University of California
Santa Barbara

Signal Models for Robust Deep Learning

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Soorya Gopalakrishnan

Committee in charge:

Professor Upamanyu Madhow, Chair
Professor Kenneth Rose
Professor Ramtin Pedarsani
Professor Naveen Verma, Princeton University

March 2020

The Dissertation of Soorya Gopalakrishnan is approved.

Professor Kenneth Rose

Professor Ramtin Pedarsani

Professor Naveen Verma, Princeton University

Professor Upamanyu Madhow, Committee Chair

March 2020

Signal Models for Robust Deep Learning

by

Soorya Gopalakrishnan

Acknowledgements

I would like to express my gratitude to Prof. Upamanyu Madhow for his guidance and support throughout the course of my PhD. His vision, insights and enthusiasm have been invaluable in shaping my research. I thank Prof. Kenneth Rose, Prof. Ramtin Pedarsani and Prof. Naveen Verma for serving on my committee. I am fortunate to have had the opportunity to collaborate with Prof. Ramtin Pedarsani and Prof. Naveen Verma. I am grateful to them, and to my collaborators Zhinus Marzi, Metehan Cekic and Can Bakiskan, for many insightful research discussions over the years. I appreciate the help, company and friendship of my labmates Ahmet, Anant, Aseem, Babak, Can, Faruk, Maryam, Metehan, Mohammed and Zhinus.

Curriculum Vitæ

Soorya Gopalakrishnan

Education

2020	Ph.D. in Electrical and Computer Engineering, University of California, Santa Barbara.
2014	B. Tech. and M. Tech. in Electrical Engineering, Indian Institute of Technology, Madras.

Publications

- S. Gopalakrishnan, Z. Marzi, M. Cekic, U. Madhow, and R. Pedarsani, "Robust adversarial learning via sparsifying front ends," *arXiv:1810.10625*.
- M. Cekic*, S. Gopalakrishnan*, and U. Madhow, "Robust wireless fingerprinting: Generalizing across space and time," *arXiv:2002.10791*.
- C. Bakiskan, S. Gopalakrishnan, M. Cekic, U. Madhow, and R. Pedarsani, "Polarizing front ends for robust CNNs," to appear in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020.
- S. Gopalakrishnan*, M. Cekic*, and U. Madhow, "Robust wireless fingerprinting via complex-valued neural networks," in *IEEE Global Communications Conference (Globecom)*, Waikoloa, Hawaii, December 2019.
- Z. Marzi*, S. Gopalakrishnan*, U. Madhow, and R. Pedarsani, "Sparsity-based defense against adversarial attacks on linear classifiers,", in *IEEE International Symposium on Information Theory (ISIT)*, Vail, Colorado, June 2018, pp. 31–35.
- S. Gopalakrishnan*, Z. Marzi*, U. Madhow, and R. Pedarsani, "Combating adversarial attacks using sparse representations," in *International Conference on Learning Representations (ICLR) Workshop Track*, Vancouver, Canada, April 2018.
- S. Gopalakrishnan, T. Moy, U. Madhow, and N. Verma, "Compressive information acquisition with hardware impairments and constraints: A case study," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, Louisiana, March 2017, pp. 6075–6079.

* Joint first authors.

Abstract

Signal Models for Robust Deep Learning

by

Soorya Gopalakrishnan

In this thesis, we illustrate via two case studies the utility of bottom-up signal modeling and processing for learning robust models. A key feature of machine learning is the ability to avoid detailed signal models by leveraging the large amounts of data and computational power available today. However, the performance of the resulting networks is hampered by vulnerability to input perturbations and easily spoofed features. We demonstrate in this work how insights from signal modeling can inform the design of robust neural networks.

We begin by studying small *adversarial* perturbations that can induce large classification errors in state-of-the-art deep networks. Here, we show that a systematic exploitation of *sparsity* in natural data is a promising tool for defense. For linear classifiers, we show that a sparsifying front end is provably effective against ℓ_∞-bounded attacks, attenuating output distortion due to the attack by a factor of roughly K/N where N is the data dimension and K is the sparsity level. We then extend this concept to deep networks, showing that a "locally linear" model can be used to develop a theoretical foundation for crafting attacks and defenses. Experiments on the MNIST and CIFAR-10 datasets show the efficacy of the proposed sparsifying front end. Along related lines, we also investigate *compressive* front ends that can be implemented via binary computations in low-power hardware. Key design questions here include the impact of hardware impairments and constraints on the fidelity of information acquisition. We show that a compressive approach is robust to stochastic nonlinearities, and that spatially localized computations are effective, by evaluating classification and reconstruction performance based on the information acquired.

The second case study pertains to robustness in a radio frequency (RF) setting. We focus here on a potentially powerful tool for wireless security: RF device signatures capable of distinguishing between devices sending *exactly* the same message. Such signatures should be robust to standard spoofing techniques, and to different levels of noise in data. Since the information in wireless signals resides in complex baseband, we employ complex-valued neural networks to learn these fingerprints. We demonstrate that, while there are potential benefits to using sections of the signal beyond just the preamble to learn signatures, the network cheats when it can, using information such as the device ID, which can be easily spoofed, to artificially inflate performance. We also show that noise augmentation by inserting additive white Gaussian noise can lead to significant performance gains, which indicates that this counter-intuitive strategy helps in learning more robust fingerprints. We provide results for two different wireless protocols, WiFi and ADS-B, demonstrating the effectiveness of the proposed method.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Deep neural networks represent the state of the art in machine learning in a growing number of fields, including vision, speech and natural language processing. By leveraging large amounts of data and computational power, these networks enable one to substitute domain expertise with purely data-driven models. However, recent work raises important questions about the robustness of such architectures: these networks are vulnerable to classification errors due to tiny, almost imperceptible, perturbations [10], and, as we show below, through reliance on easily spoofed features. In this work, we study the thesis that one can learn robust features by using a small amount of domain expertise in conjunction with data-driven learning. Specifically, we provide two case studies that show how insights from a bottom-up signal modeling viewpoint can inform the design of robust neural networks.

In the first case study, we consider robustness in a rather general setting, studying small *adversarial* perturbations that can fool state-of-the-art networks. We show how *sparse* signal models can be used to combat these perturbations at the front end of a network. Since the existence of adversarial examples has been conjectured to be due to the excessive linearity of deep networks, we begin with a study of linear classifiers, and then extend our results to standard neural networks. We also study design challenges in the hardware implementation of

compressive front ends that leverage sparsity of natural data. The second case study focuses on robustness in a radio frequency (RF) setting, with the goal of learning RF device signatures from subtle nonlinear variations in wireless signals. Our interest here is in resilience to spoofing by an adversary, and to noise in data. We demonstrate the susceptibility of such networks to easily spoofed features and show how noise augmentation helps in learning robust signatures.

1.1 Sparsifying Front Ends

Recent work in machine learning security points out the vulnerability of deep neural networks to adversarial perturbations [10, 7, 11]. As shown in Fig. 1.1, these perturbations can be designed to be barely noticeable to the human eye, but can cause large classification errors in state of the art deep networks. In this section, we attempt to provide fundamental insight into both the vulnerability of deep networks to carefully designed perturbations, and a systematic, theoretically justified framework for designing defenses against adversarial perturbations.

Our starting point is the original intuition in Goodfellow et al. [7] that deep networks are vulnerable to small perturbations not because of their complex, nonlinear structure, but because of their being "too linear". Consider the simple example of a binary linear classifier $\boldsymbol{w}$ operating on N-dimensional input $\boldsymbol{x}$, producing the decision statistic $g(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x}$. The effect

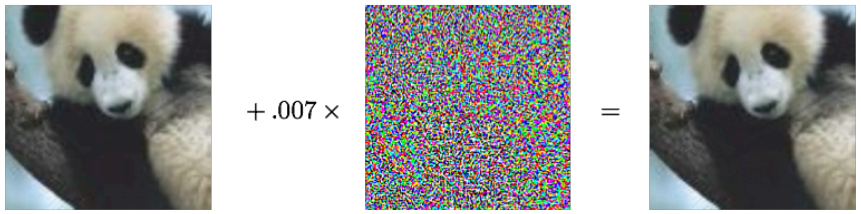


Figure 1.1: An adversarial example on GoogLeNet [6]. Figure taken from [7].

of a perturbation $\boldsymbol{e}$ to the input is given by $g(\boldsymbol{x}+\boldsymbol{e})-g(\boldsymbol{x})=\boldsymbol{w}^T\boldsymbol{e}$. If $\boldsymbol{e}$ is bounded by ℓ_∞ norm (i.e., $\|\boldsymbol{e}\|_\infty = \max_i |e_i| \leq \epsilon$), then the largest perturbation that can be produced at the output is caused by $\boldsymbol{e}=\epsilon\,\mathrm{sgn}(\boldsymbol{w})$, and the resulting output perturbation is $\Delta=\epsilon\sum_{i=1}^{N}|w_i|=\epsilon\|\boldsymbol{w}\|_1$. The latter can be made large unless the ℓ_1 norm of $\boldsymbol{w}$ is constrained in some fashion. To see what happens without such a constraint, suppose that $\boldsymbol{w}$ has independent and identically distributed components, with bounded expected value and variance. It is easy to see that $\|\boldsymbol{w}\|_1=\Theta(N)$[1] with high probability, which means that the effect of ℓ_∞-bounded input perturbations can be blown up at the output as the input dimension increases.

The preceding linear model provides a remarkably good approximation for today's deep CNNs. Convolutions, subsampling, and average pooling are inherently linear. A ReLU unit is piecewise linear, switching between slopes at the bias point. A max-pooling unit is a switch between multiple linear transformations. Thus, the overall transfer function between the input and an output neuron can be written as $\boldsymbol{w}_{\mathrm{eq}}(\boldsymbol{x})^T\boldsymbol{x}$, where $\boldsymbol{w}_{\mathrm{eq}}(\boldsymbol{x})$ is an equivalent "locally linear" transformation that exhibits input dependence through the switches corresponding to the ReLU and max pooling units. For small perturbations, relatively few switches flip, so that $\boldsymbol{w}_{\mathrm{eq}}(\boldsymbol{x}+\boldsymbol{e})\approx\boldsymbol{w}_{\mathrm{eq}}(\boldsymbol{x})$. Note that the preceding argument also applies to more general classes of nonlinearities: the locally linear approximation is even better for sigmoids, since slope changes are gradual rather than drastic. These observations motivate us to begin with a study of linear classifiers, before extending our results to neural networks via a locally linear model.

As we discuss in Section 2.2, the current state-of-the-art defense is based on retraining networks with adversarial examples [12, 13]. However, the lack of insight into what it does fundamentally limits how much we can trust the resulting networks: the sole means of verifying robustness is through empirical evaluations. In contrast, our goal is to provide a systematic approach with theoretical guarantees, with the potential of yielding a longer-term solution to the design of robust neural networks. We take a bottom-up signal processing perspective and exploit

[1] See Section 2.1 for the definitions of the order notation $\Theta(.)$, $\mathcal{O}(.)$, $\omega(.)$, $o(.)$.

the rather general observation that input data must be sparse in *some* basis in order to avoid the curse of dimensionality. Sparsity is an intuitively plausible concept: we understand that humans reject small perturbations by focusing on the key features that stand out. Our proposed approach is based on this intuition. In this work, we show via both theoretical results and experiments that a sparsity-based defense is provably effective against ℓ_∞-bounded adversarial perturbations.

Specifically, we assume that the N-dimensional input data has a K-sparse representation (where $K \ll N$) in a known orthonormal basis. We then employ a sparsifying front end that projects the perturbed data onto the K-dimensional subspace. The intuition behind why this can help is quite clear: small perturbations can add up to a large output distortion when the input dimension is large, and by projecting to a smaller subspace, we can limit the damage. Theoretical studies show that this attenuates the impact of ℓ_∞-bounded attacks by a factor of roughly K/N (the sparsity level), and experiments show that sparsity levels of the order of 1-5% give an excellent tradeoff between the accuracy of input representation and rejection of small adversarial perturbations.

1.1.1 Contributions

We develop a theoretical framework to assess and demonstrate the effectiveness of a sparsity-based defense against adversarial attacks on linear classifiers and neural networks. Our main contributions are as follows:

- For linear classifiers, we quantify the achievable gain of the sparsity-based defense via an ensemble-averaged analysis based on a stochastic model for weights. We prove that with high probability, the adversarial impact is reduced by a factor of roughly K/N, where K is the sparsity of the signal and N is the signal's dimension.

- For neural networks, we use a "locally linear" model to provide a framework for understanding the impact of small perturbations. Specifically, we characterize a high SNR regime

in which the fraction of switches flipping for ReLU nonlinearities for an ℓ_∞-bounded perturbation is small.

- Using the preceding framework, we show that a sparsifying front end is effective for defense, and devise a new attack based on locally linear modeling.
- We supplement our theoretical results with experiments on the MNIST [14] and CIFAR-10 [15] datasets for a variety of recent attacks.

1.2 Compressive Front Ends

Since most natural signals are sparse in some basis, compressive projections [16] are a promising general-purpose approach for low-power front ends for acquisition of information for downstream estimation/learning tasks (Fig. 1.2). They can be realized using inner products with binary coefficients, which is attractive for hardware implementation, and are expected to be resilient to a broad class of impairments. Successful signal reconstruction has been demonstrated under theoretical models of impairments such as outliers [17] [18] and severe quantization [19] [20], and successful image classification was demonstrated in recent experiments on a Large Area

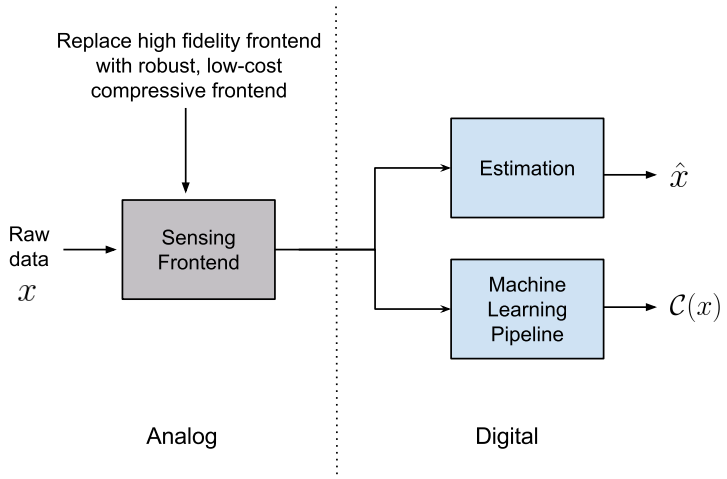


Figure 1.2: Using the compressive framework as a front end for learning applications.

Electronics (LAE)-based image acquisition platform [8], despite significant nonlinearities.

In this work, we carry out a case study of the LAE-based system in [8] to obtain insight into tradeoffs in designing compressive hardware. We model and evaluate the effect of the stochastic nonlinearities in this system, and use the model to explore alternative design choices [21]. While we consider a large-area system, we believe that a similar approach would be highly effective in the design of nanoscale hardware: as semiconductor processes are scaled down, significant stochastic impairments begin to appear in the computational fabric [22].

1.2.1 Contributions

Our key results are as follows:

- We develop a synthetic model for the effect of stochastic nonlinearities in the LAE-based system, which allows us to investigate the impact of potential modifications in hardware design via simulations. We provide insight into the effect of these impairments by further simplifying the synthetic model via a (slightly optimistic) Gaussian approximation.
- The LAE-based system employs row-by-row compressive sensing, with the same matrix employed for all rows. Based on recent theory [21], we expect this to be suboptimal. However, we show that row-by-row compressive projections, which are far easier to implement than projections on the entire image, are competitive in performance, as long as the compressive matrices used are independent across rows.

1.3 RF Signatures

With the proliferation of wireless devices in everyday life, assuring the security of such devices becomes a critical concern. We focus here on a potentially powerful tool for this purpose: wireless fingerprints based on hardware imperfections unique to each device (Fig. 1.3). Prior

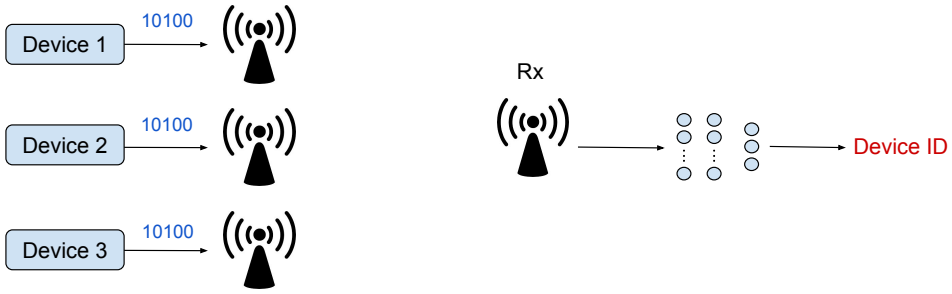


Figure 1.3: Learning RF signatures to distinguish between devices sending the same message.

work shows that it is possible to extract such fingerprints, but it is often based on features extracted with knowledge of the underlying protocol [23, 24, 25, 26, 27, 28, 29, 30, 31, 32]. In this section, we investigate the use of a protocol-agnostic approach, employing supervised learning of fingerprints via a neural network.

Our goal is to extract a fingerprint that enables us to distinguish between two devices sending exactly the same message, using as input the complex baseband signal at the receiver. Since the input is complex-valued and one-dimensional (1D), we employ a 1D convolutional neural network (CNN) with complex-valued weights. When compared to prior approaches [33, 34] that use real-valued networks (with real and imaginary parts of input data treated as independent channels), these networks have a smaller degree of freedom available at the synaptic level. It has been observed that this confers generalization benefits [35], and our results in Section 4.2 corroborate this advantage.

While we would like to develop wireless fingerprinting techniques that are protocol-agnostic, we must remain vigilant against locking onto easily spoofed features. A naive protocol-agnostic scheme would not distinguish between any segments of the message from which the fingerprint is being extracted. However, for any communication protocol, the message contains transmitter ID information, e.g. the MAC address in WiFi packets, the ICAO aircraft address in ADS-B (Automatic Dependent Surveillance-Broadcast) air traffic control signals, etc. Such ID information can be spoofed, hence any fingerprinting technique that uses the entire message must

prove that it does not "cheat" by focusing just on the ID. We demonstrate that a completely protocol-agnostic CNN is vulnerable to such involuntary cheating, and then show that using the preamble, which is common to all packets from all transmitters, suffices to obtain reasonable accuracies, despite the relatively short length of the preamble compared to the length of the entire message. We then explore the impact of noise on training, and propose a noise augmentation strategy for enhancing performance.

1.3.1 Contributions

We propose a protocol-agnostic fingerprinting technique using complex-valued CNNs and demonstrate its robustness to various real-world imperfections. Our main contributions are as follows:

- We demonstrate that supervised learning using complex-valued CNNs works well for two different wireless protocols, WiFi and ADS-B, and compare the performance of different complex activation functions and architectures.
- When making use of portions of the signal beyond just the preamble, we show that networks will "cheat" whenever given the chance, resulting in artificially high accuracies (that are independent of the noise level) by focusing on the transmitter ID information present in these sections.
- We then focus on learning fingerprints from the preamble. Restricting to the preamble is not strictly protocol-agnostic, but, in principle, the location and extent of the preamble can be identified in unsupervised fashion for any given protocol by correlating packets across different transmitters. We study the robustness of our approach to noise in the data, and find that performance is better when the training set has lower SNR than the test set.

- We show that noise augmentation, or insertion of additional white Gaussian noise (AWGN), can significantly improve performance, presumably because it aids in learning more robust fingerprints. In particular, it is important to add noise to test data as well as the training data (with more noise added to training data) to yield benefits.

Chapter 2

Sparsifying Front Ends for Adversarial Learning

In this chapter, we discuss how a systematic exploitation of sparsity in natural data can be used to combat ℓ_∞-bounded adversarial perturbations in deep neural networks. Section 2.2 covers relevant background on adversarial attacks and defenses, along with an overview of work on sparse representations. We then present the details of our sparsity-based defense scheme in Section 2.3. An ensemble-averaged performance analysis of our approach is provided in Section 2.4 for linear classifiers, followed by an extension to neural networks in Section 2.5. Finally, Section 2.6 details experimental results demonstrating the effectiveness of the proposed defense.

2.1 Notation

Here we define the notations $\Theta(.)$, $\mathcal{O}(.)$, $\omega(.)$, and $o(.)$:

- $f = \mathcal{O}(g)$ if and only if there exists a constant $C > 0$ such that $|f/g| < C$,
- $f = \Theta(g)$ if and only if there exist two constants $C_1, C_2 > 0$ such that $C_1 < |f/g| < C_2$,

- $f = \omega(g)$ if and only if there does not exist a constant $C > 0$ such that $|f/g| < C$, and
- $f = \mathcal{O}(g)$ if and only if $g = \omega(f)$.

2.2 Background

It was initially surmised that the vulnerability of deep networks to adversarial perturbations [10] is due to their highly complex, nonlinear nature. However, the success of linearization-based attacks [7, 36] indicates that it is instead due to their "excessive linearity." This is further backed up by work on the curvature profile of deep neural networks [37, 38] showing that their decision boundaries in the vicinity of natural data can be approximated as flat along most directions. Our work on attack and defense is grounded in a locally linear model for neural networks, and the results discussed later demonstrate the efficacy of this approach. We now put our work in context by discussing some of the leading attack and defense strategies, followed by a brief background on sparse signal processing.

2.2.1 Adversarial attacks

Attacks are commonly in the form of additive perturbations to the data, limited in size via an ℓ_p norm constraint. The first known adversarial attack was developed by Szegedy et al. [10]. Given an image $\boldsymbol{x} \in [0,1]^N$, a classifier $f : [0,1]^N \to \{1, \ldots, L\}$ which assigns label $l = f(\boldsymbol{x})$, and an adversarial target $t \neq l$, this attack finds an ℓ_2-small perturbation $\boldsymbol{e}$ via

$$\begin{aligned} \min_{\boldsymbol{e}} \quad & c\,|\boldsymbol{e}| + L(\boldsymbol{x}+\boldsymbol{e},\, t) \\ \text{s.t.} \quad & \boldsymbol{x}+\boldsymbol{e} \in [0,1]^N. \end{aligned} \tag{2.1}$$

where L is the loss function associated with the classifier. The above optimization was solved via box-constrained L-BFGS (limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm [39]),

with a line search to find the smallest c for which $f(\boldsymbol{x}+\boldsymbol{e}) = t$. This approach was effective, but slow.

The next attack proposed by Goodfellow et al. [7] is the well-known *Fast Gradient Sign Method* (FGSM), which devises an ℓ_∞-constrained perturbation. This is a single-step attack that exploits local linearity. Given an ℓ_∞ attack budget $\|\boldsymbol{e}\|_\infty < \epsilon$, FGSM employs a first-order Taylor expansion of the loss function $L(\cdot)$:

$$\boldsymbol{e} = \epsilon \operatorname{sgn}(\nabla_{\boldsymbol{x}} L(\boldsymbol{x}, l)), \tag{2.2}$$

where l is the true label. This is a highly efficient and popular attack. It has been observed that adversarial examples generated using FGSM can be transferred across different architectures [40], enabling the use of substitute networks to attack real-world black box models [41]. Empirical studies indicate that these examples lie in a high-dimensional subspace that overlaps across different models, leading to transferability [42].

