEE Transactions on Information Theory*, 62(5):2334–2354, 2016.

[68] L. Wang, G. W. Wornell, and L Zheng. Fundamental limits of communication with low probability of detection. *IEEE Transactions on Information Theory*, 62(6):3493–3503, 2016.

[69] B. A. Bash, D. Goeckel, D. Towsley, and S. Guha. Hiding information in noise: fundamental limits of covert wireless communication. *IEEE Communications Magazine*, 12(53):26–31, Dec. 2015.

[70] P. Mukherjee and S. Ulukus. Covert bits through queues. In *IEEE CNS*, pages 626–630, Oct 2016.

[71] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus. Channel-aware adversarial attacks against deep learning-based wireless signal classifiers. *IEEE Transactions on Wireless Communications*, 21(6):3868–3880, 2022.

[72] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus. Adversarial attacks with multiple antennas against deep learning-based modulation classifiers. In *IEEE Global Communications Conference (Globecom)*, 2020.

[73] B. Kim, Y. E. Sagduyu, T. Erpek, K. Davaslioglu, and S. Ulukus. Channel effects on surrogate models of adversarial attacks against wireless signal classifiers. In *ICC 2021-IEEE International Conference on Communications*, pages 1–6. IEEE, 2021.

[74] F. Restuccia, S. D'Oro, A. Al-Shawabka, B. C. Rendon, K. Chowdhury, S. Ioannidis, and T. Melodia. Generalized wireless adversarial deep learning. In *ACM Workshop on Wireless Security and Machine Learning (WiseML)*, 2020.

[75] T. Hou, T. Wang, Z. Lu, Y. Liu, and Y. E. Sagduyu. Iotgan: Gan powered camouflage against machine learning based iot device identification. In *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021.

[76] R. Sahay, D. J. Love, and C. G. Brinton. Robust automatic modulation classification in the presence of adversarial attacks. In *2021 55th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2021.

[77] J. Seo, S. Park, and J. Kang. Adversarial, yet friendly signal design for secured wireless communication. In *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–7. IEEE, 2021.

[78] B. Kim, Y. Sagduyu, T. Erpek, and S. Ulukus. Adversarial attacks on deep learning based mmwave beam prediction in 5g and beyond. In *IEEE Statistical Signal Processing Workshop (SSP)*, 2021.

[79] T. Hou, T. Wang, Z. Lu, Y. Liu, and Y. E. Sagduyu. Undermining deep learning based channel estimation via adversarial wireless signal fabrication. In *ACM Workshop on Wireless Security and Machine Learning (WiseML)*, 2022.

[80] B. Kim, Y. Shi, Y. E. Sagduyu, T. Erpek, and S. Ulukus. Adversarial attacks against deep learning based power control in wireless communications. In *IEEE Globecom*, 2021.

[81] M. Z. Hameed, A. Gyorgy, and D. Gunduz. Communication without interception: Defense against modulation detection. In *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2019.

[82] Y. E. Sagduyu, R. Berry, and A. Ephremides. Jamming games in wireless networks with incomplete information. *IEEE Commun. Soc. Mag.*, 2008.

[83] C. N. Barati, S. A. Hosseini, M. Mezzavilla, T. Korakis, S. S. Panwar, S. Rangan, and M. Zorzi. Initial access in millimeter wave cellular systems. *IEEE Transactions on Wireless Communications*, 15(12):7926–7940, 2016.

[84] M. Giordani, M. Mezzavilla, and M. Zorzi. Initial access in 5g mmwave cellular networks. *IEEE communications Magazine*, 54(11):40–47, 2016.

[85] M. S. Sim, Y. Lim, S. H. Park, L. Dai, and C. Chae. Deep learning-based mmwave beam selection for 5g nr/6g with sub-6 ghz channel information: Algorithms and prototype validation. *IEEE Access*, 8:51634–51646, 2020.

[86] S. Rezaie, E. De Carvalho, and C. N. Manchón. A deep learning approach to location-and orientation-aided 3d beam selection for mmwave communications. *IEEE Transactions on Wireless Communications*, 2022.

[87] A. Alkhateeb, Y. H. Nam, M. S. Rahman, J. Zhang, and R. W. Heath. Initial beam association in millimeter wave cellular systems: Analysis and design insights. *IEEE Trans. on Wireless Commun.*, 2017.

[88] T. S. Cousik, V. K. Shah, T. Erpek, Y. E. Sagduyu, and J. H. Reed. Deep learning for fast and reliable initial access in AI-Driven 6G mmwave networks. *arXiv preprint arXiv:2101.01847*, 2021.

[89] T. S. Cousik, V. K. Shah andJ. H. Reed, T. Erpek, and Y. E. Sagduyu. Fast initial access with deep learning for beam prediction in 5G mmwave networks. *arXiv preprint arXiv:2006.12653*, 2020.

[90] L. Sanguinetti, A. Zappone, and M. Debbah. Deep learning power allocation in massive mimo. In *2018 52nd Asilomar conference on signals, systems, and computers*, pages 1257–1261. IEEE, 2018.

