UC Santa Barbara

UC Santa Barbara Electronic Theses and Dissertations

Title

Robust Learning Techniques for Deep Neural Networks

Permalink

https://escholarship.org/uc/item/0127x8p2

Author

Cekic, Metehan

Publication Date

2022

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

University of California
Santa Barbara

Robust Learning Techniques for Deep Neural Networks

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

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Metehan Cekic

Committee in charge:

Professor Upamanyu Madhow, Chair
Professor Kenneth Rose
Professor Ramtin Pedarsani
Professor Zheng Zhang

September 2022

The Dissertation of Metehan Cekic is approved.

Professor Kenneth Rose

Professor Ramtin Pedarsani

Professor Zheng Zhang

Professor Upamanyu Madhow, Committee Chair

August 2022

Robust Learning Techniques for Deep Neural Networks

Copyright © 2022

by

Metehan Cekic

Acknowledgements

I want to start by expressing my gratitude to my advisor, Professor Upamanyu Madhow, whose knowledge and experience were crucial in developing this thesis. I feel very fortunate to have had your insightful guidance and support throughout the course of my Ph.D. I also want to extend thanks to my thesis committee, Prof. Kenneth Rose, Prof. Ramtin Pedarsani, and Prof. Zheng Zhang for their valuable feedback and suggestions.

Furthermore, I am fortunate to have had the opportunity to collaborate with Can, Ahmet, Soorya, Ruirui, Zeya, Yuguang, and Andreas. I am grateful to them and to my labmates Faruk, Maryam, Zhinus, Anant, Mohammed, Bhagyashree, Lalitha, and Ganesh for many insightful research discussions over the years.

In addition, I would like to thank my parents, Mehmet and Dursun, and my brother Melihsah for their unending support and sympathetic ear. You are always there for me. Finally, I need to thank my friends who supported me throughout this time: Arda, Burak, Furkan, Berkay, Asutay, Ilayda, Arinc, Utku, Emre, Mert, Zeki, and Tarik. I appreciate their help, company, and friendship.

Lastly, I thank my partner, Seyda, for her love, support, and encouragement during the final stage of my Ph.D.

Curriculum Vitæ

Metehan Cekic

Education

2022	Ph.D. in Electrical and Computer Engineering (Expected), University of California, Santa Barbara.
2019	M.S. in Electrical and Computer Engineering, University of California, Santa Barbara.
2017	B.S. in Electrical & Electronics Engineering, Bogazici University, Istanbul, Turkey.
2017	B.S. in Physics, Bogazici University, Istanbul, Turkey.

Publications

1. **M. Cekic**, C. Bakiskan, U. Madhow, "Neuro-Inspired Deep Neural Networks with Sparse, Strong Activations", to appear in *IEEE International Conference in Image Processing (ICIP)*, Bordeaux, France, Oct 2022.
2. C. Bakiskan, **M. Cekic**, U. Madhow, "Early Layers Are More Important For Adversarial Robustness", *International Conference on Machine Learning (ICML) 2022 Workshop – New Frontiers in Adversarial Machine Learning*, Baltimore, USA, Jul 2022.
3. **M. Cekic**, C. Bakiskan, U. Madhow, "Layerwise Hebbian/anti-Hebbian (HaH) Learning In Deep Networks: A Neuro-inspired Approach To Robustness", *International Conference on Machine Learning (ICML) 2022 Workshop – New Frontiers in Adversarial Machine Learning*, Baltimore, USA, Jul 2022.
4. **M. Cekic**, C. Bakiskan, U. Madhow, "Towards Robust, Interpretable Neural Networks via Hebbian/anti-Hebbian Learning: A Software Framework for Training with Feature-Based Costs", *Software Impacts* (2022).
5. **M. Cekic**, R. Li, Z. Chen, Y. Yang, A. Stolcke, U. Madhow, "Self-Supervised Speaker Recognition Training Using Human-Machine Dialogues", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, May 2022.
6. **M. Cekic**, S. Gopalakrishnan, U. Madhow, "Wireless Fingerprinting via Deep Learning: The Impact of Confounding Factors", *IEEE Asilomar Conference on Signals, Systems, and Computers*, Nov. 2021.
7. C. Bakiskan, **M. Cekic**, A. D. Sezer, U. Madhow, "Sparse Coding Frontend For Robust Neural Networks", *International Conference on Learning Representations (ICLR), Workshop on Security and Safety in Machine Learning Systems*, May 2021.