Kurakin *et al.* [43, 44] proposed an iterative modification to FGSM:

$$\boldsymbol{e}_{k+1} = \operatorname{clip}_\epsilon(\boldsymbol{e}_k + \delta \operatorname{sgn}(\nabla_{\boldsymbol{x}} L(\boldsymbol{x}+\boldsymbol{e}_k, l))) \tag{2.3}$$

where $\delta \ll \epsilon$ and $\operatorname{clip}_\epsilon(\cdot)$ enforces the ℓ_∞ attack budget at each iteration. While FGSM and iterative FGSM are both non-targeted attacks in their original forms, they can be modified to target a label t by simply replacing $L(\boldsymbol{x}, l)$ with $-L(\boldsymbol{x}, t)$ [43, 44]. A variant of iterative FGSM which starts from a random perturbation around $\boldsymbol{x}$ is known as Projected Gradient Descent (PGD) [12].

Our approach to ℓ_∞-bounded attacks is similar in approach to FGSM and iterative FGSM. Indeed, it is identical to these for binary classification, but yields better performance because it takes a locally optimal approach to creating classification errors.

Another iterative attack that exploits the linearity hypothesis is *DeepFool*, developed by Moosavi-Dezfooli *et al.* [36]. This attack seeks minimal perturbations that cause misclassification for the version of the classification function $g(\cdot)$ linearized around $\boldsymbol{x}+\boldsymbol{e}_k$ at iteration k. For simplicity we present the algorithm for binary classifiers. Each iteration solves

$$\begin{aligned} \min_{\boldsymbol{e}_{k+1}} \quad & \|\boldsymbol{e}_{k+1}\|_2 \\ \text{s.t.} \quad & g(\boldsymbol{x}+\boldsymbol{e}_k) + \nabla_{\boldsymbol{x}} g(\boldsymbol{x}+\boldsymbol{e}_k)^T \boldsymbol{e}_{k+1} = 0. \end{aligned} \tag{2.4}$$

The solution to this optimization can be computed in closed form. We refer to [36] for details on the multi-class version. An extension of DeepFool [45] that iterates over different data points can be used to design data-agnostic perturbations, resulting in attacks that can be transferred across images. The explanation proposed in [45] is based on empirical studies of the geometry of decision boundaries, which indicate that they are highly correlated in the vicinity of natural images.

Finally, one of the strongest currently known attacks is the ℓ_2-bounded attack proposed by Carlini and Wagner [46]. Given an image $\boldsymbol{x} \in [0,1]^N$ and the pre-softmax outputs of a network $\{Z_1(\boldsymbol{x}), Z_2(\boldsymbol{x}), \dots Z_L(\boldsymbol{x})\}$, the Carlini-Wagner attack solves

$$\begin{aligned} \min_{\boldsymbol{e}} \quad & \|\boldsymbol{e}\|_2^2 + c \, \max\left(\max_{i \neq t} Z_i(\boldsymbol{x}+\boldsymbol{e}) - Z_t(\boldsymbol{x}+\boldsymbol{e}), -\kappa \right) \\ \text{s.t.} \quad & \boldsymbol{x}+\boldsymbol{e} \in [0,1]^N, \end{aligned} \tag{2.5}$$

This is an improved formulation of an optimization originally proposed by Szegedy *et al.* [10]. The box constraint in equation (2.5) can be removed by using the substitution $\boldsymbol{x}+\boldsymbol{e} = (\tanh(\boldsymbol{w})+1)/2$; the optimization is then readily solved via gradient-based techniques and a binary search for c. This attack is computationally expensive, but it has been known to break several defenses that claimed robustness against other attacks [46, 47, 48]. [46, 47, 48].

In the NIPS 2017 competition on black box adversarial attacks and defenses [49], the top

three attacks all used extensions of iterative FGSM: by adding a momentum term [50], and generating it on an ensemble of classifiers with random perturbations and augmentation.

Thus, state of the art attacks are all based on some form of iterative optimization, and hence require some gradient computation. While some defenses implicitly or explicitly make gradients difficult to compute, it was shown by Athalye, Carlini and Wagner [51] that it is possible to effectively approximate the gradient even when it is being "obfuscated". Indeed, this paper provides an excellent snapshot of the most powerful adversarial attacks to date, using variants of the PGD and Carlini-Wagner attacks to break a number of defenses.

The results in [51] indicate that attempting to put hurdles in the attacker's path to optimization have a limited chance of success. Our approach to defense, instead, is to try to construct machines that naturally reject perturbations that are "small enough", no matter how cleverly they are crafted. In essence, now that machines are competitive with humans in image and speech recognition in many settings, we now wish to also attain the robustness naturally exhibited by the human visual and auditory systems, which have no problem rejecting small perturbations.

We also note that there has been work on bounding the minimal perturbation necessary for misclassification, averaged over the distribution of input data. Upper bounds on this quantity have been developed as a function of class distinguishability and misclassification rate [52], and loose lower bounds have been developed based on model parameters [53]. Unfortunately, these provide little insight on either attack or defense strategies.

2.2.2 Defense techniques

Existing defense mechanisms against adversarial attacks can be broadly divided into two categories: (a) empirical defenses, based purely on intuitively plausible strategies, together with experimental evaluations, and (b) provable defenses with theoretical guarantees of robustness.

We briefly discuss the empirical work related to our defense, and then give an overview of provable defense techniques.

Empirical Defenses

Given that the success of deep networks can be largely attributed to increased availability of data and computational power, a natural defense strategy is to simply train the network with adversarial examples. Such *adversarial training* was first proposed in [7], which used adversarial examples generated using the FGSM attack. However, retraining with just single-step perturbations is ineffective against iterative attacks [43]. Improved versions include retraining with single-step attacks transferred from other models [54] and with iterative attacks with random starting points [12]. These have proven to be more effective: for example, the defense in [12] is one of the few that showed resistance to the state of the art attacks in [51]. One disadvantage of adversarially trained networks is the higher computational cost due to the large number of examples that must be generated and used for training. Observing that adversarial training can lead to gradient obfuscation if the number of attack steps is small, Qin et al. [13] try to increase the local linearity of the loss surface via a regularizer. This is cost-effective (the number of attack steps can be made smaller) and surprisingly, performs better than standard adversarial training for ImageNet. We note that this is different from the local linearity of the *logits* that we consider. In our view, a fundamental drawback of such empirical approaches is that we have no insight or explicit control on the structure of the network, or of the neuronal outputs, which leaves empirical experiments as the only means of checking their robustness. Thus, while adversarial training is an invaluable tool, it does not provide the assurances we are ultimately seeking.

Preprocessing the input to the network is another important class of defenses. Indeed, the top two defenses in the NIPS 2017 competition on adversarial attacks and defenses [49] were based on such preprocessing, via a neural network based denoiser [55], and via random resizing

and padding [56]. A variety of other defenses utilize preprocessing techniques that are implicitly sparsity-based, including JPEG compression [57, 58], PCA [59], and projection onto GAN-based generative models [60, 61]. However many such defenses were found to confer robustness purely by obfuscating gradients necessary for the adversary, and were successfully defeated by the gradient approximation techniques developed by Athalye, Carlini and Wagner in [51]. In contrast, our sparsity-based defense is robust to the attack techniques in [51]. Overall, the evaluations in such prior work have been purely empirical in nature. Our proposed framework, grounded in a locally linear model for neural networks, provides a foundation for systematic pursuit of sparsity-based preprocessing.

Provable Defenses

There is a growing body of work focused on developing *provable* guarantees of robustness against adversarial perturbations [62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76]. While we do not provide an exhaustive discussion, we illustrate some typical characteristics of these approaches.

Many provable defenses focus on retraining the network with an optimization criterion that promotes robustness towards all possible small perturbations around the training data. For example, the "certified defense" of Raghunathan et al. [62] considers neural networks with one hidden layer and tries to upper bound the ℓ_1 norm of the gradient of the classifier around the data point by the optimal value of a semidefinite program (SDP). It then optimizes this SDP relaxation during training to obtain a more robust network. The certificates provided are quite loose, and are outperformed by other defenses, including ours. A tighter SDP relaxation is developed in [63] to certify the robustness of any given network, but it is not used to train a new robust model. Both the SDP certificates are computationally restricted to fully-connected networks. Another provable defense by Kolter and Wong [64] employs a linear programming (LP) based approach to bound the robust error. This is more efficient than the SDP approach,

and can be scaled to larger networks [65]. The LP technique works well for small perturbations, but for larger perturbation sizes there is a penalty on the accuracy without attacks.

A different training technique based on distributionally robust optimization was proposed by Sinha et al [66], providing a certificate of robustness for attacks whose probability distribution is bounded in Wasserstein distance from the original data distribution. Although the theoretical guarantees do not directly translate to norm-bounded attacks, this approach yields good results in practice for small perturbation sizes. Another defense that works well is that of Mirman et al [67], who use the framework of abstract interpretation to develop various upper bounds on the adversarial loss which can be optimized over during training. Some other provable defenses try to limit Lipschitz constants related to the network output function, for example via cross-Lipschitz regularization [68] and ℓ_2 matrix-norm regularization of weights [69] in order to defend against ℓ_2-bounded attacks. However the bounds in [68] are limited to 2-layer networks, and [69] is based on layerwise bounds that have been shown to be loose [62, 64].

All of the above techniques try to train the network so as to make its output less sensitive to input perturbations by modifying the optimization framework employed for network training. In contrast, our defense takes a bottom-up signal processing approach, exploiting the sparsity of natural data to combat perturbations at the front end, while using conventional network training. It is therefore potentially complementary to defenses based on modifying network training. Another related work that uses sparsity-based preprocessing is [71], which studies ℓ_0-bounded attacks. These are not visually imperceptible (unlike the ℓ_∞-bounded attacks that we study), but they are easy to realize in practice, for example by placing a small sticker on an image. For this class of attacks, [71] shows that the sparse projection can be reformulated as a compressive sensing (CS) problem, and obtain a provably good estimate of the original image via CS recovery algorithms.

For ℓ_2-bounded attacks, Fawzi et al. [72] derive an upper bound on the smallest attack budget that misclassifies every image, under the assumption of a generative model that maps

Gaussian random vectors to images. This upper bound can be achieved by a classifier that is linear in the latent space. While interesting, it has not yet translated to a defense strategy. It differs from our work on ℓ_∞-bounded attacks, where we find an upper bound on the output distortion for a *fixed* attack budget. There is also a line of work on *verifying* the robustness of a given network by using exact solvers based on discrete optimization techniques such as satisfiability modulo theory [73, 74] and mixed-integer programming [75, 76]. However these techniques have combinatorial complexity (in the worst-case, exponential in network size), and so far have not been scaled beyond moderate network sizes.

2.2.3 Sparse representations

It is well-known that most natural data can be compactly expressed as a sparse linear combination of atoms in *some* basis [77, 78, 79]. Such sparse representations have led to state-of-the-art results in many fundamental signal processing tasks such as image denoising [80, 81], neuromagnetic imaging [82, 83], image super-resolution [84], inpainting [85], blind audio source separation [86], source localization [87], etc. There are two broad classes of sparsifying dictionaries employed in literature: predetermined bases such as wavelets [77, 88], and learnt bases which are inferred from a set of training signals [81, 89, 90]. Bases learnt via the latter approach are usually overcomplete in nature. Overcompleteness of representations is a fundamental property exhibited by the biological receptive fields of the mammalian primary visual cortex [91, 92], as well as artifical neural networks trained with backpropagation [93]. Such bases have been found to outperform predetermined dictionaries. Sparsity has also been suggested purely as a means of improving classification performance [94], which indicates that the performance penalty for appropriately designed sparsity-based defenses could be minimal.

2.3 Sparsity Based Defense

2.3.1 Problem Setup

For simplicity we start with binary classification. Given a binary inference model $g : \mathbb{R}^N \to \mathbb{R}$, we assume that its input data $\boldsymbol{x} \in \mathbb{R}^N$ has a K-sparse representation ($K \ll N$) in a known orthonormal basis Ψ: $\left\|\Psi^T \boldsymbol{x}\right\|_0 \leq K$. Let us denote by $\hat{\boldsymbol{x}}$ a *modified* version of the data sample $\boldsymbol{x}$. We now define a performance measure Δ that quantifies the robustness of $g(\cdot)$:

$$\Delta(\boldsymbol{x}, \hat{\boldsymbol{x}}) = |g(\hat{\boldsymbol{x}}) - g(\boldsymbol{x})|.$$

For example, for a linear classifier $g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$, the performance measure is $\Delta(\boldsymbol{x}, \hat{\boldsymbol{x}}) = |\boldsymbol{w}^T(\hat{\boldsymbol{x}} - \boldsymbol{x})|$.

We now consider a system comprised of the classifier $g(\cdot)$ and two external participants: the adversary and the defense, depicted in Fig. 2.1.

1. The adversary corrupts the input $\boldsymbol{x}$ by adding a perturbation $\boldsymbol{e}$, with the goal of causing misclassification:

   $$\max_{\boldsymbol{e}} \Delta(\boldsymbol{x}, \hat{\boldsymbol{x}}) \quad \text{s.t.} \quad \|\boldsymbol{e}\|_\infty < \epsilon.$$

   We are interested in perturbations that are visually imperceptible, hence we impose an ℓ_∞-constraint on the adversary.

2. The defense preprocesses the perturbed data via a function $f : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}^{\mathbb{N}}$, with the goal of minimizing the adversarial impact Δ.

This setup can be easily extended to multiclass classification as we show in Section 2.5.5.

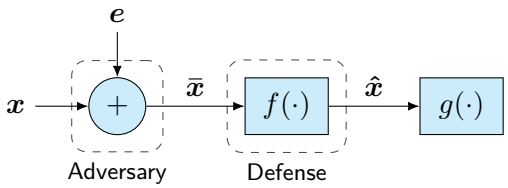


Figure 2.1: Block diagram of the system, depicting an adversarial example $\bar{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{e}$ (with ℓ_∞ constraint on $\boldsymbol{e}$), a preprocessing defense $f(\cdot)$ and a classifier $g(\cdot)$.

2.3.2 Sparsifying Front End Description

We propose a defense based on a *sparsifying front end* that exploits sparsity in natural data to combat adversarial attacks. Specifically, we preprocess the input via a front end that computes a K-sparse projection in the basis Ψ. Figure 2.2 shows a block diagram of our model for a neural network, depicting an additive perturbation followed by sparsity-based preprocessing.

Here is an intuitive explanation of how this defense limits adversarial perturbations. If the data is exactly K-sparse in domain Ψ, the front end leaves the input unchanged ($\hat{\boldsymbol{x}} = \boldsymbol{x}$) when there is no attack ($\boldsymbol{e} = 0$). The front end attenuates the perturbation by projecting it onto the space spanned by the basis functions corresponding to the K retained coefficients. If the perturbation is small enough, then the K retained coefficients corresponding to $\boldsymbol{x}$ and $\boldsymbol{x} + \boldsymbol{e}$ remain the same, in which case the neural network sees the original input plus the projected, and hence attenuated, perturbation.

Let $\mathcal{H}_{\mathrm{K}} : \mathbb{R}^N \to \mathbb{R}^N$ represent the block that enforces sparsity by retaining the K coefficients

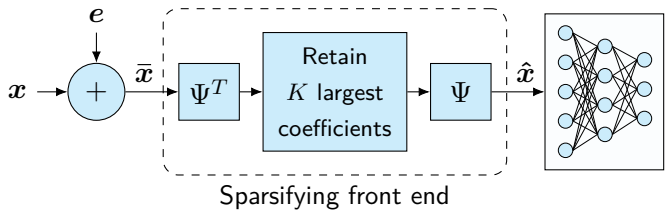


Figure 2.2: Sparsifying front end defense: For a basis in which the input is sparse, the input is projected onto the subspace spanned by the K largest basis coefficients.

largest in magnitude and zeroing out the rest. We can now define the following:

- The support $\mathcal{S}_{\mathrm{K}}$ of the K-sparse representation of $\boldsymbol{x}$:

$$\mathcal{S}_{\mathrm{K}}(\boldsymbol{x}) \triangleq \mathrm{supp}\big(\mathcal{H}_{\mathrm{K}}\big(\Psi^T\boldsymbol{x}\big)\big)$$

- The projection $\mathcal{P}_{\mathrm{K}}$ of $\boldsymbol{e}$ onto the subspace that $\boldsymbol{x}$ lies in:

$$\mathcal{P}_{\mathrm{K}}(\boldsymbol{e},\boldsymbol{x}) \triangleq \sum_{k\in\mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \boldsymbol{\psi}_k\boldsymbol{\psi}_k^T\boldsymbol{e}.$$

- The *high SNR regime* is the operating region where the perturbation does not change the subspace that $\boldsymbol{x}$ lies in:

$$\text{High SNR:} \quad \mathcal{S}_{\mathrm{K}}(\boldsymbol{x}) = \mathcal{S}_{\mathrm{K}}(\boldsymbol{x}+\boldsymbol{e}). \tag{2.6}$$

  We characterize the conditions guaranteeing (2.6) in Prop. 1.

If we operate at high SNR, we get

$$\mathcal{H}_{\mathrm{K}}\big(\Psi^T(\boldsymbol{x}+\boldsymbol{e})\big) = \mathcal{H}_{\mathrm{K}}\big(\Psi^T\boldsymbol{x}\big) + \bar{\boldsymbol{e}} = \Psi^T\boldsymbol{x} + \bar{\boldsymbol{e}},$$

where

$$\bar{e}_k = \begin{cases} \boldsymbol{\psi}_k^T\boldsymbol{e}, & \text{if } k \in \mathcal{S}_{\mathrm{K}}(\boldsymbol{x}) \\ 0, & \text{otherwise.} \end{cases}$$

Therefore, the front end preserves the signal, i.e. $\mathcal{H}_{\mathrm{K}}\big(\Psi^T(\boldsymbol{x}+\boldsymbol{e})\big) = \Psi^T\boldsymbol{x} + \mathcal{H}_{\mathrm{K}}(\Psi^T\boldsymbol{e})$, and hence

its output $\hat{\boldsymbol{x}}$ can be written as follows:

$$\hat{\boldsymbol{x}} = \boldsymbol{x} + \sum_{k \in \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \boldsymbol{\psi}_k \boldsymbol{\psi}_k^T \boldsymbol{e} = \boldsymbol{x} + \mathcal{P}_{\mathrm{K}}(\boldsymbol{e}, \boldsymbol{x}).$$

Thus the effective perturbation is $\mathcal{P}_{\mathrm{K}}(\boldsymbol{e}, \boldsymbol{x})$, which lives in a lower dimensional space. Its impact is therefore significantly reduced. In Sections 2.4 and 2.5, we quantify the reduction in adversarial impact via an ensemble-averaged analysis based on a stochastic model for the classifier $g(\cdot)$.

2.3.3 Characterizing the High SNR Regime

We can gain valuable design insight by characterizing the conditions that guarantee high SNR operation of the sparsifying front end, as stated in the following proposition:

Proposition 1 *For sparsity level K and perturbation $\boldsymbol{e}$ with $\|\boldsymbol{e}\|_\infty \leq \epsilon$, the sparsifying front end preserves the input coefficients if the following SNR condition holds:*

$$\mathrm{SNR} \triangleq \frac{\lambda}{\epsilon} > \gamma,$$

where λ is the magnitude of the smallest non-zero entry of $\mathcal{H}_{\mathrm{K}}(\Psi^T \boldsymbol{x})$ and $\gamma = 2 \max_k \|\boldsymbol{\psi}_k\|_1$.

Proof: By Holder inequality, the SNR condition implies that

$$\lambda > \epsilon\, \gamma \geq 2 \max_k \left|\boldsymbol{\psi}_k^T \boldsymbol{e}\right| \geq \left|\boldsymbol{\psi}_i^T \boldsymbol{e}\right| + \left|\boldsymbol{\psi}_j^T \boldsymbol{e}\right| \;\; \forall\, i, j.$$

In particular, we can choose i and j such that

$$\min_{i \in \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \left(\left|\boldsymbol{\psi}_i^T \boldsymbol{x}\right| - \left|\boldsymbol{\psi}_i^T \boldsymbol{e}\right|\right) > \max_{j \notin \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \left|\boldsymbol{\psi}_j^T \boldsymbol{e}\right|$$

where we have used the fact that $\lambda = \min_{i \in \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \left| \boldsymbol{\psi}_i^T \boldsymbol{x} \right|$. We can now use the triangle inequality to get

$$\min_{k \in \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \left| \boldsymbol{\psi}_k^T (\boldsymbol{x} + \boldsymbol{e}) \right| > \max_{j \notin \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})} \left| \boldsymbol{\psi}_j^T \boldsymbol{e} \right|.$$

It is easy to see that this is equivalent to $\mathcal{S}_{\mathrm{K}}(\boldsymbol{x} + \boldsymbol{e}) = \mathcal{S}_{\mathrm{K}}(\boldsymbol{x})$, which completes the proof. ■

Remark 1 *The SNR condition is easier to satisfy for bases with sparser, or more localized, basis functions (smaller M). For example, we expect a wavelet basis to be better than a DCT basis. As we will see later in Section 2.4.2, this is also a favorable criterion for performance in the white box attack scenario.*

Remark 2 *Another important design parameter is the value of K, which must be chosen keeping the following in mind: lower sparsity levels allow us to impose high SNR even for larger perturbations, but if the data is only approximately K-sparse, this results in unwanted signal perturbation. These must be traded off to optimize classification performance.*

All of our subsequent analysis in this section is based on the assumption that the SNR condition in Proposition 1 holds. In this case, the sparsifying front end is signal-preserving, hence the output distortion can be quantified solely by analyzing its effect on the adversarial perturbation. In our experiments with MNIST data, we find that the SNR condition is approximately satisfied for the range of K that works most effectively (1-5% of the coefficients in a wavelet basis). We start with a study of linear classifiers and then extend our results to neural networks via a locally linear model.

2.4 Analysis for Linear Classifiers

Consider a linear classifier $g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$. We calculate the adversarial impact for various attack models and quantify the efficacy of our defense by using a stochastic model for $\boldsymbol{w}$.

2.4.1 Impact of Adversarial Perturbation

When the front end is not present, the impact of the attack is $\Delta = \boldsymbol{w}^T\boldsymbol{e}$. By Holder inequality, we have

$$\Delta = \boldsymbol{w}^T\boldsymbol{e} \leq \|\boldsymbol{e}\|_\infty \|\boldsymbol{w}\|_1 \leq \epsilon \, \|\boldsymbol{w}\|_1 \triangleq \Delta_0, \tag{2.7}$$

where the second inequality follows from the ℓ_∞ attack budget constraint. We can observe that $\boldsymbol{e}_0 = \epsilon \, \mathrm{sgn}(\boldsymbol{w})$ achieves equality in (2.7), which means that $\boldsymbol{e}_0$ is the optimal attack when the adversary has knowledge of $\boldsymbol{w}$. We use $\Delta_0 = \epsilon\|\boldsymbol{w}\|_1$ as a baseline to assess the efficacy of our defense.

When the defense is present, the adversarial impact Δ becomes

$$\Delta = \left|\boldsymbol{w}^T(\hat{\boldsymbol{x}} - \boldsymbol{x})\right| = \left|\boldsymbol{w}^T \mathcal{P}_{\mathrm{K}}(\boldsymbol{e}, \boldsymbol{x})\right| = \left|\boldsymbol{e}^T \mathcal{P}_{\mathrm{K}}(\boldsymbol{w}, \boldsymbol{x})\right| \tag{2.8}$$

where the second equality follows from the definition of $\mathcal{P}_{\mathrm{K}}(\boldsymbol{e}, \boldsymbol{x})$. We can now consider two scenarios depending on the adversary's knowledge of the defense and the classifier:

1. *Semi-white box scenario:* Here perturbations are designed based on the knowledge of $\boldsymbol{w}$ alone, and therefore the attack remains

   $$\boldsymbol{e}_{\mathrm{SW}} = \epsilon \, \mathrm{sgn}(\boldsymbol{w}).$$

   Using (2.8), the output distortion becomes

   $$\Delta_{\mathrm{SW}} = \epsilon \left|\mathrm{sgn}(\boldsymbol{w}^T) \, \mathcal{P}_{\mathrm{K}}(\boldsymbol{w}, \boldsymbol{x})\right|. \tag{2.9}$$

   We note that the attack is aligned with $\boldsymbol{w}$.