[91] Y. Zhao, I. G. Niemegeers, and S. H. De Groot. Power allocation in cell-free massive mimo: A deep learning method. *IEEE Access*, 8:87185–87200, 2020.

[92] G. Qian, Z. Li, C. He, X. Li, and X. Ding. Power allocation schemes based on deep learning for distributed antenna systems. *IEEE Access*, 8:31245–31253, 2020.

[93] B. R. Manoj, M. Sadeghi, and E. G. Larsson. Adversarial attacks on deep learning based power allocation in a massive MIMO network. In *ICC 2021-IEEE International Conference on Communications*, 2021.

[94] Q. Shi, W. Xu, D. Li, Y. Wang, X. Gu, and W. Li. On the energy efficiency optimality of ofdma for siso-ofdm downlink system. *IEEE Communications Letters*, 17(3):541–544, 2013.

[95] C. Liaskos, S. Nie, A. Tsioliaridou, A. Pitsillides, S. Ioannidis, and I. Akyildiz. A new wireless communication paradigm through software-controlled metasurfaces. *IEEE Commun. Mag.*, Sept. 2018.

[96] E. Basar, M. Di Renzoand, J. De Rosny, M. Debbah, M.-S. Alouini, and R. Zhang. Wireless communications through reconfigurable intelligent surfaces. *IEEE Access*, Aug. 2019.

[97] S. Hu, F. Rusek, and O. Edfors. Beyond massive MIMO: The potential of data transmission with large intelligent surfaces. *IEEE Trans. Signal Process.*, May 2018.

[98] S. Khan, K. S. Khan, N. Haider, and S. Y. Shin. Deep-learning-aided detection for reconfigurable intelligent surfaces. *arXiv preprint arXiv:1910.09136*, 2019.

[99] B. Sheen, J. Yang, X. Feng, and M. M. U. Chowdhury. A deep learning based modeling of reconfigurable intelligent surface assisted wireless communications for phase shift configuration. *IEEE Open Journal of the Communications Society*, 2:262–272, 2021.

[100] S .Zhang, S. Zhang, F. Gao, J. Ma, and O. A. Dobre. Deep learning optimized sparse antenna activation for reconfigurable intelligent surface assisted communication. *IEEE Transactions on Communications*, 69(10):6691–6705, 2021.

[101] B. Sagir, E. Aydin, and H. Ilhan. Deep-learning assisted reconfigurable intelligent surfaces for cooperative communications. *arXiv preprint arXiv:2201.10648*, 2022.

[102] K. Stylianopoulos, N. Shlezinger, P. Del Hougne, and G. C. Alexandropoulos. Deep-learning-assisted configuration of reconfigurable intelligent surfaces in dynamic rich-scattering environments. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8822–8826. IEEE, 2022.

[103] A. Taha, M. Alrabeiah, and A. Alkhateeb. Enabling large intelligent surfaces with compressive sensing and deep learning. *IEEE Access*, Mar. 2021.

[104] A. Taha, Y. Zhang, F. B. Mismar, and A. Alkhateeb. Deep reinforcement learning for intelligent reflecting surfaces: Towards standalone operation. In *IEEE SPAWC*, 2020.

[105] N. Abuzainab, M. Alrabeiah, A. Alkhateeb, and Y. E. Sagduyu. Deep learning for THz drones with flying intelligent surfaces: Beam and handoff prediction. In *IEEE ICC Workshops*, 2021.

[106] C. Huang, G. C. Alexandropoulos, C. Yuen, and M. Debbah. Indoor signal focusing with deep learning designed reconfigurable intelligent surfaces. In *IEEE SPAWC*, 2019.

[107] C. Huang, R. Mo, and C. Yuen. Reconfigurable intelligent surface assisted multiuser MISO systems exploiting deep reinforcement learning. *IEEE J. Sel. Areas Commun.*, Aug. 2020.

[108] B. Yang, X. Cao, C. Huang, C. Yuen, L. Qian, and M. Di Renzo. Intelligent spectrum learning for wireless networks with reconfigurable intelligent surfaces. *IEEE Trans. Veh. Technol.*, Apr. 2021.

[109] T. Erpek, Y. E. Sagduyu, A. Alkhateeb, and A. Yener. Autoencoder-based communications with reconfigurable intelligent surfaces. In *IEEE DySPAN*, 2021.

[110] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[111] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus. How to make 5G communications "invisible" adversarial machine learning for wireless privacy. In *Asilomar Conference*, 2020.

[112] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek, and S. Ulukus. Adversarial machine learning for nextg covert communications using multiple antennas. *Entropy*, 24(8):1047, 2022.

[113] B. Kim, T. Erpek, Y. E. Sagduyu, and S. Ulukus. Covert communications via adversarial machine learning and reconfigurable intelligent surfaces. In *IEEE Wireless Communications and Networking Conference (WCNC)*, 2022.