8. C. Bakiskan, **M. Cekic**, A. D. Sezer, U. Madhow, "A Neuro-Inspired Autoencoding Defense Against Adversarial Perturbations", *IEEE International Conference on Image Processing (ICIP)*, Anchorage, Sept. 2021.
9. S. Gopalakrishnan, Z. Marzi, **M. Cekic**, U. Madhow, R. Pedarsani, "Robust Adversarial Learning via Sparsifying Front Ends", Preprint, arXiv:1810.10625.
10. C. Bakiskan, S. Gopalakrishnan, **M. Cekic**, U. Madhow, R. Pedarsani, "Polarizing Front Ends For Robust CNNs", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020.
11. S. Gopalakrishnan, **M. Cekic**, U. Madhow, "Robust Wireless Fingerprinting via Complex-Valued Neural Networks", *IEEE Global Communications Conference (Globecom)*, Hawaii, Dec. 2019.

Abstract

Robust Learning Techniques for Deep Neural Networks

by

Metehan Cekic

Deep Neural Networks (DNNs) yield state-of-the-art performance in an increasing array of applications. Despite the pervasive impact of DNNs, there remain significant concerns regarding their (lack of) stability and robustness. In this thesis, we explore several complementary approaches for guiding DNNs to learn robust and stable features, including domain expertise, domain-specific measures, and neuro-inspired modifications. We present novel augmentation techniques, cost functions, and data rejection methods that supplement conventional DNN training for reliable feature extraction.

We first study the robustness in the presence of strong confounding factors for Radio-frequency (RF) fingerprinting in which the aim is to distinguish devices using subtle hardware imperfections which vary from device to device. However, the features such as carrier frequency offset and wireless channel misguide DNNs. We point out that, unless proactively discouraged from doing so, DNNs learn these strong confounding features rather than the nonlinear device-specific characteristics that we seek to learn. We investigate and evaluate strategies based on augmentation and estimation to promote generalization across realizations of these confounding factors using WiFi data.

In our second study, we present robustness measures in the context of self-supervised contrastive learning. We investigate how to pretrain speaker recognition models by leveraging dialogues between customers and smart-speaker devices. However, the supervisory information in such dialogues is inherently noisy, as multiple speakers may speak to a device in the course of the same dialogue. To address this issue, we propose an effective

rejection mechanism that selectively learns from dialogues based on their acoustic homogeneity. We also present a novel cost function particularly designed for a corrupted dataset in the contrastive learning setting.

Lastly, we introduce a promising neuro-inspired architectural DNN design and a cost function to learn robust and interpretable features. We develop a software framework in which end-to-end costs can be supplemented with costs which depend on layer-wise activations, permitting more fine-grained control of features. We apply this framework to include Hebbian/anti-Hebbian (HaH) learning in a discriminative setting, demonstrating promising gains in robustness for the CIFAR10 image classification.

This thesis includes the following publications:

- © 2021 IEEE. Reprinted, with permission, from M. Cekic, S. Gopalakrishnan, and U. Madhow, "Wireless Fingerprinting via Deep Learning: The Impact of Confounding Factors", In *IEEE Asilomar Conference on Signals, Systems, and Computers*, Nov. 2021.

- © 2022 IEEE. Reprinted, with permission, from M. Cekic, R. Li, Z. Chen, Y. Yang, A. Stolcke, U. Madhow, "Self-Supervised Speaker Recognition Training Using Human-Machine Dialogues", In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2022.

- © 2022 IEEE. Reprinted, with permission, from M. Cekic, C. Bakiskan, U. Madhow, "Neuro-Inspired Deep Neural Networks with Sparse, Strong Activations", In *IEEE International Conference in Image Processing (ICIP)*, Oct 2022.