2. *White box scenario:* Here the adversary has the knowledge of both $\boldsymbol{w}$ and the front

end, and designs perturbations accordingly. accordingly. This results in the following optimization problem:

$$\begin{aligned} \max_{\boldsymbol{e}} \quad & \left|\boldsymbol{e}^T \mathcal{P}_{\mathrm{K}}(\boldsymbol{w}, \boldsymbol{x})\right| \\ \text{s.t.} \quad & \|\boldsymbol{e}\|_\infty < \epsilon. \end{aligned}$$

We can use the same Holder inequality based argument as before to prove that the optimal perturbation is

$$\boldsymbol{e}_{\mathrm{W}} = \epsilon \operatorname{sgn}(\mathcal{P}_{\mathrm{K}}(\boldsymbol{w}, \boldsymbol{x})),$$

The resulting output distortion can be written as

$$\Delta_{\mathrm{W}} = \epsilon \left\|\mathcal{P}_{\mathrm{K}}(\boldsymbol{w}, \boldsymbol{x})\right\|_1.$$

Thus, instead of being aligned with $\boldsymbol{w}$, $\boldsymbol{e}_{\mathrm{W}}$ is aligned to the projection of $\boldsymbol{w}$ onto the subspace that $\boldsymbol{x}$ lies in.

2.4.2 Ensemble Averaged Performance

We now provide an analysis that quantifies the robustness provided by sparsification over an ensemble of linear classifiers, by imposing a stochastic model for $\boldsymbol{w}$. Specifically, we show that the defense attenuates the output distortion by a factor of K/N for the semi-white box attack, and by at least $\mathcal{O}(K \operatorname{polylog}(N)/N)$ for the white box attack, where K is the sparsity of the signal and N is the signal dimension.

Assumption 1 *We assume a random model for* $\boldsymbol{w} = (w_1, \ldots, w_N)^T$*, where the entries* $\{w_i\}_{i=1}^N$ *are i.i.d. with zero mean and median:* $\mathbb{E}[w_1] = 0$ *and* $\mathbb{E}[\operatorname{sgn}(w_1)] = 0$*. Let* $\mathbb{E}[|w_1|] = \mu = \Theta(1)$ *and* $\mathbb{E}\left[w_1^2\right] = \sigma^2 = \Theta(1)$*.*

We observe that the baseline adversarial impact $\Delta_0 = \epsilon \|\boldsymbol{w}\|_1$ scales with N. This is formalized in the following proposition:

Proposition 2 *Δ_0/N converges to $\epsilon\,\mu$ almost surely, i.e*

$$\Pr\left(\lim_{N\to\infty} \frac{\Delta_0}{N} = \epsilon\,\mu\right) = 1.$$

Thus, with no defense, the adversarial impact scales as $\Theta(N)$.

Proof: Δ_0 is the sum of i.i.d. random variables $\epsilon\,|w_i|$ with finite mean: $\mathbb{E}[\epsilon\,|w_i|] = \epsilon\,\mu < \infty$. Hence we can apply the strong law of large numbers, which completes the proof. ■

We now state the following theorems that characterize the performance of the sparsifying front end defense in the semi-white box and white box scenarios.

1. *Semi-White Box Scenario:*

Theorem 1 *As K and N approach infinity, Δ_{SW}/K converges to $\epsilon\,\mu$ in probability, i.e.*

$$\lim_{K\to\infty} \Pr\left(\left|\frac{\Delta_{\mathrm{SW}}}{K} - \epsilon\,\mu\right| \le \delta\right) = 1 \quad \forall\ \delta > 0.$$

Remark 3 *After sparsification, the impact Δ_{SW} of the adversarial perturbation scales linearly with the sparsity level K. Thus, the sparsifying front end provides an attenuation of K/N on the effect of the semi-white box adversarial attack.*

Proof: Let us assume without loss of generality that $\mathcal{S}_{\mathrm{K}}(\boldsymbol{x}) = \{1, \ldots, K\}$. We can rewrite the adversarial impact (2.9) as

$$\Delta_{\mathrm{SW}} = \epsilon\,|Z_K|, \text{ where } Z_K = \sum_{i=1}^{K} \mathrm{sgn}(\boldsymbol{w})^T \boldsymbol{\psi}_k \boldsymbol{\psi}_k^T \boldsymbol{w}.$$

The following lemma provides an upper bound on the mean and variance of Z_K.

Lemma 1 *The mean and variance of Z_K are bounded by linear functions of K.*

$$\mathbb{E}[Z_K] = K\mu, \quad \text{var}(Z_K) \leq K\big(\sigma^2 + \mu^2\big).$$

Proof: We can write $Z_K = \sum_{i=1}^{K} U_i V_i$, where

$$U_i = \sum_{m=1}^{N} \psi_i[m]\, w_m, \; V_i = \sum_{m=1}^{N} \psi_i[m]\, \text{sgn}(w_m).$$

We observe that for $i, j \in \{1, \ldots, K\}$, $\mathbb{E}[U_i V_i] = \mu$, and

$$\begin{aligned}
\text{var}(U_i V_i) &= \sigma^2 + \mu^2 - 2\mu^2 \sum_{m=1}^{N} \psi_i^4[m], \\
\text{cov}(U_i V_i, U_j V_j) &= -2\mu^2 \sum_{m=1}^{N} \psi_i^2[m]\, \psi_j^2[m], \quad i \neq j.
\end{aligned}$$

Hence we get $\mathbb{E}[Z_K] = K\mu$, and

$$\begin{aligned}
\text{var}(Z_K) &= \sum_{i=1}^{K} \text{var}(U_i V_i) - \sum_{\substack{i,j=1 \\ i \neq j}}^{K} \text{cov}(U_i V_i, U_j V_j) \\
&= K\big(\sigma^2 + \mu^2\big) - 2\mu^2 \sum_{m=1}^{N} \sum_{i,j=1}^{K} \psi_i^2[m]\, \psi_j^2[m] \\
&\leq K\big(\sigma^2 + \mu^2\big).
\end{aligned}$$

■

We can now apply Chebyshev's inequality to $Y_K = Z_K/K$, noting that $\mathbb{E}[Y_K] = \mu$ and $\delta(Y_K) \leq \big(\sigma^2 + \mu^2\big)/K$. Using the bounds in the lemma, we obtain

$$\Pr(|Y_K - \mu| \leq \delta) \geq 1 - \frac{1}{K}\left(\frac{\sigma^2 + \mu^2}{\delta^2}\right) \quad \forall\, \delta \geq 0. \tag{2.10}$$

Note that $|\Delta_{\mathrm{SW}}/K - \epsilon\,\mu| = \epsilon\,||Y_K| - \mu| \leq \epsilon\,|Y_K - \mu|$. The statement of the theorem follows by (2.10) and letting $K \to \infty$ in the above inequality. ■

2. *White Box Scenario:*

We begin with the following lemma that provides a useful upper bound on the impact of the white box attack.

Lemma 2 *An upper bound on the white box distortion Δ_{W} is given by*

$$\Delta_{\mathrm{W}} \leq \epsilon \sum_{k=1}^{K} \big|\boldsymbol{\psi}_k^T \boldsymbol{w}\big| \|\boldsymbol{\psi}_k\|_1.$$

Proof:

$$\begin{aligned}\Delta_{\mathrm{W}} = \epsilon \|\mathcal{P}_{\mathrm{K}}(\boldsymbol{w}, \boldsymbol{x})\|_1 &= \epsilon \sum_{i=1}^{N} \left| \sum_{k=1}^{K} \big(\boldsymbol{\psi}_k^T \boldsymbol{w}\big) \psi_k[i] \right| \\ &\leq \epsilon \sum_{i=1}^{N} \sum_{k=1}^{K} \big|\boldsymbol{\psi}_k^T \boldsymbol{w}\big|\, |\,\psi_k[i]| = \epsilon \sum_{k=1}^{K} \big|\boldsymbol{\psi}_k^T \boldsymbol{w}\big| \|\boldsymbol{\psi}_k\|_1.\end{aligned}$$

■

Remark 4 *The upper bound is exact if the supports of the K selected basis functions do not overlap. In our MNIST experiments, this is approximately satisfied for the range of K that works most effectively (1-5% of the coefficients in a wavelet basis).*

Remark 5 *Since the upper bound has K terms, the distortion cannot grow slower than K. As stated in the following theorem, however, if the basis functions are "localized" with ℓ_1 norms that do not scale too fast with N, then the output distortion scales as $\mathcal{O}(K\,\mathrm{polylog}(N))$.*

Theorem 2 *Under the assumptions $\|\boldsymbol{\psi}_k\|_1 = \mathcal{O}(\log N)$, $\|\boldsymbol{\psi}_k\|_\infty = o(1)\ \forall\, k \in \{1, 2, \ldots, K\}$,*

and $\|\boldsymbol{w}\|_\infty = \mathcal{O}(1)$*, we have the following upper bound for* Δ_{W}*:*

$$\lim_{N\to\infty} \Pr(\Delta_{\mathrm{W}} \leq \mathcal{O}(K \operatorname{polylog}(N))) = 1.$$

Thus, the impact of adversarial perturbation in the case of white box attack is attenuated by a factor of $\mathcal{O}(K \operatorname{polylog}(N)/N)$ *compared to having no defense.*

Proof: We first state the following convergence result:

Lemma 3 $\boldsymbol{\psi}_k^T \boldsymbol{w} \to \mathcal{N}(0, \sigma^2)$ *in distribution.*

Proof: We show that we can apply Lindeberg's version of the central limit theorem, noting that $\boldsymbol{\psi}_k^T \boldsymbol{w} = \sum_{i=1}^N Y_i$, where $Y_i = \psi_k[i]\, w_i$ are independent random variables with $\mathbb{E}[Y_i] = 0$ and $\operatorname{var}(Y_i) = \sigma_i^2$, with $\sum_{i=1}^N \sigma_i^2 = \sigma^2$.

Now, given $\delta > 0$, we investigate the following quantity in order to check Lindeberg's condition:

$$L(\delta, N) = \frac{1}{\sigma^2} \sum_{i=1}^{N} \mathbb{E}\big[Y_i^2 \mathbb{1}_{\{|Y_i| > \delta\sigma\}}\big].$$

From the ℓ_∞ assumptions on $\boldsymbol{\psi}_k$ and $\boldsymbol{w}$, we observe that

$$\begin{aligned}\mathbb{E}\big[\psi_k^2[i] w_i^2 \mathbb{1}_{\{|Y_i| > \delta\sigma\}}\big] &\leq \mathcal{o}^2(1)\, \mathcal{O}^2(1) \Pr\{(|Y_i| > \delta\sigma)\} \\ &= \mathcal{o}^2(1)\, \mathcal{O}^2(1) \Pr\left\{ \left(|w_i| > \frac{\delta\sigma}{\mathcal{o}(1)} \right) \right\}.\end{aligned}$$

Also note that $\forall\, \delta > 0,\ \exists\, M$ s.t. $\forall N > M,\ |w_i| < \delta\sigma / \mathcal{o}(1)\ \forall\, i \in \{1, \ldots, N\}$. Hence we get $\lim_{N\to\infty} L(\delta, N) = 0$, which is Lindeberg's condition. ■

We can use Lemmas 2 and 3 to obtain

$$\begin{aligned}\Pr(\Delta_{\mathrm{W}} > \delta) &\leq \Pr\left(\epsilon \sum_{k=1}^{K} \left|\boldsymbol{\psi}_k^T \boldsymbol{w}\right| \|\boldsymbol{\psi}_k\|_1 > \delta\right) \\ &\leq \Pr\left(\bigcup_{k=1}^{K} \left\{\left|\boldsymbol{\psi}_k^T \boldsymbol{w}\right| > \frac{\delta}{\epsilon\, K \|\boldsymbol{\psi}_k\|_1}\right\}\right) \\ &\leq \sum_{k=1}^{K} \Pr\left(\left|\boldsymbol{\psi}_k^T \boldsymbol{w}\right| > \frac{\delta}{\epsilon\, K \|\boldsymbol{\psi}_k\|_1}\right) \\ &= \sum_{k=1}^{K} 2Q\left(\frac{\delta}{\epsilon\, \sigma K \|\boldsymbol{\psi}_k\|_1}\right) = 2KQ\left(\frac{\delta}{\epsilon\, \sigma}\, \mathcal{O}\left(\frac{1}{K \log N}\right)\right),\end{aligned}$$

where the last step follows from the ℓ_1 assumption on $\boldsymbol{\psi}_k$, and $Q(x)$ is the Gaussian tail distribution function $\int_x^\infty e^{-t^2/2} dt\, /\sqrt{2\pi}$. We complete the proof by setting $\delta = \mathcal{O}(K \,\mathrm{polylog}(N))$ which makes the right-hand side of the above equation vanish as N approaches infinity. ■

A practical take-away from the above theoretical results is that, in order for the defense to be effective against a white box attack, not only do we need input sparsity ($K \ll N$), but we also need that the individual basis functions be localized (small in ℓ_1 norm). The latter implies, for example, that sparsification with respect to a wavelet basis, which has more localized basis functions, should be more effective than with a DCT basis.

2.4.3 The Low SNR Scenario

So far, we have assumed operation in the high SNR regime, i.e, (2.6) holds. However, for larger values of ϵ, or in scenarios where $\boldsymbol{x}$ is only approximately K-sparse, $\mathcal{S}_{\mathrm{K}}(\boldsymbol{x}+\boldsymbol{e})$ and $\mathcal{S}_{\mathrm{K}}(\boldsymbol{x})$ could be different (and possibly overlapping) sets of K basis vectors. This would affect the optimal adversarial perturbation. Here we introduce an iterative algorithm which obtains an enhanced version of the white box attack in the low SNR regime. At iteration i, we calculate

$$\boldsymbol{e}_{\mathrm{IW}}^{[i+1]} = \boldsymbol{e}_{\mathrm{IW}}^{[i]} + \delta\, \mathrm{sgn}\Big(\mathcal{P}_{\mathrm{K}}\Big(\boldsymbol{w}, \boldsymbol{x} + \boldsymbol{e}_{\mathrm{IW}}^{[i]}\Big)\Big),$$

where $\delta < \epsilon$ Essentially, we refine our estimate of the support $\mathcal{S}_{\mathrm{K}}(\boldsymbol{x}+\boldsymbol{e})$ at each iteration by calculating the projection of $\boldsymbol{w}$ on to the top-K basis vectors of $\boldsymbol{x}+\boldsymbol{e}$ (rather than just $\boldsymbol{x}$).

However, as we discuss in our experiments (Section 2.6.1), this scenario is not of practical interest because low SNR occurs only for large values of ϵ or ρ.

2.5 Analysis for Neural Networks

In this section, we build on the preceeding insights for neural networks by exploiting locally linear approximations. For simplicity we start with a 2-layer, fully connected network trained for binary classification, and then extend our results to a general network for multi-class classification.

2.5.1 Locally Linear Representation

Consider first binary classification using a neural network with one hidden layer with M neurons, as depicted in Fig. 2.3. Since ReLU units are piecewise linear, switching between slopes of 0 and 1, they can be represented using input-dependent switches. Given input $\boldsymbol{x}$, we denote by $s_i(\boldsymbol{x}) \in \{0,1\}$ the switch corresponding to the ith ReLU unit. Now the activations of the hidden layer neurons can be written as follows:

$$a_i = \mathrm{ReLU}\big(\boldsymbol{w}_i^T\boldsymbol{x} - b_i\big) = s_i(\boldsymbol{x})\,\boldsymbol{w}_i^T\boldsymbol{x} - s_i(\boldsymbol{x})\,b_i.$$

The output of the neural network can be written as

$$\begin{aligned} y(\boldsymbol{x}) = \boldsymbol{w}_0^T\boldsymbol{a} &= \sum_{i=1}^{M} s_i(\boldsymbol{x})\,w_0[i]\,\boldsymbol{w}_i^T\boldsymbol{x} - \sum_{i=1}^{M} s_i(\boldsymbol{x})\,w_0[i]\,b_i \\ &= \boldsymbol{w}_{\mathrm{eq}}(\boldsymbol{x})^T\boldsymbol{x} - b_{\mathrm{eq}}(\boldsymbol{x}). \end{aligned}$$

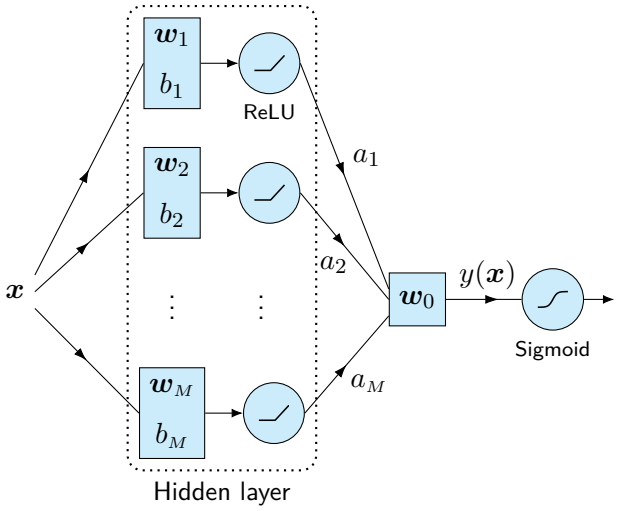


Figure 2.3: Two layer neural network for binary classification. ReLU units are piecewise linear, hence the network is locally linear: $y(\boldsymbol{x}) = \boldsymbol{w}_{\text{eq}}(\boldsymbol{x})^T \boldsymbol{x} - b_{\text{eq}}(\boldsymbol{x})$.

where

$$\boldsymbol{w}_{\text{eq}}(\boldsymbol{x}) = \sum_{i=1}^{M} s_i(\boldsymbol{x})\, w_0[i]\, \boldsymbol{w}_i, \quad b_{\text{eq}}(\boldsymbol{x}) = \sum_{i=1}^{M} s_i(\boldsymbol{x})\, w_0[i]\, b_i.$$

This locally linear model extends to any standard neural network, since convolutions and subsampling are inherently linear and max-pooling units can also be modeled as switches. For more than 2 classes, we will apply this modeling approach to the "transfer function" from the input to the inputs to the softmax layer, as discussed in Section 2.5.5.

2.5.2 Impact of Adversarial Perturbation

Now we consider the effect of an ℓ_∞-bounded perturbation $\boldsymbol{e}$ on the performance of the network. For ease of notation, we write $\boldsymbol{w}_{\text{eq}} = \boldsymbol{w}_{\text{eq}}(\boldsymbol{x})$, $\overline{\boldsymbol{w}}_{\text{eq}} = \boldsymbol{w}_{\text{eq}}(\boldsymbol{x}+\boldsymbol{e})$, and $\bar{b}_{\text{eq}} = b_{\text{eq}}(\boldsymbol{x}+\boldsymbol{e})$.

The distortion due to the attack can be written as

$$\begin{aligned}\Delta &= y(\boldsymbol{x}+\boldsymbol{e}) - y(\boldsymbol{x}) \\ &= \overline{\boldsymbol{w}}_{\text{eq}}{}^T(\boldsymbol{x}+\boldsymbol{e}) - \bar{b}_{\text{eq}} - \boldsymbol{w}_{\text{eq}}{}^T\boldsymbol{x} + b_{\text{eq}} \\ &= \overline{\boldsymbol{w}}_{\text{eq}}{}^T\boldsymbol{e} + \left[(\overline{\boldsymbol{w}}_{\text{eq}} - \boldsymbol{w}_{\text{eq}})^T\boldsymbol{x} - \left(\bar{b}_{\text{eq}} - b_{\text{eq}}\right)\right] \end{aligned} \tag{2.11}$$

We observe that the distortion can be split into two terms: (i) $\overline{\boldsymbol{w}}_{\text{eq}}{}^T\boldsymbol{e}$ that is identical to the distortion for a linear classifier, and can be analyzed within the theoretical framework of Section 2.4 and (ii) $(\overline{\boldsymbol{w}}_{\text{eq}} - \boldsymbol{w}_{\text{eq}})^T\boldsymbol{x} - \left(\bar{b}_{\text{eq}} - b_{\text{eq}}\right)$, that is determined by the ReLU units that flip due to the perturbation.

In the next section, we provide an analytical characterization of a "high SNR" regime in which the number of flipped switches is small, motivated by iterative attacks which gradually build up attack strength over a large number of iterations (with a per-iteration ℓ_∞-budget of $\delta \ll \epsilon$). When very few switches flip, the distortion is dominated by the first term in (2.11), and we can apply our prior results on linear classifiers to infer the efficacy of the sparsifying front end in attenuating the distortion. As we discuss via our numerical results, this creates a situation in which it might sometimes be better (depending on dataset and attack budget) for the adversary to try to make the most of network's nonlinearity, spending the attack budget in one go trying to flip a larger number of switches in order to try to maximize the impact of the second term in (2.11).

2.5.3 Characterizing the High SNR Regime

We now investigate the conditions that guarantee high SNR at neuron i, i.e. $\bar{s}_i = s_i$, where $\bar{s}_i$ denotes the switch when the adversary is present.

We observe that

$$\bar{s}_i = \begin{cases} 1 - s_i, & \boldsymbol{w}_i^T \boldsymbol{x} - b_i \in \big[\min\big(-\boldsymbol{w}_i^T \boldsymbol{e}, 0\big), \\ & \qquad \max\big(-\boldsymbol{w}_i^T \boldsymbol{e}, 0\big)\big] \\ s_i, & \boldsymbol{w}_i^T \boldsymbol{x} - b_i \notin \big[\min\big(-\boldsymbol{w}_i^T \boldsymbol{e}, 0\big), \\ & \qquad \max\big(-\boldsymbol{w}_i^T \boldsymbol{e}, 0\big)\big], \end{cases}$$

This implies the following sufficient condition for high SNR at neuron i: $\big|\boldsymbol{w}_i^T \boldsymbol{x} - b_i\big| > \big|\boldsymbol{w}_i^T \boldsymbol{e}\big|$.

Assumptions 1 *To establish our theoretical result, we make a few mild technical assumptions:*

1. *The data is normalized in ℓ_2-norm and bounded: $\|\boldsymbol{x}\|_2 = 1$ and $\|\boldsymbol{x}\|_\infty = \mathcal{O}(1)$.*
2. *The ℓ_∞ budget $\delta \leq |b_i|/\|\boldsymbol{w}_i\|_1 - C \;\; \forall\, i = 1, \ldots, M$ and for some $C = \Theta(1) > 0$. Note that this assumption is justified in an iterative/optimization-based attack, where the adversary gradually spends the budget over many iterations.*
3. *The number of neurons $M = \omega(1)$ as N gets large.*
4. *For each neuron $i = 1, \ldots, M$, we model the $\{w_i[k],\, k = 1, \ldots, N\}$ as i.i.d, with zero mean $\mathbb{E}[w_i[k]] = 0$. We assume that $\mathbb{E}\big[w_i[k]^2\big] = \sigma_i^2 = \Theta(1)$.*

Theorem 3 *With high probability, the high SNR condition ($\bar{\boldsymbol{s}} = \boldsymbol{s}$) holds for $1 - \mathcal{O}(1)$ fraction of neurons, i.e.*

$$\lim_{N \to \infty} \Pr\left(\frac{|S|}{M} = 1 - \mathcal{O}(1)\right) = 1,$$

where $S = \big\{i : \big|\boldsymbol{w}_i^T \boldsymbol{x} - b_i\big| > \big|\boldsymbol{w}_i^T \boldsymbol{e}\big|\big\}$.

Proof: We first state the following lemma:

Lemma 4 $\boldsymbol{w}_i^T\boldsymbol{x} \to \mathcal{N}(0,\sigma_i^2)$ *in distribution.*

Proof: We show that we can apply Lindeberg's version of the CLT, noting that $\boldsymbol{w}_i^T\boldsymbol{x} = \sum_{j=1}^N U_j$ is the sum of independent random variables, where $U_j = x_j\, w_i[j]$ with $\mathbb{E}[U_j] = 0$, $\mathrm{var}(U_j) = \sigma_i^2 x_j^2$ and $\sum_{j=1}^N \sigma_i^2 x_j^2 = \sigma_i^2$.

Now given a constant $c_1 = \Theta(1) > 0$, we investigate the following quantity in order to check the Lindeberg condition:

$$L(c_1, N) = \frac{1}{\sigma_i^2}\sum_{j=1}^{N} \mathbb{E}\Big[U_j^2 \mathbb{1}_{\{|U_j| > c_1\sigma_i\}}\Big]$$

From the assumptions on $\boldsymbol{w}_i$ and $\boldsymbol{x}$, we observe that

$$\begin{aligned}\mathbb{E}\Big[x_j^2\, w_i^2[j]\, \mathbb{1}_{\{|U_j|>\delta\sigma_i\}}\Big] &\le \mathcal{O}^2(1)\Theta(1)\Pr(|U_j| > c_1\sigma_i) \\ &= \mathcal{O}^2(1)\Theta(1)\Pr\left(|w_i[j]| > \frac{c_1\sigma_i}{\mathcal{O}(1)}\right)\end{aligned}$$

The ℓ_∞ assumption on $\boldsymbol{w}_i$ also implies that $\forall\, c_1 > 0,\ \exists\, N_0$ s.t. $|w_i[j]| < c_1\sigma_i/\,\mathcal{O}(1),\ \forall\, j \in \{1,\ldots,N\}$, $N > N_0$. Hence we obtain that $\lim_{N\to\infty} L(\delta, N) = 0$, which verifies the Lindeberg condition. ■

Noting that $\boldsymbol{w}_i^T\boldsymbol{x} - b_i \to \mathcal{N}(-b_i, \sigma_i^2)$, we can now write

$$\begin{aligned}&\Pr\big(|\boldsymbol{w}_i^T\boldsymbol{x} - b_i| > |\boldsymbol{w}_i^T\boldsymbol{e}|\big) \ge \Pr\big(|\boldsymbol{w}_i^T\boldsymbol{x} - b_i| > \delta\|\boldsymbol{w}_i\|_1\big) \\ &= \Pr\big(\boldsymbol{w}_i^T\boldsymbol{x} - b_i > \delta\|\boldsymbol{w}_i\|_1\big) + \Pr\big(\boldsymbol{w}_i^T\boldsymbol{x} - b_i < -\delta\|\boldsymbol{w}_i\|_1\big) \\ &= Q\left(\frac{\delta\|\boldsymbol{w}_i\|_1 + b_i}{\sigma_i}\right) + Q\left(\frac{\delta\|\boldsymbol{w}_i\|_1 - b_i}{\sigma_i}\right) \\ &\ge Q\left(\frac{\delta\|\boldsymbol{w}_i\|_1 - |b_i|}{\sigma_i}\right) = Q\left(\frac{\|\boldsymbol{w}_i\|_1}{\sigma_i}\left(\delta - \frac{|b_i|}{\|\boldsymbol{w}_i\|_1}\right)\right) \\ &= Q\left(\Theta(N)\left(\delta - \frac{|b_i|}{\|\boldsymbol{w}_i\|_1}\right)\right) \to 1 \quad \text{as} \quad N\to\infty,\end{aligned}$$

where $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-t^2/2}dt$ and $\delta < \frac{|b_i|}{\|\boldsymbol{w}_i\|_1}$ by Assumption 2. The theorem follows by using a union bound over $i = 1,\ldots,M$. ■

2.5.4 Attacks

We assume that the adversary knows the true label, a reasonable (mildly pessimistic) assumption given the high accuracy of modern neural networks. We consider attacks that focus on maximizing the first term in (2.11), using a high SNR approximation to the distortion:

$$\Delta = y(\hat{\boldsymbol{x}}) - y(\boldsymbol{x}) = \boldsymbol{w}_{\rm eq}{}^T \, \mathcal{P}_{\rm K}(\boldsymbol{e}, \boldsymbol{x}) = \boldsymbol{e}^T \, \mathcal{P}_{\rm K}(\boldsymbol{w}_{\rm eq}, \boldsymbol{x}),$$

Here $\boldsymbol{w}_{\rm eq} \approx \boldsymbol{w}_{\rm eq}(\boldsymbol{x}) = \sum_{i=1}^{M} s_i \, w_0[i] \, \boldsymbol{w}_i$ if we are applying a small perturbation to the input data. However, for iterative attacks with multiple small perturbations, $\boldsymbol{w}_{\rm eq}$ would evolve across iterations.

We can now define attacks in analogy with those for linear classifiers. The adversary can use an "effective input" $\boldsymbol{x}_1$ to compute the locally linear model $\boldsymbol{w}_{\rm eq} = \boldsymbol{w}_{\rm eq}(\boldsymbol{x}_1)$. For example, the adversary may choose $\boldsymbol{x}_1 = \boldsymbol{x}$ if making a small perturbation, or may iterate computation of its perturbation using $\boldsymbol{x}_1 = \boldsymbol{x} + \boldsymbol{e}$. The adversary can also use a possibly different "effective input" $\boldsymbol{x}_2$ to estimate the set of basis coefficients retained by the sparse front ends. Armed with this notation, we can define two attacks:

Semi-white box: $A_{\rm SW}(\boldsymbol{x}_1, \epsilon) = \epsilon \, \mathrm{sgn}(\boldsymbol{w}_{\rm eq}(\boldsymbol{x}_1))$,

White box: $A_{\rm W}(\boldsymbol{x}_1, \boldsymbol{x}_2, \epsilon) = \epsilon \, \mathrm{sgn}(\mathcal{P}_{\rm K}(\boldsymbol{w}_{\rm eq}(\boldsymbol{x}_1), \boldsymbol{x}_2))$.

We make no claims on the optimality of these attacks. They are simply sensible strategies based on the locally linear model, and as we show in the next section, they are more powerful than existing FGSM attacks for multiclass classification.

For simplicity, we set $\boldsymbol{x}_1 = \boldsymbol{x}_2$ for the white box attack, and simplify notation by denoting it by $A_{\rm W}(\boldsymbol{x}_1, \epsilon)$. A (suboptimal) default choice is to set $\boldsymbol{x}_1 = \boldsymbol{x}_2 = \boldsymbol{x}$, relying on a high SNR approximation for both the network switches and for the sparsifying front end. However, we can also refine these choices iteratively, as follows.

Iterative versions: We choose a particularly simple approach, in which we use a small attack budget δ to change $\boldsymbol{w}_{\text{eq}}$ by small amounts and update the "direction" of the attack, maintaining the overall ℓ_∞ constraint at each stage:

$$\boldsymbol{e}_{k+1} = \boldsymbol{e}_k + A(\boldsymbol{x} + \boldsymbol{e}_k, \delta)$$

$$\boldsymbol{e}_{k+1} = \text{clip}_\epsilon(\boldsymbol{e}_{k+1}),$$

where $\text{clip}_\epsilon(\boldsymbol{e}) \triangleq \max(\min(\boldsymbol{e}, \epsilon), -\epsilon)$. We believe there is room for improvement in how we iterate, but this particular choice suffices to illustrate the power of locally linear modeling.

Remark 6 *The Fast Gradient Sign Method (FGSM) attack puts its attack budget along the gradient of the cost function $J(\cdot)$ used to train the network. For binary classification and the cross-entropy cost function, we can show that it is equivalent to the semi-white box attack with $\boldsymbol{x}_1 = \boldsymbol{x}$. Specifically, we can show that*

$$\boldsymbol{e}_{\text{FGSM}} = \epsilon \,\text{sgn}(\nabla_{\boldsymbol{x}} J(\boldsymbol{x}, l)) = A_{\text{SW}}(\boldsymbol{x}, \epsilon)$$

where l is the true label, by verifying that the gradient is proportional to $\boldsymbol{w}_{\text{eq}}(\boldsymbol{x})$. For a larger number of classes, however, insights from our locally linear modeling can be used to devise more powerful attacks than FGSM.

2.5.5 Multiclass Classification

In this subsection, we consider a multilayer (deep) network with L classes. Each of the outputs of the network can be modeled using the analysis in the previous section as follows:

$$y_i = \boldsymbol{w}_{\text{eq}}^{\{i\}^T} \boldsymbol{x} - b_{\text{eq}}^{\{i\}}, \;\; i = 1, \ldots, L,$$

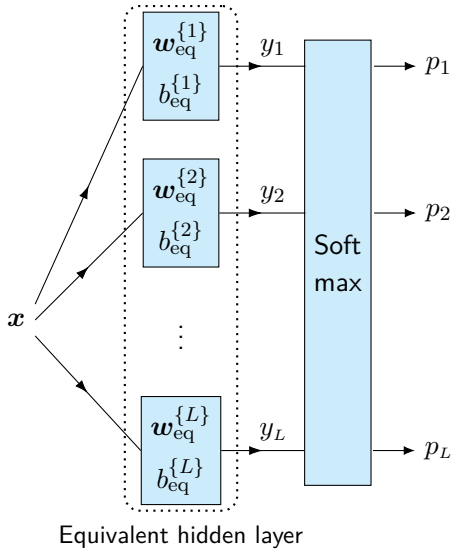


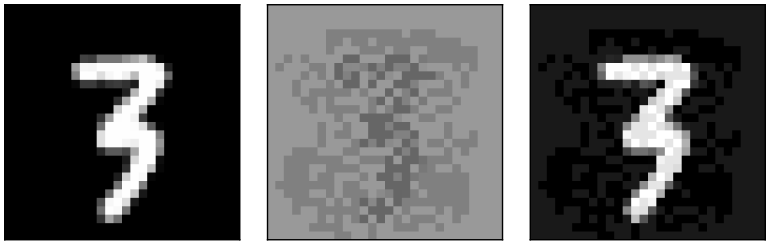
Figure 2.4: Multilayer (deep) neural network with L classes. Each of the L pre-softmax outputs (logits) is locally linear: $y_i = \boldsymbol{w}_{\text{eq}}^{\{i\}^T} \boldsymbol{x} - b_{\text{eq}}^{\{i\}},\ i = 1, \ldots, L$.

where $\boldsymbol{y} = [y_1, y_2, ..., y_L]^T$, $p_i = S_i(\boldsymbol{y})$, and the softmax function $S_i(\boldsymbol{y}) = e^{y_i} / \left(\sum_{j=1}^{L} e^{y_j} \right)$. Assume that $\boldsymbol{x}$ belongs to class t (with label t known to the adversary).

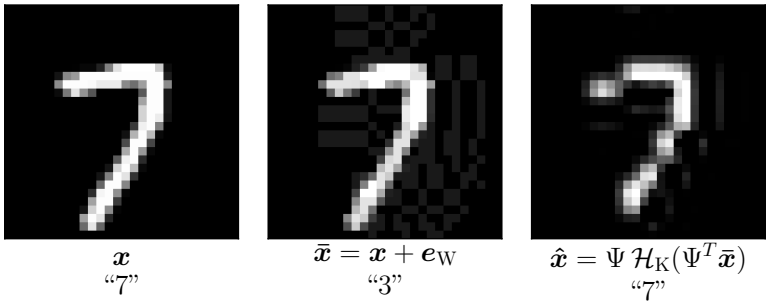
Locally linear attack: The adversary can sidestep the nonlinearity of the softmax layer, since its goal is simply to make $y_i > y_t$ for *some* $i \neq t$. Thus, the adversary can consider $L-1$ binary classification problems, and solve for perturbations aiming to maximize $y_i - y_t$ for each $i \neq t$. We now apply the semi-white and white box attacks, and their iterative versions, to each pair, with $\boldsymbol{w}_{\text{eq}} = \boldsymbol{w}_{\text{eq}}^{\{i\}} - \boldsymbol{w}_{\text{eq}}^{\{t\}}$ being the equivalent locally linear model from the input to $y_i - y_t$. After computing the distortions for each pair, the adversary applies its attack budget to the *worst-case* pair for which the distortion is the largest:

$$\max_{i, \boldsymbol{e}} \quad y_i(\boldsymbol{x} + \boldsymbol{e}) - y_t(\boldsymbol{x} + \boldsymbol{e}), \qquad \text{s.t.} \quad \|\boldsymbol{e}\|_\infty \leq \epsilon$$

FGSM: Unfortunately, the FGSM attack does not have a clean interpretation in the multi-class setting. Taking the gradient of the cross-entropy betwen one-hot encoded vector of the



(a) A natural image and its adversarial counterpart (misclassified as "7").



(b) After sparsification, the perturbed image is no longer adversarial.

Figure 2.5: Sample images depicting the interplay between attack and defense: tiny adversarial attacks can fool a classifier, but sparsity-based preprocessing can restore accuracy by projecting the attack down to a lower dimensional subspace.

true label $\boldsymbol{l}$ ($l[k] = \delta_{tk}$) and the final output of the model $\boldsymbol{p} = [p_1, p_2, ..., p_L]$, with $J(\boldsymbol{l}, \boldsymbol{p}) = -\sum_{i=1}^{L} l_i \log(p_i)$, we obtain

$$\boldsymbol{e}_{\mathrm{FGSM}} = \epsilon\,\mathrm{sgn}\left(\boldsymbol{w}_{\mathrm{eq}}^{\{t\}}(p_t - 1) + \sum_{k \neq t} \boldsymbol{w}_{\mathrm{eq}}^{\{k\}} p_k\right).$$

This does not take the most direct approach to corrupting the desired label, unlike our locally linear attack, and is expected to perform worse.

2.6 Experimental Results

In this section we demonstrate the effectiveness of sparsifying front ends on inference tasks on the MNIST [14] and CIFAR-10 [15] datasets. Code is available at https://github.com/soorya19/sparsity-based-defenses/. We first provide results for binary linear classifiers ("3" versus "7"), and then report on experiments with neural networks, for both binary and multi-class classification.

2.6.1 Linear Classifiers

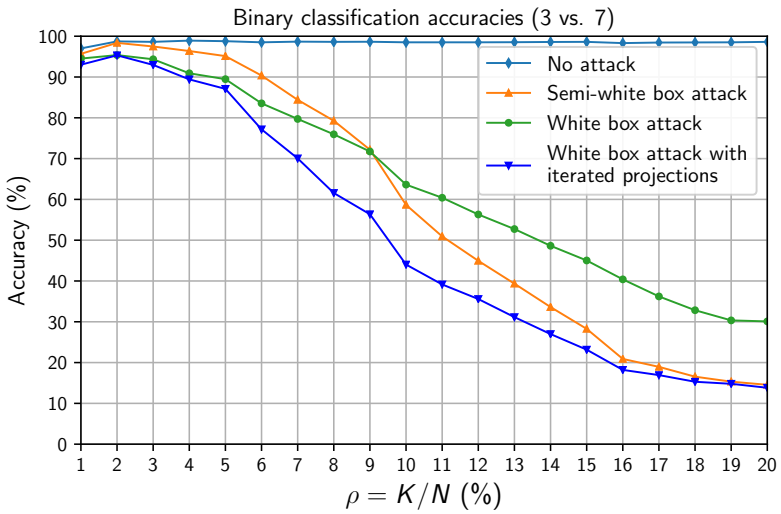
Set-up: Here we train a linear SVM $g(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b$ to classify digit pairs d_1 and d_2 from the MNIST dataset. The "direction" of the attack is opposite that of the correct class: if the SVM predicts class d_1 when $g(\boldsymbol{x}) < 0$ and d_2 when $g(\boldsymbol{x}) > 0$, the perturbation is of the form $\bar{\boldsymbol{x}} = \boldsymbol{x} + \epsilon\,\mathrm{sgn}(\boldsymbol{w})$ for images of class d_1, and $\bar{\boldsymbol{x}} = \boldsymbol{x} - \epsilon\,\mathrm{sgn}(\boldsymbol{w})$ for class d_2. We assume that the adversary has access to the true labels. For the defense, we use the Daubechies-5 wavelet [95] to perform sparsification and retrain the SVM with sparsified images (for various sparsity levels) before evaluating performance.

Results: We consider the case of 3 versus 7 classification. When no defense is present, an attack[1] with $\epsilon = 0.1$ completely overwhelms the classifier, with accuracy dropping from 98.64% to 0.25% as depicted in Fig. 2.5a. Fig. 2.5a shows a sample image before and after attack.

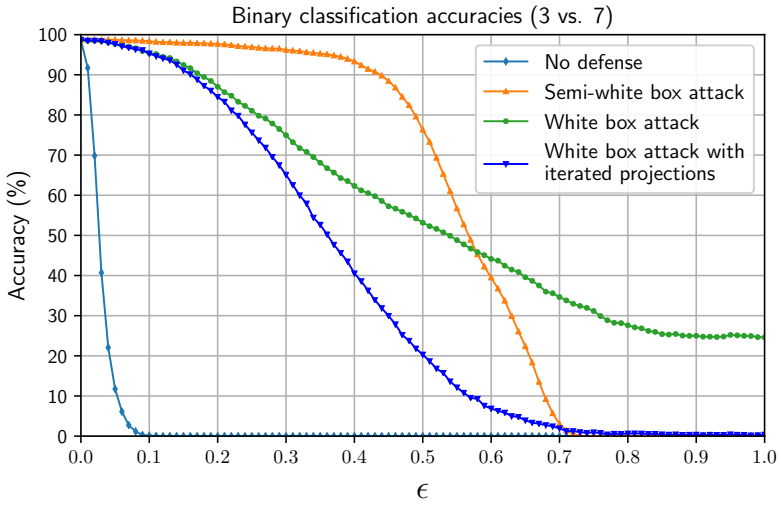
[1]The reported values of ϵ correspond to images normalized to $[0, 1]$.

Table 2.1: Binary classification accuracies for linear SVM, with $\epsilon = 0.1$ for attacks and $\rho = 2\%$ for defense.

	No defense	Sparsifying front end
Semi-white box attack	0.25	98.37
White box attack	0.25	95.37



(a) Accuracy vs. sparsity level, where the attacks use $\epsilon = 0.1$.



(b) Accuracy vs. attack budget, where the defense uses $\rho = 2\%$.

Figure 2.6: Binary classification accuracies for the linear SVM as a function of the sparsity level ρ and attack budget ϵ.

Insertion of the sparsifying front end confers resiliency to attacks: at low values of ρ, accuracy is restored to near-baseline levels. The optimal value of ρ must trade off signal distortion versus perturbation attenuation. We find $\rho = 2\%$ to be the best choice for the 3 versus 7 scenario, and report on the accuracies obtained in Table 2.1. Results for other digit pairs show a similar trend. Insertion of the front end greatly improves resilience to adversarial attacks. The optimal

value of ρ lies between $1-5\%$, with $\rho = 2\%$ working well for all scenarios.

To give a concrete feel of the front end at work, Fig. 2.5b shows an example image, the attacked image, and the attacked image after sparsification.

Fig. 2.6 reports on accuracy as a function of sparsity level ρ and attack budget ϵ. At the low values of ρ that we are interested in, the white box attack is more damaging than the semi-white box attack. At higher ρ, a white box attack performs worse than the semi-white box attack: the high SNR condition in Proposition 1 is no longer satisfied, hence the white box attack is attacking the "wrong subspace". This behavior can also be observed in Fig. 2.6b where attacks with larger ϵ violate the high SNR condition. As discussed in 2.4.3, it is easy to devise iterative white box attacks that do better by refining the estimate of the K-dimensional subspace in the following manner:

$$\boldsymbol{e}^{[i+1]} = \boldsymbol{e}^{[i]} + \delta \operatorname{sgn}\Big(\mathcal{P}_{\mathrm{K}}\Big(\boldsymbol{w},\, \boldsymbol{x} + \boldsymbol{e}^{[i]}\Big)\Big),$$

with $\delta < \epsilon$. Essentially, we refine our estimate of the support using multiple steps, and also reduce the step size in each iteration, so that the support does not vary too much in between iterations. We can observe from the figures that the attack with iterated projections performs better in the low SNR region. However, this scenario is not of practical interest, since front ends with large ρ do not provide enough attenuation of the adversarial perturbation, and perturbations with large ϵ are no longer visually imperceptible.

2.6.2 Neural Networks

For neural networks, we perturb images with the following attacks in the white box setting:

(a) the locally linear attack,

(b) its iterative version,

(c) FGSM [7],

(d) iterative FGSM [43],

(e) projected gradient descent (PGD) [12], and

(f) momentum iterative FGSM [50] (winner of the NIPS 2017 competition on adversarial attacks and defenses [49]).

For PGD, we use multiple random restarts and calculate accuracy over the most successful restart(s) for each image. We evaluate three versions of the attacks: one that uses the backward pass differential approximation (BPDA) technique of [51] to approximate the gradient of the front end as $\mathbf{1}$, a second version where the gradient is calculated as the projection onto the top K basis vectors of the input, and a third version where we iteratively refine the projection as described in the previous section. We report accuracies for the version that causes the most damage.

Set-up: For binary classification of MNIST digits "3" and "7", we use a 2-layer fully-connected network with 10 neurons. For multi-class MNIST, we use a 4-layer CNN consisting of two convolutional layers (with 20 and 40 feature maps, each with 5x5 filters) and two fully-connected layers (containing 1000 neurons each, with dropout) [96]. For CIFAR-10, we use a 32-layer ResNet [97] and follow the data augmentation strategy of [97] for training. For the sparsifying front end, we use the Daubechies-5 wavelet for binary MNIST, Coiflet-1 for multiclass MNIST and Symlet-9 for CIFAR-10, and retrain the networks with sparsified images.

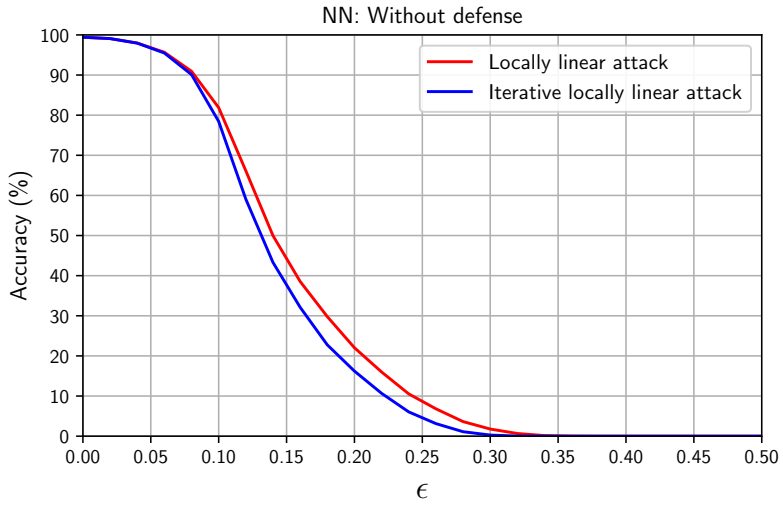
MNIST results: Figure 2.7 reports on binary classification accuracies for the 2-layer NN as a function of attack budget, showing that the front end improves robustness across a range of ϵ. As shown in Section 2.5.4, FGSM is identical to the locally linear attack for binary classification, so we do not label it in the figure. We note that images are normalized to the range $[0, 1]$, so by the end of the chosen range of ϵ, perturbations are no longer visually imperceptible.

Table 2.2: Multiclass classification accuracies for 4-layer CNN, with $\epsilon = 0.2$ for attacks and $\rho = 3.5\%$ for defense.

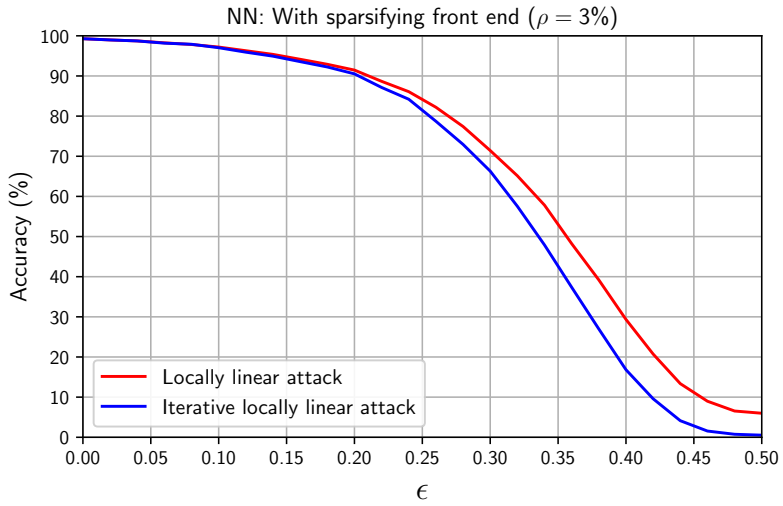
	No defense	Sparsifying front end
No attack	99.31	98.97
Locally linear attack	28.43	78.27
FGSM	42.01	85.35
Iterative locally linear attack	7.36	74.38
Iterative FGSM	6.34	74.97
Momentum iterative FGSM	6.99	73.55
PGD (100 random restarts)	5.12	61.04

Table 2.2 reports on multiclass classification accuracies for the 4-layer CNN, with attacks using an ℓ_∞ budget of $\epsilon = 0.2$. Iterative attacks are run for 1000 steps with a per-iteration budget of $\delta = 0.01$, except for PGD and the iterative locally linear attack which use 100 steps of $\delta = 0.05$. Without any defense, a strong adversary can significantly degrade performance, reducing accuracy from 99.31% to 5.12%. In contrast, when a sparsifying front end is present (with sparsity level $\rho = 3.5\%$), the network robustness measurably improves, increasing accuracy to 61.04% in the worst-case scenario. We note that the locally linear attack is stronger than FGSM, and the iterative locally linear attack is competitive with single runs of the other iterative attacks.

Tables 2.3 compares the front end defense with the state-of-the-art empirical defense of [12] and the provable defenses of [62] and [65]. We run the adversarial training defense on our network architecture and report on accuracies obtained. For [62] and [65], we use publicly available pretrained models and report both empirical and certified performance. When the attack budget is small ($\epsilon = 0.1$), the sparsifying front end is competitive with existing defenses, with better accuracies than the provable defense of [62]. At the higher budget of $\epsilon = 0.2$, the adversarial training defense of Madry et al provides better performance. The front end still



(a) Accuracy vs. attack budget, when no defense is present.



(b) Accuracy vs. attack budget, with sparsifying front end ($\rho = 3\%$).

Figure 2.7: Binary classification accuracies for 2-layer NN as a function of ϵ (with the iterative attack using 1000 steps).

offers a measurable degree of robustness, with the advantage that it is computationally much less expensive than adversarial training, which requires one to retrain the network with adversarial images, resulting in an M-fold slowdown if we use M steps for the attack [12]. For $\epsilon = 0.3$, the performance of the sparsifying front end drops to 13.15%. This is not entirely surprising, since our theoretical guarantees apply for a high SNR regime corresponding to relatively small

Table 2.3: Comparison to other defenses on MNIST. For defenses with certified bounds, numbers in blue denote lower bounds on adversarial accuracy. Numbers in black denote PGD attack accuracy with 100 steps and 100 random restarts.

Attack budget	Defense	Adversarial accuracy	Natural accuracy
$\epsilon = 0.1$	Sparsifying front end	92.18	98.97
	Madry et al	95.83	99.54
	Raghunathan et al	88.00, 65.00	95.72
	Wong et al	97.33, 96.33	98.81
$\epsilon = 0.2$	Sparsifying front end	61.04	98.97
	Madry et al	92.95	99.20
$\epsilon = 0.3$	Sparsifying front end	13.15	98.97
	Madry et al	90.57	99.15
	Wong et al	86.21, 54.34	88.84

perturbations. We believe this is because sparse projections alone, especially with an off-the-shelf basis, are not "nonlinear enough" to discriminate between desired signal and perturbation for large attack budgets.

We have also performed experiments that combine the front end with adversarial training, with the result that accuracy improves from 61.04% (when using the front end alone) to 80.07% at $\epsilon = 0.2$. We note that there is no performance increase compared to pure adversarial training if the same value of ϵ is used for both training and testing. However, sparsification seems to provide some robustness to mismatch between training and test conditions; in particular, if we test at a value of ϵ larger than what the network was trained for. For example, accuracy increases from 35.65% (using adversarial training alone) to 69.04% when we train at $\epsilon = 0.1$ and test at $\epsilon = 0.2$. These results are obtained using a 100-step PGD attack with 100 random restarts; in the training phase, we use a 40-step PGD attack as in [12].

CIFAR-10 results: Table 2.4 reports on CIFAR-10 accuracies with a 32-layer ResNet. Front end sparsification is performed after converting color images to the (Y, C_b, C_r) decorrelated color

Table 2.4: CIFAR-10 accuracies for ResNet-32 at $\epsilon = 2/255$. Defenses are tested with a 1000-step PGD attack.

Defense	Adversarial accuracy	Natural accuracy
No defense	11.32	91.11
Sparsifying front end	39.60	62.30
Adversarial training	66.82	88.63
Front end + adv. training	67.21	88.02

space and then projecting to the wavelet basis. Since human vision is relatively less sensitive to chrominance, we impose higher sparsity in the C_b and C_r axes.

When using the front end alone, we can improve robustness from 11.32% to 39.60% at sparsity levels of (3.5, 1, 1)%. However, there is a drop in natural accuracy due to the high level of sparsity imposed. We can significantly improve both metrics by combining the front end defense, using less drastic sparsity levels, with adversarial training. Specifically, a front end with sparsity levels (50, 25, 25)%, together with adversarial training using 5 steps of PGD, results in 67.21% adversarial and 88.08% natural accuracies, respectively. We note that the accuracy under attack is 0.4% better than using adversarial training alone.

Chapter 3

Robust Compressive Front Ends for Learning

This chapter explores the use of compressive sensing based front ends for downstream learning and inference tasks. Section 3.1 provides a brief review of the compressed sensing principle. Since measurement matrices with random binary coefficients are effective, compressive front ends can be implemented in low-power hardware, as described in Section 3.2. Such an approach should be robust to stochastic noise in hardware, and be amenable to design constraints that may require localized computations. Section 3.3 presents a synthetic model of the measurement process, followed by analysis of stochastic nonlinearities and row-wise computations, showing that compressive information acquisition is robust to these impairments and constraints.

3.1 Background

A generic approach to dimension reduction for high-dimensional data with an unknown low-dimensional structure is to use compressive transformations that pseudo-randomly project observations to a low-dimensional subspace. Compressive sensing theory [98, 99] states that if

an N-dimensional signal $\boldsymbol{x}$ is K-sparse in an orthonormal basis Ψ,

$$\boldsymbol{x} = \Psi \boldsymbol{s}, \qquad \|s\|_0 \leq K$$

then it can be recovered from $\mathcal{O}(K \log(N/K))$ projections which can be chosen independently of the specific signal realization. Specifically, M-dimensional measurements $\boldsymbol{y}$ of the form

$$\boldsymbol{y} = \Phi \boldsymbol{x} = \Phi \Psi \boldsymbol{x}$$

are effective when the overall matrix $\Phi\Psi$ satisfies the restricted isometry property (RIP).

Definition 1 *A matrix $A \in \mathbb{R}^{M \times N}$ is said to satisfy the restricted isometry property of order K if there exists $\delta_K \in (0, 1)$ such that*

$$1 - \delta_K \leq \frac{\|A\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} \leq 1 + \delta_K \qquad \forall \boldsymbol{x} \;\; s.t \;\; \|\boldsymbol{x}\|_0 \leq K. \tag{3.1}$$

For example, matrices with i.i.d. subgaussian entries can be shown to satisfy the RIP with $\delta_K \approx 0$, with high probability, when $M = \mathcal{O}(K \log(N/K))$. Such measurement matrices, along with recovery algorithms such as ℓ_1 minimization [98], iterative hard thresholding (IHT) [100] and orthogonal matching pursuit (OMP) [101], provide strong recovery guarantees. In particular, elements of Φ can be chosen ± 1 with equal probability, hence compressive signal representations can be obtained at low complexity using binary matrix transformations. As shown by Eftekhari *et al.* [21], it is also possible to satisfy the RIP with compressive projections which operate on portions of the original vector (i.e., with block diagonal Φ), which is attractive for hardware implementations and is directly relevant for our case study, as discussed later.

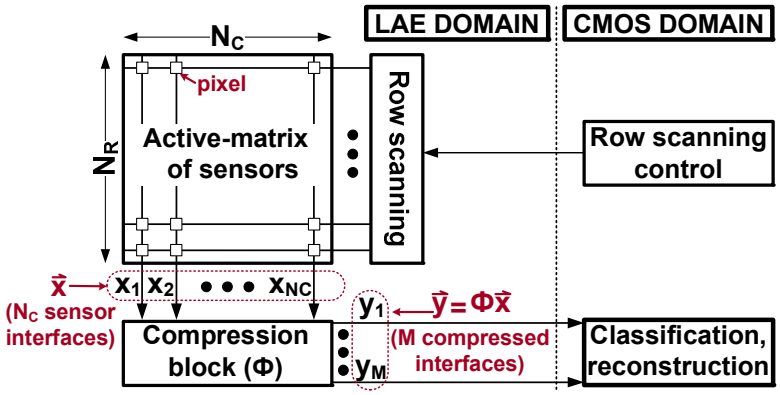


Figure 3.1: Architecture of image acquisition system [8].

3.2 Image Acquisition System

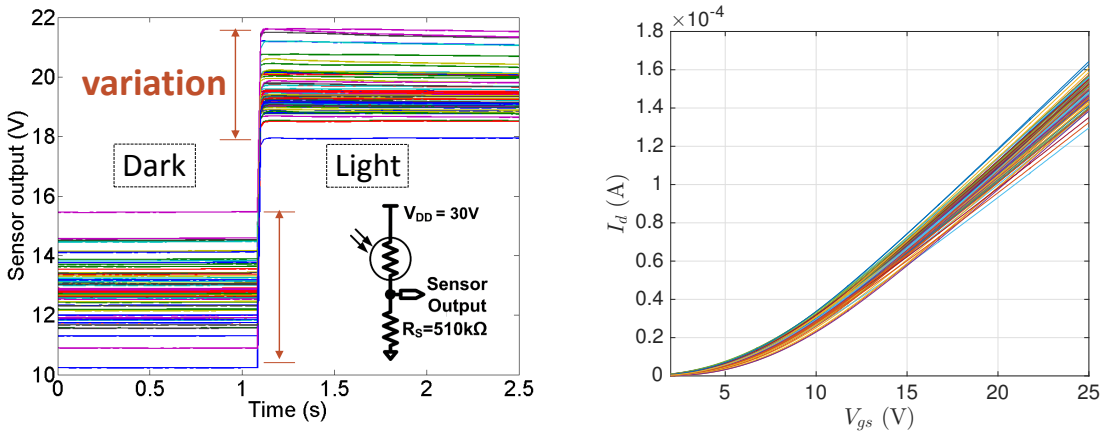
A practical example of a low-power compressive front end is the LAE-based image acquisition platform [8] in Fig. 3.1. Images are projected onto an array of photoconductor sensors, and illumination levels are sensed as voltages. The matrix of sensors is then scanned row-by-row, and each row $\boldsymbol{x}$ is fed into a compression block. The compression matrix, which is the same for each row, consists of ± 1 elements, implemented via highly variable, low-performance thin-film transistors (TFTs). The compressed signal $\boldsymbol{y}$ is collected for all rows and then used for classification and reconstruction tasks.

There are two sources of noise in the measurement process (Fig. 3.2): variations in the image sensors, and variations in I_d vs. V_{gs} curves across TFTs in the compression block.

3.3 Synthetic Model

We model the nonlinear measurement process as follows:

1. Transform each row of the image $\boldsymbol{x}$ to the range $[10, 22]$



(a) (b)

Figure 3.2: Sources of noise in the measurement process: (a) Variation in the output voltage of image sensors, for low and high input illumination, (b) I_d vs. V_{gs} curve for TFTs in the compression block, showing variation over 80 devices.

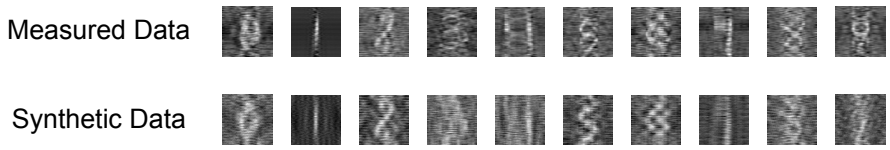


Figure 3.3: Images reconstructed from 5x compression: synthetic vs. measured data.

2. Add piecewise uniform noise, with parameter c (nominal value = 5)

$$\tilde{\boldsymbol{x}} = \begin{cases} u(10, 10+c) & \boldsymbol{x} \leq 10+c \\ u(\boldsymbol{x}-c/2, \boldsymbol{x}+c/2) & 10+c \leq \boldsymbol{x} \leq 22-c \\ u(22-c, 22) & 22-c \leq \boldsymbol{x} \end{cases}$$

where $u(a,b)$ denotes a realization of the uniform random variable $\mathcal{U}(a,b)$.

3. Perform random scaling from a set of 80 functions, and then compress:

$$\boldsymbol{y} = \Phi f_R(\tilde{\boldsymbol{x}})$$

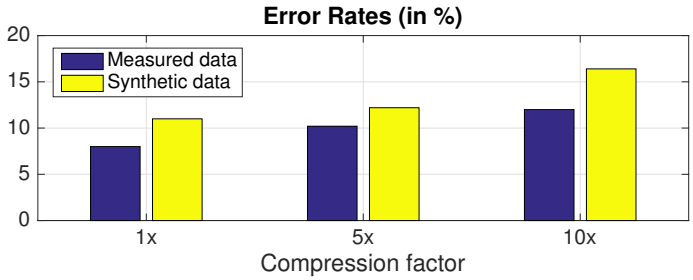


Figure 3.4: Classification performance: synthetic vs. measured data.

Overall, this results in the following synthetic model for each row of the image:

$$\boldsymbol{y} = \Phi f_R(\boldsymbol{x} + n(\boldsymbol{x})) \tag{3.2}$$

3.3.1 Verification with Measured Data

We first attempt to verify the synthetic model via comparisons of reconstruction and classification performance with measured data. Measurements are available for 1500 MNIST images (resized to 80×80). We perform RBF-SVM classification on the measured and synthetic data using 1000 training and 500 test images, and reconstruction via the GPSR algorithm [102] with the assumption that the data is sparse in the 2D-DFT basis. Reconstruction performance (Fig. 3.3) is qualitatively similar for both measured and synthetic data, while classification performance (Fig. 3.4) indicates that the synthetic model is pessimistic.

Now we proceed to analyze the impact of hardware impairments. We divide our analysis into two parts: impact of row-by-row compression, and stochastic nonlinearities.

3.3.2 Row-by-row compression

Eftekhari *et al.* [21] study block diagonal compressive matrices operating on portions of the signal, and show that RIP properties depend on the characteristics of the basis in which the signal is sparse. These results are directly relevant here, since we consider row-by-row compressive

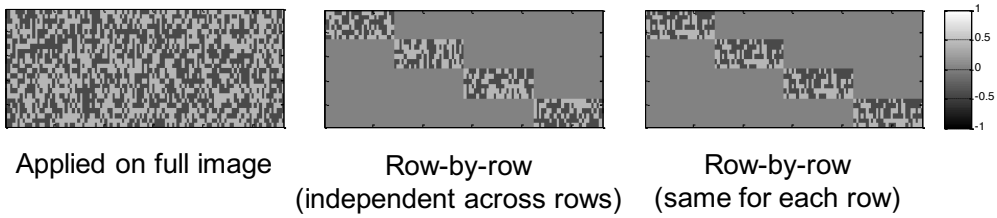


Figure 3.5: Three types of compression matrices.

measurements for an image. Specifically, we consider 3 types of binary compression matrices (Fig. 3.5): (1) Compression on the full image; (2) row-by-row compression with independent matrices across rows; (3) row-by-row compression with the same compressive matrix for each row.

The results in [21] show that matrices of types 2 and 3 can satisfy RIP with high probability, but the minimum number of measurements required scales linearly with the *coherence* of the sparsifying basis for type 2, and with its *block coherence* for type 3. Definitions can be found in Appendix B.

While we do not exactly know the sparsifying basis for images, we can gain design intuition

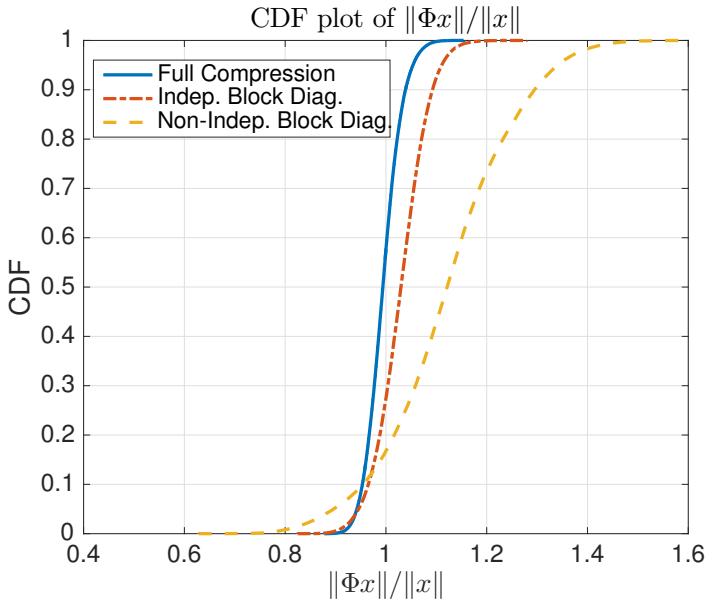


Figure 3.6: CDF of $\|\Phi \boldsymbol{x}\|/\|\boldsymbol{x}\|$ over 60,000 MNIST images (upscaled to 80×80)

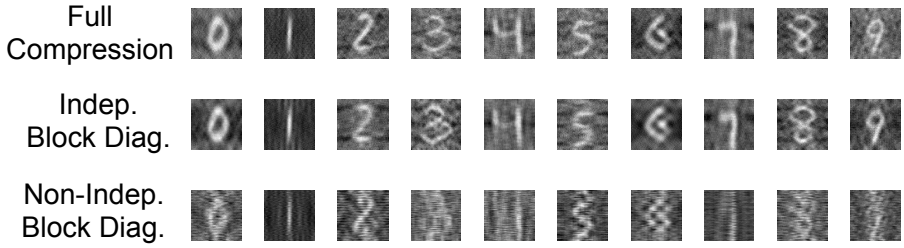


Figure 3.7: Images reconstructed from 5x compression with different matrix types.

by assuming that images are sparse in the 2D-DFT basis. The 2D-DFT basis has small coherence and large block coherence (see Appendix), so that type 2 matrices (independent across rows) are expected to perform better than type 3 matrices (used in the experimental system). We test this hypothesis by plotting the CDF of $\|\Phi\boldsymbol{x}\|/\|\boldsymbol{x}\|$ over 60,000 MNIST images (Fig. 3.6). We see that $\|\Phi\boldsymbol{x}\|/\|\boldsymbol{x}\|$ deviates from 1 when the compressive matrix is the same for each row, but that use of independent matrices leads to near-RIP behavior.

Next we compare the 3 matrices with respect to GPSR reconstruction (Fig. 3.7) and linear SVM classification, using 60,000 images for training and 10,000 for testing (Fig. 3.8). Reconstruction and classification results show the same trend as for geometry preservation, especially at higher compression factors.

3.3.3 Stochastic nonlinearities

In order to derive insight into how the nonlinearities are impacting performance, we analyze the synthetic noise $z = \Phi f_R(\boldsymbol{x} + n(\boldsymbol{x})) - \Phi f_0(\boldsymbol{x})$ by plotting a histogram over 60,000 MNIST

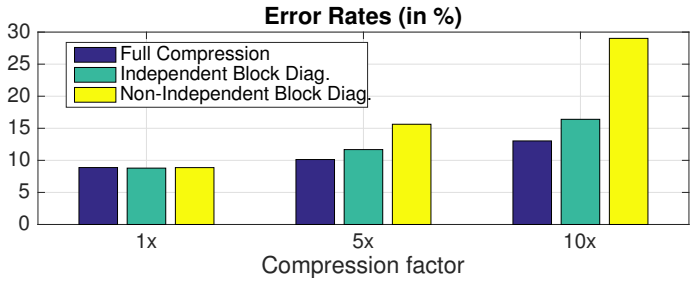


Figure 3.8: Classification performance for different matrix types.

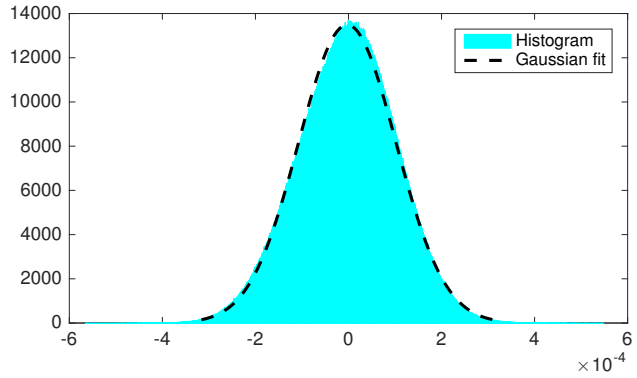


Figure 3.9: Histogram of synthetic noise.

images (Fig. 3.9). We observe the close match with a Gaussian, and therefore propose a simplified noise model:

$$\boldsymbol{y} = \Phi f_0(\boldsymbol{x}) + \boldsymbol{n}_{\text{Gauss}} \tag{3.3}$$

The variance of n_{Gauss} is estimated from the histogram.

To verify our simplified model, we compare reconstruction and classification performance (Figs. 3.10 and 3.11), via GPSR reconstruction and linear SVM classification with 60,000 MNIST images for training and 10,000 for testing. (Images are compressed using matrix type 2). Classification results indicate that the Gaussian model is slightly optimistic, but it is still a reasonable approximation that provides insight into the effect of stochastic impairments.

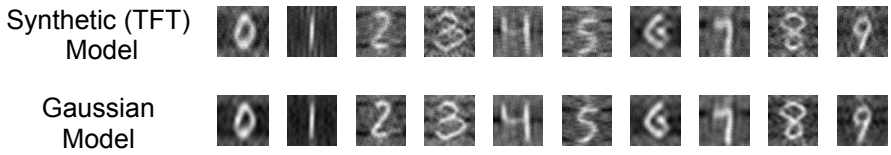


Figure 3.10: Images reconstructed from 5x compression (matrix type 2): synthetic vs. Gaussian model.

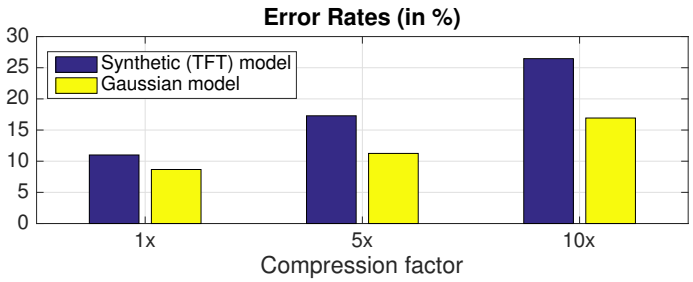


Figure 3.11: Classification performance: synthetic vs. Gaussian model.

Chapter 4

Robust RF Signatures

This chapter studies the robustness of radio frequency (RF) signatures that can distinguish between devices sending exactly the same message, by focusing on subtle hardware impairments unique to each device. Our goal is to extract such signatures in a protocol-agnostic fashion via supervised learning. Section 4.1 provides background on nonlinear effects that can contribute towards such signatures, along with a discussion of prior work in this area. We describe the complex-valued network architecture we use in Section 4.2 and explore the effect of different design choices for data from two wireless protocols: WiFi and ADS-B. In Section 4.4, we demonstrate the pitfalls of usng the entire packet to learn signatures, in which case networks rely on easily spoofed components of data. Finally, the impact of noise on performance is detailed in Section 4.3, motivating the use of noise augmentation to learn robust signatures.

4.1 Background

A generic model for a radio frequency (RF) wireless transmitted signal (shown in Fig. 4.1) is as follows:

$$s_{RF}(t) = s_c(t)\cos 2\pi f_c t - s_s(t)\sin 2\pi f_c t$$

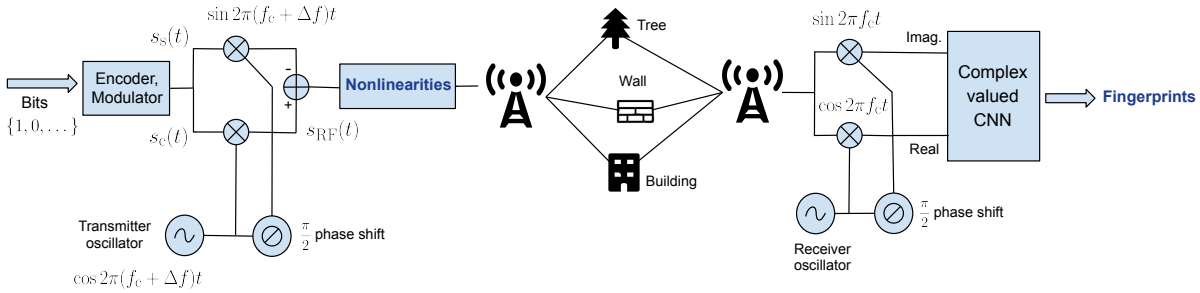
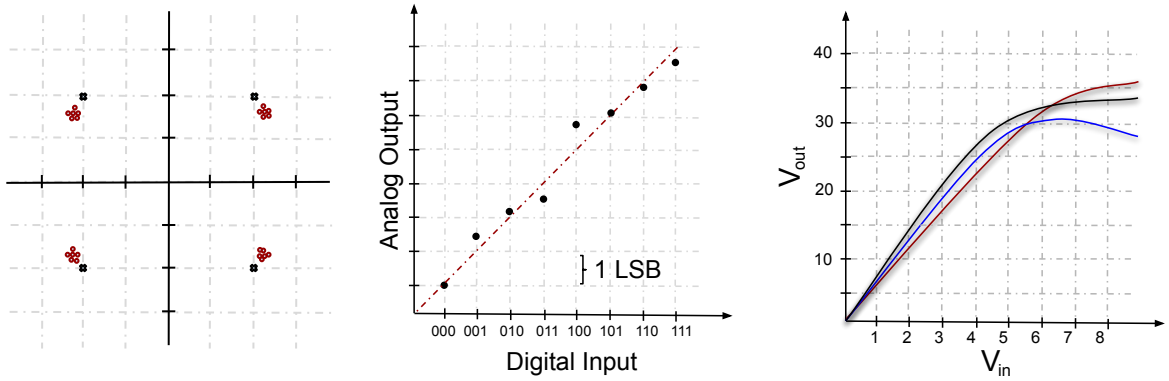


Figure 4.1: Block diagram of a wireless communication system.

where f_c denotes the *carrier* frequency, or the frequency of the electromagnetic wave that "carries" the information-bearing waveforms s_c (riding on the cosine of the carrier) and s_s (riding on the sine of the carrier). Typical parameters for WiFi, for example, are f_c of 2.4 or 5.8 GHz, and s_c, s_s having bandwidths of 20 MHz.

The receiver strips the carrier away to recover $s_c(t)$ and $s_s(t)$, and then processes them to decode the information bits that they carry. For a typical wireless channel, there are multiple paths from transmitter to receiver, so multiple delayed, attenuated and phase-shifted versions of the transmitted waveform sum up at the receiver. These transformations are best modeled by thinking of the information-bearing waveform as a complex-valued signal, $s(t) = s_c(t) + js_s(t)$, where $j = \sqrt{-1}$. The effect of a wireless channel is then modeled as a complex-valued convolution. The carrier frequency used at the receiver is not precisely the same as at the transmitter, and the impact of such carrier frequency offset (CFO) is also most conveniently modeled in the complex domain.

While RF processing is designed to produce as little distortion as possible, in practice, there are nonlinearities, typically with some characteristics unique to each transmitter because of manufacturing variations, which can in principle provide RF signatures. Variations in components such as DACs and PAs are inevitable even for transmitters manufactured using exactly the same process. Transistors, resistors, inductors, and capacitors within a device vary around nominal values, typically within a designed level of tolerance, and the goal is to translate the resulting



(a) (b) (c)

Figure 4.2: (a) Scatterplots of noisy QPSK constellation points with and without I-Q imbalance, (b) Example of DAC non-ideality, (c) Example variations of PA nonlinearities across transmitters.

variations in transmitter characteristics into a device signature. We discuss here some example effects, depicted in Figure 4.2, that may contribute towards such a signature.

- *I-Q Imbalance:* This results from mismatch in the gain and phase of the in-phase (I) and quadrature (Q) signal paths for upconversion. The phase of the cosine and sine of the carriers may not be offset by exactly $\pi/2$, and the path gains along the branches may not be equal.

- *Differential Nonlinearity (DNL) due to DAC:* DNL is defined as the discrepancy between the ideal and obtained analog values of two adjacent digital codes due to circuit component non-idealities [103].

- *Power Amplifier Nonlinearity:* Power amplifiers (PAs) are ideally linear, but start saturating at high input voltages. There is a significant literature on PA modeling [104, 105, 106, 107], as well as on the impact of PA nonlinearities on communication systems with high dynamic range such as OFDM [108, 109]. A common model is a memoryless polynomial

fit (typically up to third order) of the form:

$$y(t) = a_1 x(t) + a_2 x^2(t) + a_3 x^3(t) + ... + a_n x^n(t) \tag{4.1}$$

Recent promising results on wireless fingerprints for PA nonlinearities, extracted using DNNs, are reported in [110].

Of course, we seek to devise DNNs that extract signatures based on a combination of characteristics such as those in Figure 4.2. It is possible to extract signatures from either the transient (microsecond-length) signals transmitted during the on/off operation of devices, or via the steady-state packet information present in between the start and end transients. We focus here on work that employs the steady-state method since it is of more practical utility [25]. Work in this area can be broadly divided into two categories: traditional approaches that use handcrafted features as device characteristics, and techniques that employ machine learning to obtain fingerprints.

4.1.1 Traditional approaches

Remote physical device fingerprinting using small, microscopic deviations in device hardware called clock skews was introduced in [111]. The clock skew of a single device was observed to be fairly consistent over time, but clock skews varied significantly across devices, enabling fingerprinting. For wired devices in wide area networks, [111] estimated clock skews using TCP/IP packet headers. This technique was extended by [26] to wireless local area networks where more accurate measurements are possible from the Time Synchronization Function timestamps in IEEE 802.11 frames. However, these two detection methods were defeated by [27] which devised attacks to spoof the clock skew of a fake device to mimic that of a real one. Using more parameters such as jitter and fitting errors to measure the authenticity of the skew can mitigate these spoofing attempts. More recently, [32] proposed using the carrier frequency offset (CFO)

as a long-term device fingerprint, with the offset estimated using channel state information (CSI) measurements. While application layer spoofing of CFO is difficult [32], using the CFO as a mechanism for physical security has two key drawbacks: first, it does not provide a stable signature, since the oscillator frequency drifts over time; second, an adversary manipulating baseband signals can easily alter the CFO.

4.1.2 Machine learning based approaches

The first use of discriminatory classifiers for fingerprinting was in [25], which used a k-nearest neighbor (k-NN) classifier after preprocessing WiFi data to extract the log-spectral-energy of the preamble. A different preprocessing step was proposed in [24], involving demodulation error metrics such as frequency offset and I/Q offset, followed by a support vector machine (SVM). For the ADS-B air traffic control protocol, [28] performed k-means clustering on features based on inter-arrival times of aircraft position, velocity and identification messages. A similar inter-arrival approach was shown in [29] to be effective for WiFi fingerprinting, with a real-valued neural network (NN) operating on the extracted features. In [30], the carrier phase offset of ADS-B signals was used as input to an NN to learn fingerprints. For IEEE 802.15.4 ZigBee devices, [31] proposed the use of a real-valued CNN operating on an error signal obtained by subtracting out the ideal estimated signal from received data. These techniques work well, but they rely on protocol-specific signal modeling and preprocessing prior to learning, in contrast to our approach.

A purely learning based approach was studied in [33, 112], albeit for modulation recognition and not device fingerprinting. Each packet was sliced into multiple training examples using sliding windows, with the real and imaginary parts of complex data treated as independent channels. These were then input to a real-valued CNN capable of recognizing different analog and digital modulation types. The use of a real-valued CNN for WiFi device fingerprinting

was studied in [34], with sliding window preprocessing similar to prior work. As discussed in Section 1.3, our proposed method of learning complex-valued representations has potential generalization benefits over real-valued approaches [35].

4.2 Architecture

4.2.1 Overview

We use neural networks with complex-valued weights and biases to learn features from complex-valued wireless signals. Such complex-valued embeddings have found use in speech, music and vision tasks [113, 9]. Here we employ the framework of [9] which performs complex backpropagation by using partial derivatives of the cost with respect to the real and imaginary parts of each parameter. We make use of 1D complex convolutional layers with the following choices of activation functions (depicted in Fig. 4.3):

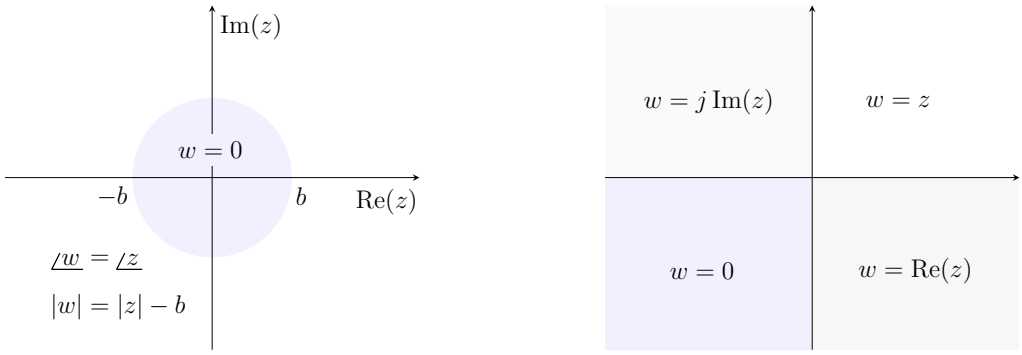
- *ModReLU* - This function preserves input phase and affects only the absolute value. Here b is a learned bias.

$$\text{ModReLU}(z) = \max(|z| - b, 0)\ e^{j\angle z}.$$

- *CReLU* - Unlike ModReLU this function does not preserve phase, with separate ReLUs applied on the real and imaginary parts of the input. The phase of the output is therefore limited to $[0, \pi/2]$.

$$\text{CReLU}(z) = \max(\text{Re}(z), 0) + j \max(\text{Im}(z), 0).$$

The loss in phase information can be potentially compensated by using filters with a larger number of channels that are capable of providing phase derotation.



(a) $w = \text{ModReLU}(z)$ (b) $w = \text{CReLU}(z)$

Figure 4.3: ModReLU and CReLU activation functions in the complex plane. ModReLU preserves the phase of all inputs outside a disc of radius b, while CReLU distorts all phases outside $[0, \pi/2]$ (the first quadrant). Figure adapted from [9].

Fig. 4.4 depicts a sample complex convolutional architecture for ADS-B signals. We use a series of complex 1D convolutions followed by an $|\cdot|^2$ layer to convert complex representations to real ones, and then a series of real-valued layers after a temporal averaging layer to obtain the fingerprint.

4.2.2 Performance

We provide results for an external database for two different wireless protocols:

- WiFi data containing a mix of IEEE 802.11a ($f_c = 5.8$ GHz) and 802.11g ($f_c = 2.4$ GHz) from commercial off-the-shelf devices, with a signal bandwidth of 20 MHz.
- ADS-B narrowband air traffic control signals ($f_c = 1.09$ GHz, narrowband) collected in the wild from 100 airplanes.

We start by using only the preamble for fingerprinting, with signals normalized to unit power. When sampled at 20 MHz, the length of the preamble is 320 I/Q samples for both protocol types.

We report accuracies for the following networks:

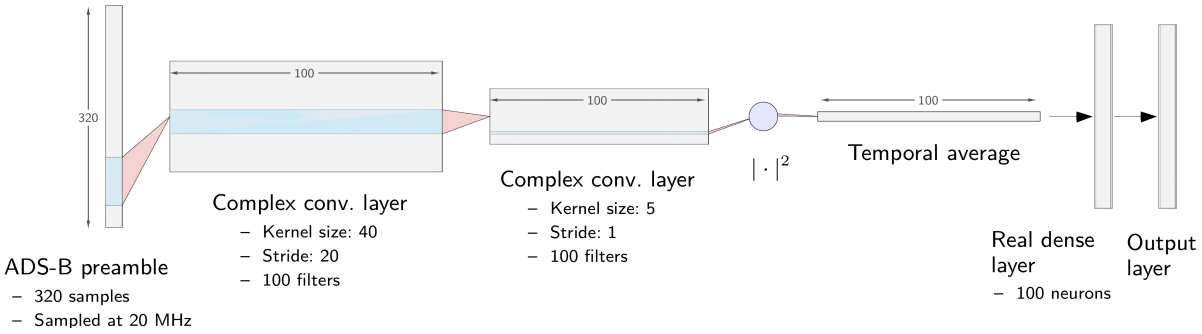


Figure 4.4: Complex-valued 1D CNN architecture for ADS-B signals.

- *ADS-B*: $100\,C\,40 \times 20 - 100\,C\,5 \times 1 - |\cdot|^2 - \text{Avg} - 100\,D$.
- *WiFi*: $100\,C\,20 \times 10 - 100\,C\,10 \times 1 - |\cdot|^2 - \text{Avg} - 100\,D$.

The notation should be read as follows:

- $\langle$number of filters$\rangle$ C $\langle$convolution size$\rangle \times \langle$stride$\rangle$
- $\langle$number of neurons$\rangle$ D

where C denotes a convolutional layer and D a fully connected layer, with complex-valued layers prior to the $|\cdot|^2$ layer and real-valued layers afterward. 'Avg' denotes a temporal averaging layer. We train networks for 200 epochs with a batch size of 100, using the Adam optimizer with default hyperparameters and ℓ_2 regularization constant of 10^{-3}.

We achieve 99.53% fingerprinting accuracy for 19 WiFi devices without channel distortion, using 200 samples per device for training and 100 for testing. For the ADS-B protocol, we obtain 81.66% accuracy with 100 devices (using 400 samples per device for training and testing). Fig. 4.5 compares the convergence of ModReLU and CReLU architectures. Both activation functions have similar convergence time, with ModReLU resulting in slightly higher accuracy for both the training and test sets.

Table 4.1 compares the performance of complex-valued and real-valued networks (for which real and imaginary parts of data are treated as different channels). If we fix the number of feature maps, a complex filter would contain twice as many parameters as an equivalent real

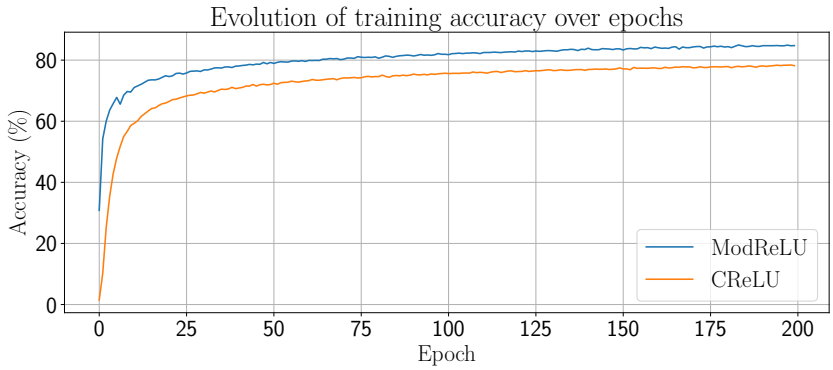


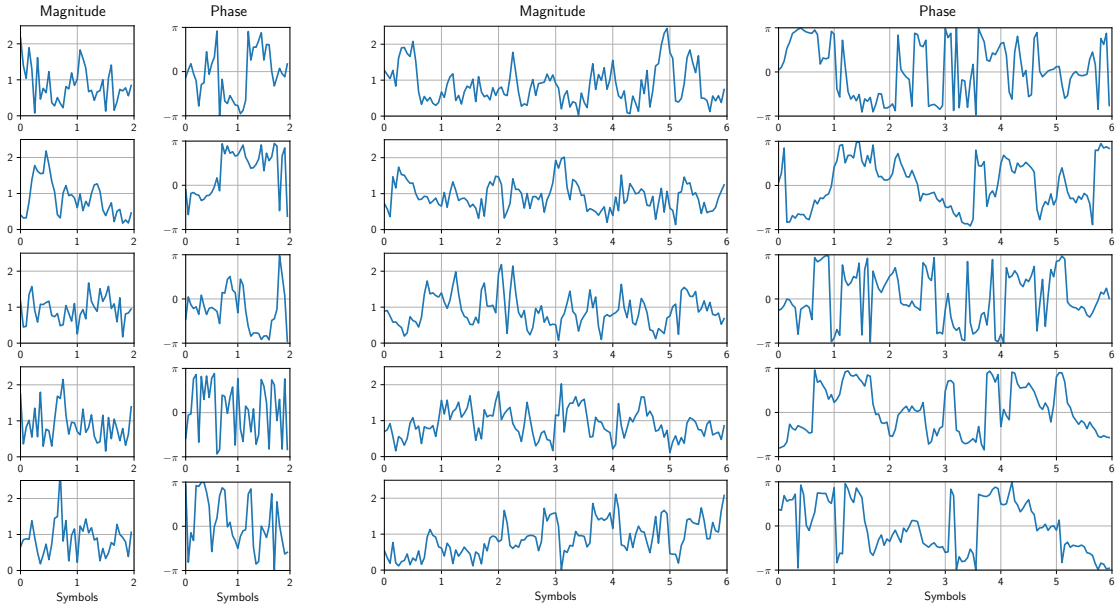
Figure 4.5: Evolution of training accuracy over epochs for ModReLU and CReLU architectures (ADS-B, 100 devices). ModReLU provides a small (5%) gain in train and test accuracies over CReLU, with similar convergence behavior.

filter. Therefore, we consider real networks where the number of channels is scaled by factors of 1, 1.4 and 2. We find that the complex network outperforms its real counterparts, with a gain in accuracy of 6.66% for ADS-B and 1.64% for WiFi.

Fig. 4.6 visualizes the first and second convolutional layer of the ADS-B architecture, showing the input signal that maximizes the activations of each filter. Since transmitter-characteristic nonlinear effects manifest themselves primarily in short-term transitions of amplitude and phase, the filters in the first layer can capture these effects by spanning a small multiple of the symbol

Table 4.1: Performance comparison between networks with complex and real weights (when using only the preamble).

Dataset	Network type	Accuracy	No. of real parameters
ADS-B	Complex	**81.66**	128,400
	Real	73.84	78,400
	Real (1.4x)	73.25	133,680
	Real (2x)	75.00	246,600
WiFi	Complex	**99.53**	216,219
	Real	97.32	116,219
	Real (1.4x)	97.53	217,899
	Real (2x)	97.89	430,419



(a) Layer 1 (b) Layer 2

Figure 4.6: Visualizations of the first and second convolutional layer for ADS-B (ModReLU architecture). Each row shows the input signal that maximizes the activation of a particular filter, computed using gradient ascent starting from random noise. Convolutional filters in the first layer span 2 input symbols; filters in the second layer span 6 symbols.

interval (2 symbols). To compute these signals, we start from randomly generated noise and use 200 steps of gradient ascent to maximize the absolute value of each filter output, with the signal normalized to unit power at each step.

4.3 Resilience to Spoofing of ID

In this section, we investigate the potential benefits to using post-preamble portions of the signal and analyze the robustness of our network to the presence of device ID information in such portions. We would like the network to focus on nonlinear transmitter characteristics embedded in the packets rather than the device ID which can be easily spoofed. We expect these nonlinear features to be stable over time, as compared to the device ID which is localized in time. Here we focus on the ADS-B protocol and begin by describing its packet structure.

4.3.1 ADS-B Packet Structure

We consider two different types of ADS-B packets: Mode S and Mode S Extended, depicted in Fig. 4.7. For both packet types, the first 16 symbols consist of a preamble that is identical across devices, while symbols 17-40 contain the ICAO address which is unique to each device. The two modes have different packet lengths, with 64 symbols in Mode S and 120 symbols in Mode S Extended. For this reason, we prune Mode S Extended packets to 64 consecutive symbols, using an offset to determine the first selected symbol. We consider three different scenarios for the offset: an offset of zero, a randomly chosen offset and a fixed offset where we choose the last 64 symbols.

4.3.2 Performance

Performance for each scenario is shown in Fig. 4.8. We report on accuracies for 100 devices, using 400 samples per device for training and testing. We obtain a very high accuracy of 99.29% when we do not use any offset, but this reduces to 65.64% and 75.49% in the scenarios with offsets. The picture becomes clearer when we examine the performance for Mode S and Mode S Extended: the two packet types have identical accuracies in the scenario without offset, but in the other scenarios, Mode S dominates performance. Such a temporal dependence indicates that the network is not learning the true nonlinearities, but rather focusing on device IDs from the payload for Mode S. It is easy to obtain 99% accuracy by restricting attention to just the ICAO address (which can be easily spoofed), which is a clear indicator of "cheating".

A natural approach to prevent such involuntary cheating might be to delete symbols 17-40 (which correspond to the ICAO address). However the presence of parity bits towards the end of the packet makes this approach insufficient. One can observe that a combination of parity and preamble sections can potentially reconstruct the ICAO address, and indeed in practice we obtain artificially high accuracies similar to prior scenarios. In contrast, when we restrict

attention to the preamble alone, accuracy decreases to 81.66%, which is still much better than pure chance. Another approach might be to set the kernel size of the first convolutional layer to 2 symbols, so as to prohibit the network from learning the ICAO address even if we allow access to the entire packet. This reduces accuracy to 97.28%, but it is still much higher than when we use only the preamble. At first glance it may seem like small filter sizes at the first layer are sufficient to prevent cheating, but one just needs to look at the second layer to see that its filters actually extend over 6 symbols.

These experiments show that allowing networks to access ID information is unwise: networks "cheat" whenever given the chance and ignore transmitter-characteristic nonlinearities in favor of localized device information. We can mitigate this by allowing access to only the preamble, in which case we obtain the nonlinear fingerprints we are looking for.

4.4 Impact of Noise

This section studies the effect of noise on classification accuracy. We first study the impact of real-world noise and then discuss noise augmentation strategies to enhance performance.

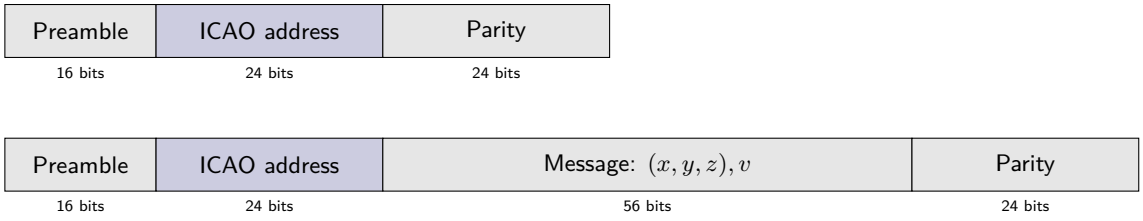


Figure 4.7: Packet structure of ADS-B signals. Top: Mode S; bottom: Mode S Extended. The first 16 symbols of both packet types are device-independent, while the next 24 symbols are highly device-dependent.

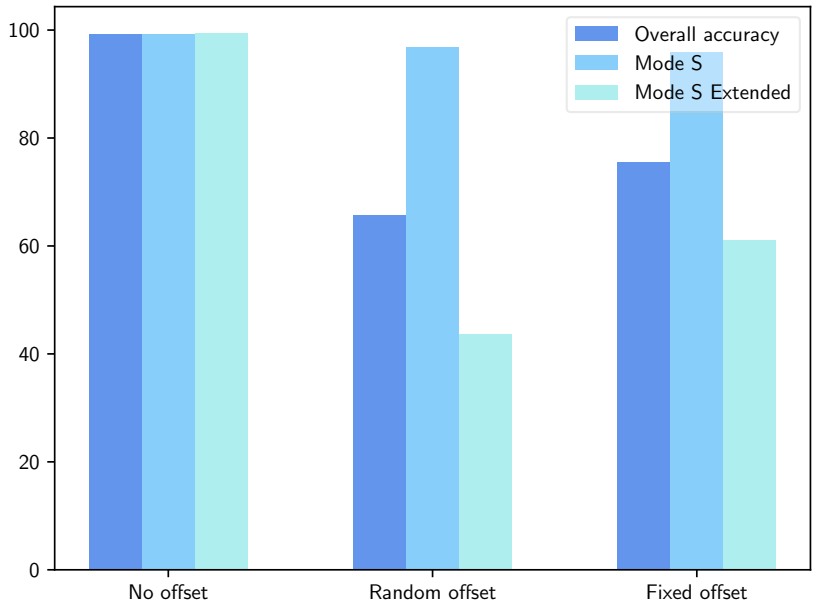


Figure 4.8: Classification accuracies for ADS-B (100 devices) when using post-preamble data. Here we use architecture $100\,C\,100{\times}50\,-\,|\cdot|^2\,-\,100\,C\,10{\times}2\,-\,\text{Avg}\,-\,100\,D$. For details on notation, see Section 4.2.2.

4.4.1 Impact of naturally occurring noise

We study the effect of different levels of noise in the training and test sets, using ADS-B data with 100 devices in each scenario. When we cheat by using the ICAO address as described in the previous section, we obtain artificially high accuracies that are independent of the noise level, indicating that such a network can be easily spoofed even when the data is noisy.

When we use only the preamble, we observe a surprising trend (shown in Table 4.2): performance improves when the training data is *noisier* than the test data. In contrast, when the training SNR is higher than the test SNR, we obtain high training accuracies but low test accuracies. While this result might seem initially counter-intuitive, it is a reasonable hypothesis that noise forces the network to learn features that are more robust to perturbations.

4.4.2 Noise augmentation

We perform noise augmentation by inserting various levels of additional white Gaussian noise (AWGN) in the training and test sets, and report on accuracies as a function of injected noise levels in Table 4.3. Here SNR_{aug} denotes the signal to artificially injected noise ratio, so that $\text{SNR}_{\text{aug}} = \infty$ corresponds to no noise injection. We consider two datasets: 100 ADS-B devices corresponding to the first row of Table 4.2, with 400 signals per device for training and testing; and 100 WiFi devices in an outdoor environment, with 800 signals per device for training and 200 for testing.

Noise insertion yields significant performance benefits, with 19.83% improvement for ADS-B and 7.64% improvement for outdoor WiFi. We note, however, that it is important to add noise to both the training and test sets. Adding noise to only the training set can result in poor performance: for the WiFi data, at 20 dB train SNR_{aug}, test accuracy drops to 2.09% (with 76.32% training accuracy).

Table 4.2: Accuracy as a function of SNR for ADS-B (100 devices), using only the preamble. Here low SNR corresponds to <2 dB, medium SNR to 2-5 dB and high SNR to >5 dB.

Test SNR	Train SNR	Test accuracy	Train accuracy
Low	High	32.29	90.50
	Medium	51.26	84.83
Medium	High	68.85	90.80
	Low	71.13	80.51
High	Medium	81.66	83.71
	Low	73.48	80.53

Table 4.3: Effect of noise augmentation on ADS-B and outdoor WiFi fingerprinting. The ADS-B dataset corresponds to the first row of Table 4.2 (with low test SNR, high train SNR). Noise injection improves ADS-B performance from 32.29% (which corresponds to train SNR_{aug} = test $\text{SNR}_{\text{aug}} = \infty$, i.e. no noise insertion) to 52.12% when train SNR_{aug} = 10 dB and test SNR_{aug} = 50 dB. Outdoor Wifi accuracy improves from 61.73% to 69.37%.

(a) ADS-B (100 devices)

Train SNR_{aug} \ Test SNR_{aug}	20 dB	50 dB	100 dB	∞
10 dB	48.39	52.12	51.89	43.28
15 dB	52.12	50.75	51.98	40.63
20 dB	35.25	47.63	45.44	15.29
25 dB	36.38	45.74	45.29	11.82
∞	25.67	33.55	34.77	32.29

(b) WiFi (100 devices, outdoor environment)

Train SNR_{aug} \ Test SNR_{aug}	20 dB	50 dB	100 dB	∞
10 dB	61.65	62.21	61.90	3.04
15 dB	63.37	62.92	61.00	2.97
20 dB	69.37	69.06	67.83	2.09
25 dB	68.53	69.02	68.17	2.87
∞	29.90	31.45	30.93	61.73

Chapter 5

Conclusions and Future Work

5.1 Sparsifying Front Ends

Our results make the case that sparsity is a crucial tool for limiting the impact of adversarial attacks on neural networks. We have also shown that a "locally linear" model for the network, an implicit premise behind state-of-the-art iterative attacks, provides key design insights, both for devising and combating adversarial perturbations. Our proposed sparsifying front end makes an implicit assumption on the generative model for the data that we believe must hold quite generally for high-dimensional data, in order to evade the curse of dimensionality. We believe that these results are the first steps towards establishing a comprehensive design framework, firmly grounded in theoretical fundamentals, for robust neural networks, that is complementary to alternative defenses based on modifying the manner in which networks are trained.

While many state of the art defenses are based on modifying the optimization involved in training the overall neural network, ours is a bottom-up approach to robustness which is potentially more amenable to interpretation, and to theoretical guarantees based on a statistical characterization of the input. However, much further work is required in order to realize this potential. First, developing sparse generative models matched to various datasets is required

for design of sparsifying front ends. Even for the simple MNIST dataset considered here, the orthogonal wavelet basis considered here is only a first guess, and we believe that it can be improved upon by learning from data, and by use of overcomplete bases. Second, our placeholder scheme of picking the largest K coefficients could be improved by devising computationally efficient and data-adaptive techniques for enforcing sparsity. Lastly, while our work highlights the role of sparse projections in attenuating perturbations, our theoretical framework applies to a high SNR regime corresponding to relatively small attack budgets. In practice, the accuracy with our defense deteriorates for large attack budgets. Thus, sparse projections alone are not enough to provide robustness against large adversarial perturbations, and additional ideas are needed to construct a bottom-up approach that is competitive with the current state of the art defense based on adversarial training of the entire network.

While we have restricted attention to simple datasets and small networks in this work in order to develop insight, the design of larger (deeper) networks for more complex datasets is our ultimate objective. Our preliminary results for CIFAR-10 with a 32-layer ResNet are encouraging, showing that some level of sparsification, along with adversarial training, yields slightly better accuracy under attack than adversarial training alone. However, detailed insight into the front end and network *structure* required to attain robustness remains a wide open research problem.

5.2 Compressive Front Ends

We show that compressive projections are robust to real-world hardware nonlinearities via our case study of an LAE-based image acquisition system. Our synthetic model, together with guidance from recent theory on block diagonal compressive matrices [21], enables us to explore design tradeoffs. Specifically, spatially localized compressive projections, which are easier to implement, can be highly effective, as long as the compressive matrices are appropriately

designed (e.g., independent across rows of an image).

5.3 Robust RF Signatures

In this work, we have demonstrated the efficacy of complex-valued CNNs for wireless fingerprinting. This technique does not rely on signal domain knowledge and, as illustrated by our experiments with WiFi and ADS-B data, can be used across diverse wireless protocols. We show the vulnerability of the approach to "cheating" using transmitter ID when using the entire message to extract the fingerprint. When using the preamble alone, reasonably high accuracies are obtained, and performance is significantly enhanced by noise augmentation. Open issues worth investigating include (a) provably non-cheating, protocol-agnostic strategies that use the entire packet, (b) automatic extraction of the preamble given data corresponding to any protocol, (c) utilizing multiple packets for decisionmaking; and (d) developing detailed insight into the nature of the signatures extracted, and the impact of noise augmentation.

Appendices

Appendix A

Additional Empirical Results

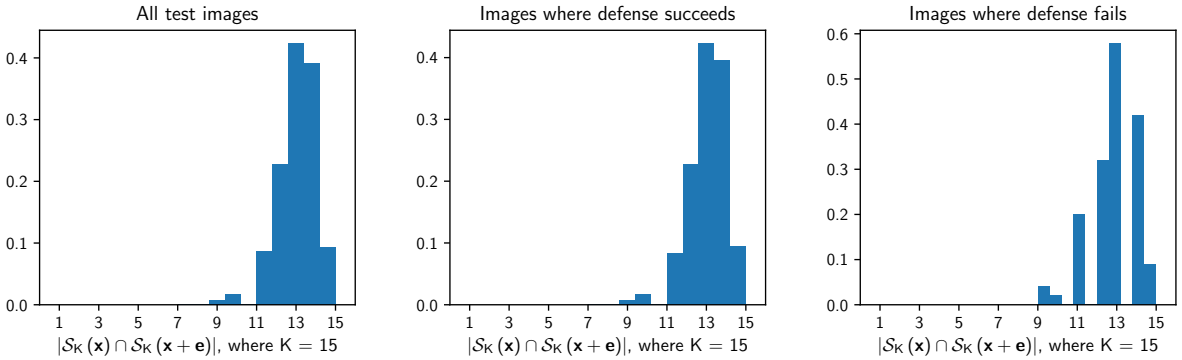
SNR of sparsifying front end

Figure A.1 reports on the overlap in the K-dimensional supports of $\boldsymbol{x}$ and $\boldsymbol{x}+\boldsymbol{e}$ for attacks on linear SVM and CNN. We can observe that the SNR condition in Section 2.3.3 is approximately satisfied for most of the images.

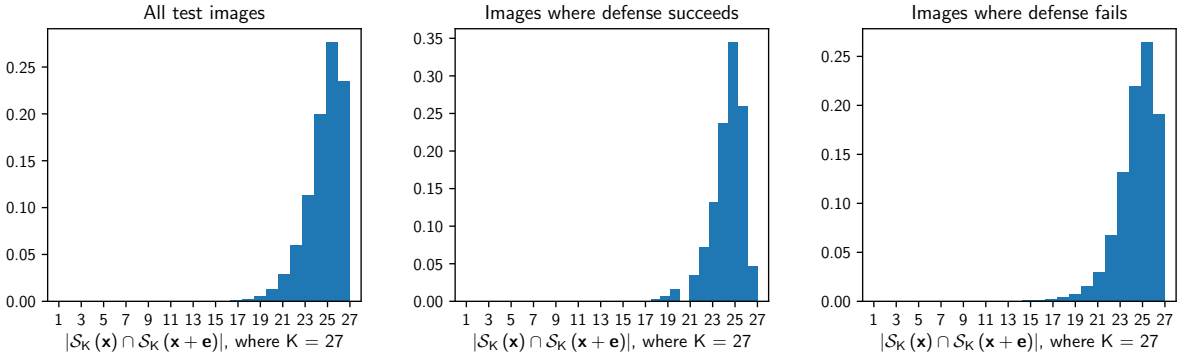
Figure A.4 depicts perturbed images with various support overlaps for the SVM, showing an example in each scenario where the defense succeeds and another where the defense fails. For the fraction of images with low SNR, the adversary can cause significant image distortion by shifting the support of the K selected basis functions, but however such distortions do not necessarily lead to misclassification. As we can observe from the histograms, the distribution of SNRs for images where the defense fails is almost identical to those for which it succeeds.

SNR of ReLU units

Figure A.2 reports on the percentage of ReLU units that flip in one iteration of the iterative locally linear attack with $\delta = 0.01$. When the defense is present, the high SNR condition in

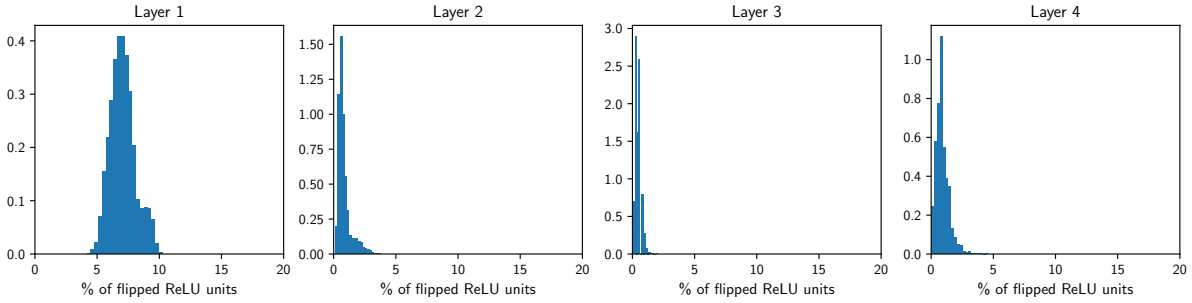


(a) Support overlap for white box attack on linear SVM with $\epsilon = 0.1$ and $K = 15$.

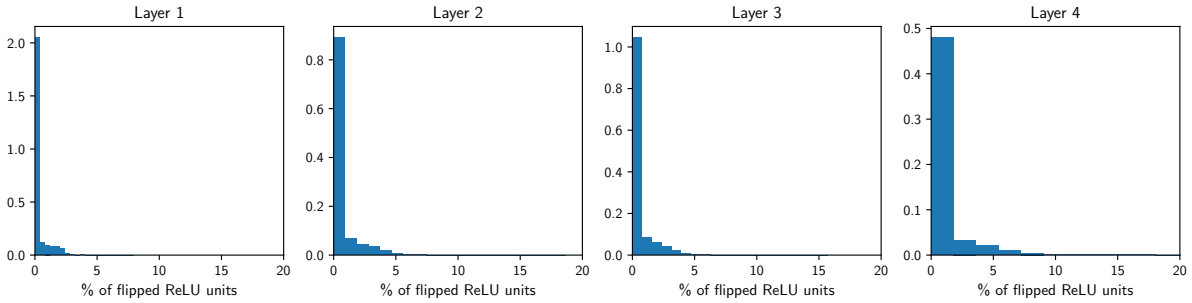


(b) Support overlap for PGD attack (with 100 random restarts) on CNN with $\epsilon = 0.2$ and $K = 27$.

Figure A.1: Histograms of support overlap, i.e. $|\mathcal{S}_{\mathrm{K}}(\boldsymbol{x}) \cap \mathcal{S}_{\mathrm{K}}(\boldsymbol{x}+\boldsymbol{e})|$ for attacks on linear SVM and CNN. Plots are normalized to be probability densities, i.e. the area under each histogram sums to 1.



(a) No defense.



(b) With sparsifying front end ($\rho = 3.5\%$).

Figure A.2: Histograms of the percentage of ReLU units that flip in a single step of the iterative locally linear attack with $\delta = 0.01$, for the 4-layer CNN on MNIST. Plots are normalized to be probability densities.

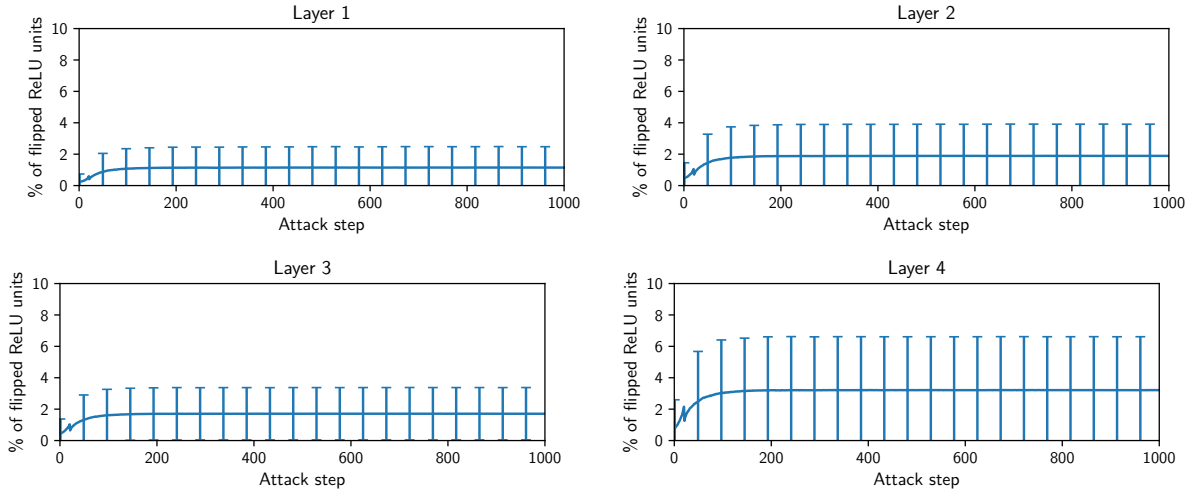


Figure A.3: Plots showing the mean percentage of ReLU units that flip in each attack step, for a 1000-step iterative FGSM attack with $\delta = 0.01$ and $\epsilon = 0.2$ on the 4-layer CNN with defense ($\rho = 3.5\%$). Error bars represent 1 standard deviation from the mean.

Section 2.5.2 is approximately satisfied. Figure A.3 shows the evolution of the SNR condition with attack step for a 1000-step iterative FGSM attack. We can observe that on average, the percentage of ReLUs that flip in each iteration stays relatively small over attack iterations.

However, as the next section shows, it could be better for the adversary to try to make the most of the network's nonlinearity, for example by using iterative attacks with random initializations (PGD), so as to cause a large number of switches to flip to maximize the impact of the second term in Eq. (2.11), Section 2.5.2.

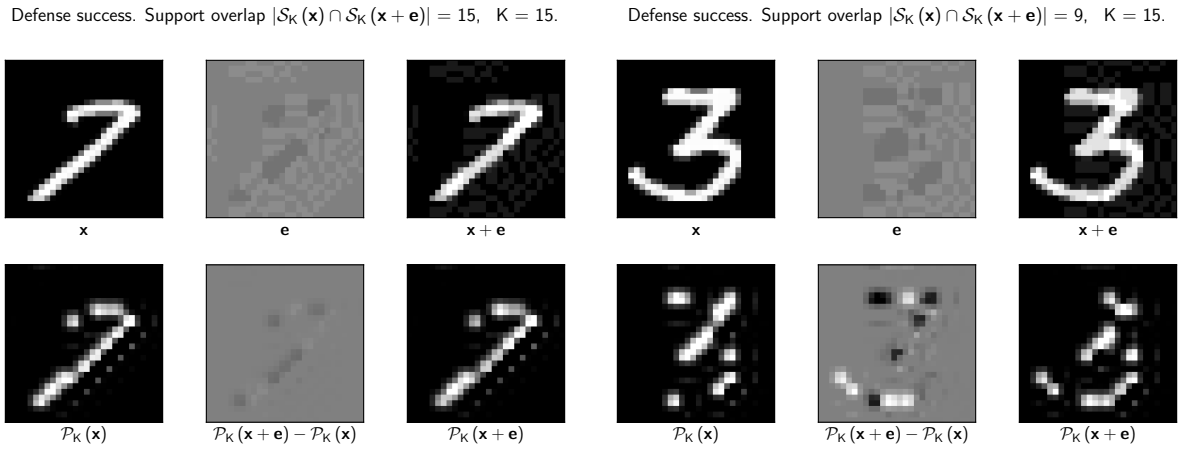
More details on attack performance

Table A.1 reports on attack performance for the 3 versions described in Section 2.6.2: one that uses the backward pass differential approximation (BPDA) technique of [51] to approximate the gradient of the front end as $\mathbf{1}$, a second version where the gradient is calculated as the projection onto the top K basis vectors of the input, and a third version where we iteratively refine the projection. For iterative attacks, we refine the projection for 20 steps within each attack step.

For PGD, we use 100 random restarts and report accuracy over the most successful restart(s) for each image. We report on the effect of changing the per-iteration budget and number of steps for iterative FGSM in Table A.2, with little change in accuracies beyond 100 iterations.

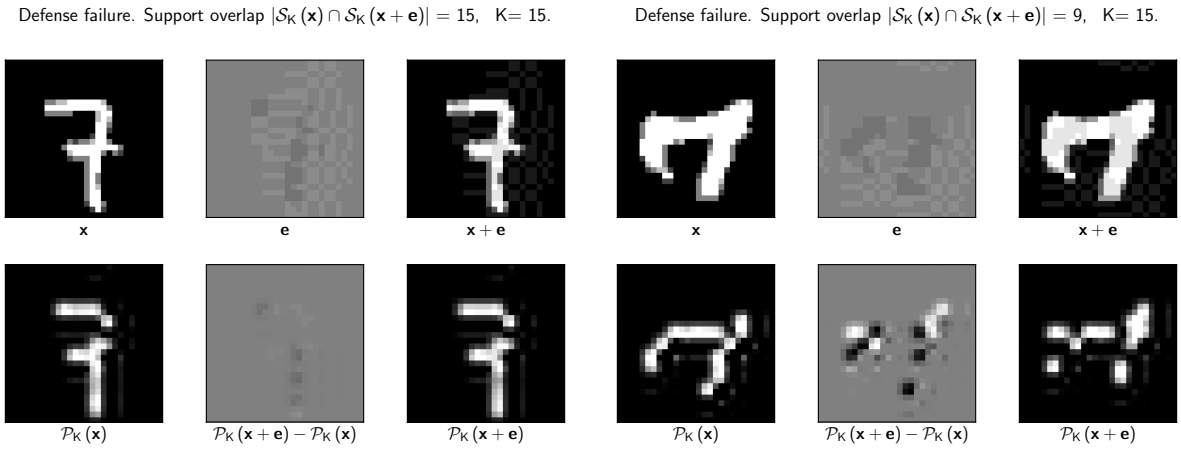
Effect of sparsification on performance without attacks

Sparsification causes a slight performance hit in the accuracy without attacks: for the 4-layer CNN, accuracy reduces from 99.31% to 98.97%. For the 2-layer NN, we report implicitly on this effect in Figure 2.7 (the $\epsilon = 0$ points): the accuracy goes from 99.33% to 99.28%. We note that sparsity has also been suggested purely as a means of improving classification performance



(a) Defense success, high support overlap.

(b) Defense success, low support overlap.



(c) Defense failure, high support overlap.

(d) Defense failure, low support overlap.

Figure A.4: Sample images with low and high support overlap, for white box attack on the linear SVM with $\epsilon = 0.1$ and $\rho = 2\%$. The first row of each subfigure shows the original image, the perturbation and the attacked image, while the second row shows the effect of sparsification. Here $\mathcal{P}_{\mathrm{K}}(\boldsymbol{x}+\boldsymbol{e})$ denotes the projection of $\boldsymbol{x}+\boldsymbol{e}$ onto its own K-dim. support i.e. $\mathcal{S}_{\mathrm{K}}(\boldsymbol{x}+\boldsymbol{e})$.

Table A.1: Multiclass classification accuracies for 4-layer CNN on MNIST, with $\epsilon = 0.2$ for attacks and $\rho = 3.5\%$ for defense. Iterative attacks were run for 1000 steps of $\delta = 0.01$, except for the attacks marked * which use 100 steps of $\delta = 0.05$. Projections were iterated for 20 steps within each attack step.

	BPDA of $\mathbf{1}$	Projections	Iterated projections
Locally linear attack	82.02	86.08	78.27
FGSM	85.35	88.18	85.84
Iter. locally linear attack	76.00	77.27	74.38*
Iter. FGSM	74.97	79.88	75.33
Momentum iter. FGSM	74.79	73.55	—
PGD (100 restarts)	64.62*	65.48*	61.04*

Table A.2: Iterative FGSM accuracies ($\epsilon = 0.2$) as a function of the per-iteration budget δ and number of steps used, for the 4-layer CNN with front end ($\rho = 3.5\%$), using BPDA of $\mathbf{1}$.

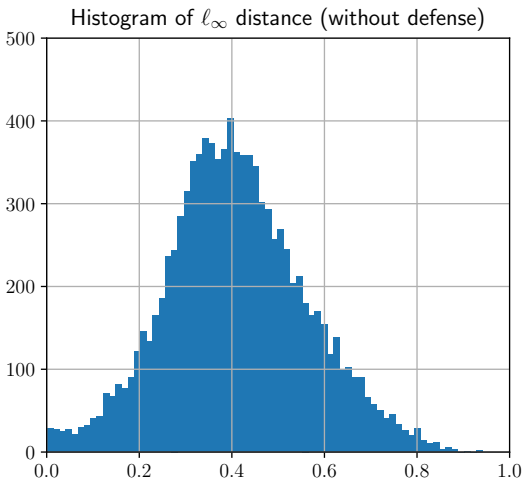
	$\delta = 0.01$	$\delta = 0.05$	$\delta = 0.1$
20 steps	80.84	75.52	75.32
100 steps	75.13	75.17	75.02
1000 steps	74.97	75.10	75.02

(e.g., see [94]) and hence believe that with additional design effort, the performance penalty will be minimal.

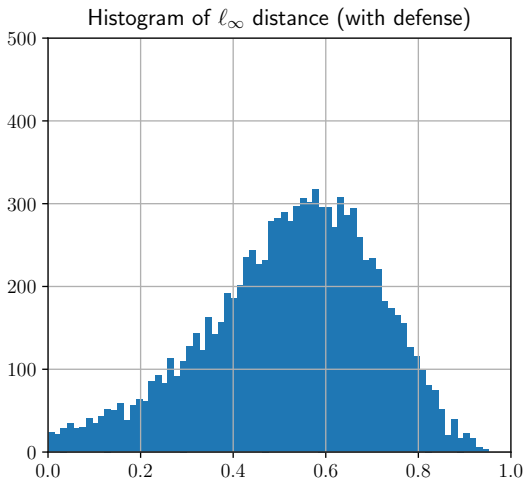
Experiments on the Carlini-Wagner ℓ_2 attack

Since the Carlini-Wagner ℓ_2 attack does not have a fixed bound on the distance between adversarial and true images [46], we report on histograms of distances in Figure A.5 (for the 4-layer CNN). We set the confidence level of the attack to 0, which corresponds to the smallest possible ℓ_2 distance in the C&W attack formulation. The final classification accuracy is 0.92%, but 94.18% of the perturbed images lie outside the ℓ_∞ budget of interest ($\epsilon = 0.2$). The attack is successful in terms of causing misclassification, but since it is an ℓ_2 attack, it fails to produce perturbations conforming to the ℓ_∞ budget. We generate the attack using the CleverHans library (v2.1.0), with default values of attack hyperparameters (listed below) and a backward pass differential approximation (BPDA) [51] of **1** for the gradient of the front end.

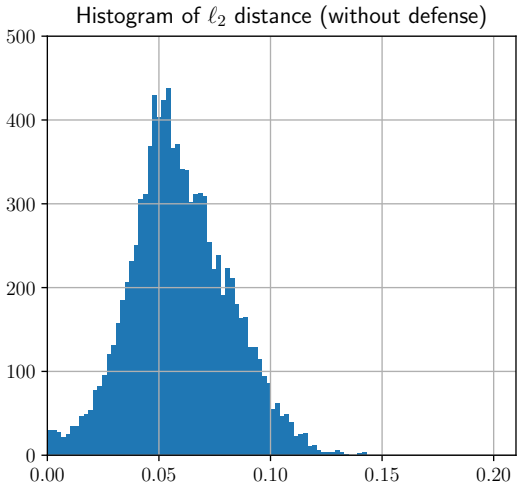
```
learning_rate = 5e-3, binary_search_steps = 5, max_iterations = 1000,
    initial_const = 1e-2.
```



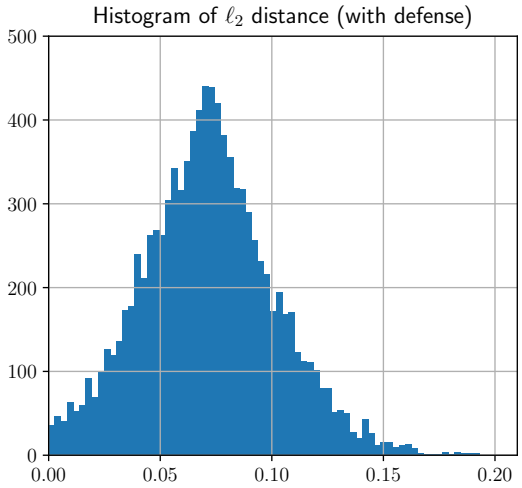
(a) ℓ_∞ distance, with no defense.



(b) ℓ_∞ distance, with front end ($\rho = 3.5\%$).



(c) ℓ_2 distance, with no defense.



(d) ℓ_2 distance, with front end ($\rho = 3.5\%$).

Figure A.5: Histograms of adversarial examples generated by the C&W ℓ_2 attack on the 4-layer CNN.

Appendix B

Coherence of the 2D Fourier Basis

Consider an image $X \in \mathbb{R}^{L \times L}$. If W is the 1D-DFT matrix of size $L \times L$, then the 2D-DFT of X is

$$\mathcal{F}_2(X) = WXW.$$

Vectorizing, we get

$$\text{vec}(\mathcal{F}_2(X)) = \text{vec}(WXW) = (W \otimes W)\text{vec}(X),$$

using the property $\text{vec}(ABC) = (C^T \otimes A)\text{vec}(B)$, where $\otimes$ is the Kronecker product. Hence the 2D-DFT matrix of size $L^2 \times L^2$ is:

$$W_2 = W \otimes W. \tag{B.1}$$

Now we compute the coherence and block-coherence as defined in [21]. Assume that W_2 is of size $N \times N$ and there are J blocks in the compressive matrix. The coherence of W_2 is defined

as the similarity between W_2 and the canonical basis:

$$\begin{aligned}\mu(W_2) &= \sqrt{N}\max_{i,j}|\langle W_2(i), e_j\rangle| \\ &= \sqrt{N}\max_{i,j}|W_2(i,j)| \\ &= 1,\end{aligned} \tag{B.2}$$

where e_j denotes the jth column of the canonical basis for $\mathbb{C}^N$.

The block-coherence $\gamma(W_2)$ is defined as

$$\gamma(W_2) = \sqrt{J}\max_{1\leq n\leq N}\|[U_1\, e_n \;\; U_2\, e_n \;\; \ldots \;\; U_J\, e_n]\|,$$

where $W_2 = [U_1^T, ...U_J^T]^T$ and $U_j \in \mathbb{C}^{\frac{N}{J}\times N}$. The second term is essentially the maximal spectral norm when any column of W_2 is reshaped into an $(N/J)\times J$ matrix. Now the entries of the first column of W_2 all equal $1/\sqrt{N}$, so when reshaped into matrix form its spectral norm is 1. Hence the block-coherence of W_2 is

$$\gamma(W_2) = \sqrt{J}. \tag{B.3}$$

Bibliography

[1] S. Gopalakrishnan, M. Cekic, and U. Madhow, "Robust wireless fingerprinting via complex-valued neural networks," in *IEEE Global Communications Conference (Globecom)*, Waikoloa, Hawaii, December 2019.

[2] Z. Marzi, S. Gopalakrishnan, U. Madhow, and R. Pedarsani, "Sparsity-based defense against adversarial attacks on linear classifiers," in *IEEE International Symposium on Information Theory (ISIT)*, Vail, Colorado, June 2018, pp. 31–35.

[3] S. Gopalakrishnan, Z. Marzi, U. Madhow, and R. Pedarsani, "Combating adversarial attacks using sparse representations," in *International Conference on Learning Representations (ICLR) Workshop Track*, Vancouver, Canada, April 2018.

[4] S. Gopalakrishnan, Z. Marzi, M. Cekic, U. Madhow, and R. Pedarsani, "Robust adversarial learning via sparsifying front ends," *arXiv:1810.10625*.

[5] S. Gopalakrishnan, T. Moy, U. Madhow, and N. Verma, "Compressive information acquisition with hardware impairments and constraints: A case study," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, Louisiana, March 2017, pp. 6075–6079.

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[7] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.

[8] T. Moy, W. Rieutort-Louis, S. Wagner, J. C. Sturm, and N. Verma, "A thin-film, large-area sensing and compression system for image detection," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 11, pp. 1833–1844, 2016.

[9] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," in *International Conference on Learning Representations*, 2018.

[10] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations*, 2014.

[11] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "The robustness of deep networks: A geometrical perspective," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 50–62, 2017.

[12] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[13] C. Qin, J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth, and P. Kohli, "Adversarial robustness through local linearization," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 824–13 833.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[15] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[16] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[17] S. A. Razavi, E. Ollila, and V. Koivunen, "Robust greedy algorithms for compressed sensing," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, 2012, pp. 969–973.

[18] F. Krzakala, M. Mézard, and L. Zdeborová, "Compressed sensing under matrix uncertainty: Optimum thresholds and robust approximate message passing," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 5519–5523.

[19] L. Jacques, K. Degraux, and C. De Vleeschouwer, "Quantized iterative hard thresholding: Bridging 1-bit and high-resolution quantized compressed sensing," *arXiv preprint arXiv:1305.1786*, 2013.

[20] G. Tsagkatakis and P. Tsakalides, "Recovery of quantized compressed sensing measurements," in *SPIE/IS&T Electronic Imaging*. International Society for Optics and Photonics, 2015, pp. 940 106–940 106.

[21] A. Eftekhari, H. L. Yap, C. J. Rozell, and M. B. Wakin, "The restricted isometry property for random block diagonal matrices," *Applied and Computational Harmonic Analysis*, vol. 38, no. 1, pp. 1–31, 2015.

[22] N. R. Shanbhag, R. A. Abdallah, R. Kumar, and D. L. Jones, "Stochastic computation," in *Proceedings of the 47th Design Automation Conference*. ACM, 2010, pp. 859–864.

[23] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, April 2005.

[24] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, 2008, pp. 116–127.

[25] I. O. Kennedy, P. Scanlon, F. J. Mullany, M. M. Buddhikot, K. E. Nolan, and T. W. Rondeau, "Radio transmitter fingerprinting: A steady state frequency domain approach," in *2008 IEEE 68th Vehicular Technology Conference*, 2008, pp. 1–5.

[26] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," *IEEE Transactions on Mobile Computing*, vol. 9, no. 3, pp. 449–462, 2010.

[27] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the reliability of wireless fingerprinting using clock skews," in *Proceedings of the 3rd ACM Conference on Wireless Network Security*, 2010, pp. 169–174.

[28] M. Strohmeier and I. Martinovic, "On passive data link layer fingerprinting of aircraft transponders," in *Proceedings of the First ACM Workshop on Cyber-Physical Systems-Security and/or PrivaCy*, 2015, pp. 1–9.

[29] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "GTID: A technique for physical device and device type fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 519–532, 2015.

[30] M. Leonardi, L. Di Gregorio, and D. Di Fausto, "Air traffic security: Aircraft classification using ADS-B messages phase-pattern," *Aerospace*, vol. 4, no. 4, p. 51, 2017.

[31] K. Merchant, S. Revay, G. Stantchev, and B. Nousain, "Deep learning for RF device fingerprinting in cognitive communication networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 160–167, 2018.

[32] J. Hua, H. Sun, Z. Shen, Z. Qian, and S. Zhong, "Accurate and efficient wireless device fingerprinting using channel state information," in *IEEE International Conference on Computer Communications*, 2018, pp. 1700–1708.

[33] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *International Conference on Engineering Applications of Neural Networks*, 2016, pp. 213–226.

[34] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized Radio clAssification through Convolutional neuraL nEtworks," in *IEEE International Conference on Computer Communications*, 2019.

[35] A. Hirose and S. Yoshida, "Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 541–551, 2012.

[36] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.

[37] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto, "Classification regions of deep neural networks," *arXiv preprint arXiv:1705.09552*, 2017.

[38] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli, "Exponential expressivity in deep neural networks through transient chaos," in *Advances in Neural Information Processing Systems*, 2016, pp. 3360–3368.

[39] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 1999.

[40] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016.

[41] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against deep learning systems using adversarial examples," in *ACM Asia Conference on Computer and Communications Security*, 2017, pp. 506–519.

[42] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "The space of transferable adversarial examples," *arXiv preprint arXiv:1704.03453*, 2017.

[43] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," in *International Conference on Learning Representations*, 2017.

[44] ——, "Adversarial examples in the physical world," in *ICLR Workshop*, 2017.

[45] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 86–94.

[46] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy*, 2017, pp. 39–57.

[47] ——, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 3–14.

[48] ——, "MagNet and "Efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[49] A. Kurakin, I. Goodfellow, S. Bengio, Y. Dong, F. Liao, M. Liang, T. Pang, J. Zhu, X. Hu, C. Xie *et al.*, "Adversarial attacks and defences competition," in *The NIPS'17 Competition: Building Intelligent Systems*. Springer, 2018, pp. 195–231.

[50] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9185–9193.

[51] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *International Conference on Machine Learning*, 2018, pp. 274–283.

[52] A. Fawzi, O. Fawzi, and P. Frossard, "Analysis of classifiers' robustness to adversarial perturbations," *Machine Learning*, 2017.

[53] J. Peck, Y. Saeys, B. Goossens, and J. Roels, "Lower bounds on the robustness to adversarial perturbations," in *Advances in Neural Information Processing Systems*, 2017, pp. 804–813.

[54] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *NIPS Workshop on Machine Deception*, 2017.

[55] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1778–1787.

[56] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *International Conference on Learning Representations*, 2018.

[57] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau, "Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression," *arXiv preprint arXiv:1705.02900*, 2017.

[58] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," in *International Conference on Learning Representations*, 2018.

[59] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal, "Enhancing robustness of machine learning systems via data transformations," in *52nd Annual Conference on Information Sciences and Systems (CISS)*, 2018, pp. 1–5.

[60] A. Ilyas, A. Jalal, E. Asteri, C. Daskalakis, and A. G. Dimakis, "The robust manifold defense: Adversarial training using generative models," *arXiv preprint arXiv:1712.09196*, 2017.

[61] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-GAN: Protecting classifiers against adversarial attacks using generative models," in *International Conference on Learning Representations*, 2018.

[62] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," in *International Conference on Learning Representations*, 2018.

[63] ——, "Semidefinite relaxations for certifying robustness to adversarial examples," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 900–10 910.

[64] E. Wong and Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," in *International Conference on Machine Learning*, 2018, pp. 5283–5292.

[65] E. Wong, F. Schmidt, J. H. Metzen, and J. Z. Kolter, "Scaling provable adversarial defenses," in *Advances in Neural Information Processing Systems*, 2018, pp. 8410–8419.

[66] A. Sinha, H. Namkoong, and J. Duchi, "Certifying some distributional robustness with principled adversarial training," in *International Conference on Learning Representations*, 2018.

[67] M. Mirman, T. Gehr, and M. Vechev, "Differentiable abstract interpretation for provably robust neural networks," in *International Conference on Machine Learning*, 2018, pp. 3575–3583.

[68] M. Hein and M. Andriushchenko, "Formal guarantees on the robustness of a classifier against adversarial manipulation," in *Advances in Neural Information Processing Systems*, 2017, pp. 2266–2276.

[69] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, "Parseval networks: Improving robustness to adversarial examples," in *International Conference on Machine Learning*, 2017, pp. 854–863.

[70] Z. Marzi, S. Gopalakrishnan, U. Madhow, and R. Pedarsani, "Sparsity-based defense against adversarial attacks on linear classifiers," in *IEEE International Symposium on Information Theory (ISIT)*, 2018, pp. 31–35.

[71] M. Bafna, J. Murtagh, and N. Vyas, "Thwarting adversarial examples: An L_0-robust sparse fourier transform," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 096–10 106.

[72] A. Fawzi, H. Fawzi, and O. Fawzi, "Adversarial vulnerability for any classifier," in *Advances in Neural Information Processing Systems*, 2018, pp. 1178–1187.

[73] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 97–117.

[74] R. Ehlers, "Formal verification of piece-wise linear feed-forward neural networks," in *International Symposium on Automated Technology for Verification and Analysis*. Springer, 2017, pp. 269–286.

[75] C.-H. Cheng, G. Nührenberg, and H. Ruess, "Maximum resilience of artificial neural networks," in *International Symposium on Automated Technology for Verification and Analysis*. Springer, 2017, pp. 251–268.

[76] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output range analysis for deep feedforward neural networks," in *NASA Formal Methods Symposium*. Springer, 2018, pp. 121–138.

[77] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[78] T. Blumensath and M. E. Davies, "Gradient pursuits," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2370–2382, 2008.

[79] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM Review*, vol. 51, no. 1, pp. 34–81, 2009.

[80] D. L. Donoho, I. M. Johnstone *et al.*, "Ideal denoising in an orthonormal basis chosen from a library of bases," *Comptes rendus de l'Académie des sciences. Série I, Mathématique*, vol. 319, no. 12, pp. 1317–1322, 1994.

[81] M. Aharon, M. Elad, and A. Bruckstein, "k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[82] I. F. Gorodnitsky and B. D. Rao, "Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm," *IEEE Transactions on Signal Processing*, vol. 45, no. 3, pp. 600–616, 1997.

[83] S. J. Wright, R. D. Nowak, and M. A. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[84] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[85] M. Elad, J.-L. Starck, P. Querre, and D. L. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)," *Applied and Computational Harmonic Analysis*, vol. 19, no. 3, pp. 340–358, 2005.

[86] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition in a signal dictionary," *Neural Computation*, vol. 13, no. 4, pp. 863–882, 2001.

[87] D. Malioutov, M. Cetin, and A. S. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Transactions on Signal Processing*, vol. 53, no. 8, pp. 3010–3022, 2005.

[88] S. Mallat, *A Wavelet Tour of Signal Processing*. Elsevier, 1999.

[89] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2121–2130, 2010.

[90] R. Rubinstein, T. Peleg, and M. Elad, "Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model," *IEEE Transactions on Signal Processing*, vol. 61, no. 3, pp. 661–677, 2013.

[91] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, p. 607, 1996.

[92] P. Földiak, "Forming sparse representations by local anti-hebbian learning," *Biological cybernetics*, vol. 64, no. 2, pp. 165–170, 1990.

[93] P. Agrawal, R. Girshick, and J. Malik, "Analyzing the performance of multilayer neural networks for object recognition," in *European conference on computer vision*. Springer, 2014, pp. 329–344.

[94] A. Makhzani and B. Frey, "k-sparse autoencoders," in *International Conference on Learning Representations*, 2014.

[95] I. Daubechies, *Ten Lectures on Wavelets*. Siam, 1992, vol. 61.

[96] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[97] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[98] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[99] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[100] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[101] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on information theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[102] M. A. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal on Selected Topics in Signal Processing*, vol. 1, pp. 586–597, 2003.

[103] K. R. Lakshmikumar, R. A. Hadaway, and M. A. Copeland, "Characterisation and modeling of mismatch in MOS transistors for precision analog design," *IEEE Journal of Solid-State Circuits*, vol. 21, no. 6, pp. 1057–1066, December 1986.

[104] A. A. M. Saleh, "Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers," *IEEE Transactions on Communications*, vol. 29, no. 11, pp. 1715–1720, November 1981.

[105] A. Zhu and T. J. Brazil, "Behavioral modeling of RF power amplifiers based on pruned volterra series," *IEEE Microwave and Wireless Components Letters*, vol. 14, no. 12, pp. 563–565, December 2004.

[106] Hyunchul Ku and J. S. Kenney, "Behavioral modeling of nonlinear RF power amplifiers considering memory effects," *IEEE Transactions on Microwave Theory and Techniques*, vol. 51, no. 12, pp. 2495–2504, December 2003.

[107] J. C. Pedro and S. A. Maas, "A comparative overview of microwave and wireless power-amplifier behavioral modeling approaches," *IEEE Transactions on Microwave Theory and Techniques*, vol. 53, no. 4, pp. 1150–1163, April 2005.

[108] E. Costa, M. Midrio, and S. Pupolin, "Impact of amplifier nonlinearities on OFDM transmission system performance," *IEEE Communications Letters*, vol. 3, no. 2, pp. 37–39, February 1999.

[109] S. Merchan, A. G. Armada, and J. L. Garcia, "OFDM performance in amplifier nonlinearity," *IEEE Transactions on Broadcasting*, vol. 44, no. 1, pp. 106–114, March 1998.

[110] S. S. Hanna and D. Cabric, "Deep learning based transmitter identification using power amplifier nonlinearity," in *International Conference on Computing, Networking and Communications (ICNC)*, February 2019, pp. 674–680.

[111] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, 2005.

[112] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[113] S. Wisdom, T. Powers, J. Hershey, J. Le Roux, and L. Atlas, "Full-capacity unitary recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 4880–4888.