- M. Cekic, C. Bakiskan, U. Madhow, "Towards robust, interpretable neural networks via Hebbian/anti-Hebbian learning: A software framework for training with feature-based costs", In *Software Impacts*, 2022 [1].

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Introduction

Deep neural networks have come a long way to attain outstanding performance in a wide variety of fields, including vision [6, 7], game-playing agents [8, 9], natural language processing [10, 11], and speech [12]. Arguably, a key contributor to this explosive growth is the evolution of a powerful yet generic computational infrastructure for training DNNs with a very large number of parameters with variants of stochastic gradient descent on an end-to-end cost function without requiring domain expertise. In other words, abundant data combined with computational infrastructure empowered them to substitute domain expertise with solely data-driven models. On the other hand, the lack of robustness in DNNs are still in question: these networks are vulnerable to noise [4], distribution shifts [13], label corruptions [14, 15], and adversarial perturbations [16, 17]. The network learns the easiest set of features that it can in order to accomplish the desired task on training data: these features, depending on the task, can be unstable over time, easy to manipulate for an adversary, or inaccurate. In this thesis, we show how domain knowledge and particular measurements can help DNNs learn stable and reliable features. To enhance

traditional DNN training for accurate feature extraction, we provide new augmentation techniques, cost functions, and data rejection strategies.

In the first part of this thesis, we focus on robustness in radio frequency (RF) fingerprinting, in which the goal is to classify wireless devices using nonlinear circuit-level variations. While these subtle variations are enough to distinguish between devices, deep neural networks rather use strong spurious correlations such as wireless channel and carrier frequency offset [13], which are not stable over time. We are interested in discouraging neural networks from capturing these spurious correlations instead of utilizing the stable yet subtle nonlinear physical layer features.

Our second study presents novel techniques for dealing with highly noisy data in a self-supervised learning setting for speaker recognition. We aim to train a robust deep neural network to distinguish between different speakers using an unlabelled dialogue sessions dataset. While most of the dialogues consist of a single speaker, the presence of multi-speaker dialogues poses a threat to learning good speaker ID features. Therefore, we present a self-supervised rejection mechanism in order to ignore multi-speaker dialogues and complement it with a novel loss function to deal with noisy data. Lastly, we present a neuro-inspired approach to improve the robustness of deep neural networks to small adversarial perturbations. We demonstrate that our neuro-inspired model is substantially more robust to a range of corruptions than a baseline end-to-end trained model.

1.1 RF Fingerprints

Recent advances in wireless communication systems and the availability of cheaper and more compact computational infrastructure flourished the IoT devices. Nowadays, more IoT devices are readily in use than the human population on Earth. Therefore, it is unsurprising that more and more effort is put into securing these devices and their

networks. In this work, we focus on *RF fingerprints* and their potential use in the security of IoT devices. In short, RF fingerprints are subtle nonlinear features in wireless signals due to the nonlinear variations across circuit components in wireless transmitters. *RF fingerprinting* is interested in using these features as a unique identity for devices.

The most recent body of work demonstrates that end-to-end deep neural networks, including complex-valued convolutional neural networks [18], recurrent neural networks [19], and transformers [20], are very successful at teasing out these features to classify devices. In this work, we inspect the stability of these DNNs; specifically, we evaluate these networks under strong confounding factors such as wireless channel and carrier frequency offset. We demonstrate that these networks have stability issues, especially if they are trained with data collected over a short period of time. Confounding factors such as wireless channel and CFO stays consistent among the packets collected from the same device; therefore, they deliver a spurious correlation that a DNN can exploit to classify devices.

Our key message is that given subtle stable features and strong confounding factors, neural networks fail to choose/extract appropriately. As a result, the blind adoption of these networks may result in disastrous stability issues. We explore using signal modelings to discourage neural networks from learning spurious correlations. We specifically demonstrate the usefulness of augmentation strategies in learning stable RF signatures while highlighting their inherent limits in a small dataset setting.

Our contributions are summarized below.

1.1.1 Contributions

- Using controlled emulations on a clean WiFi dataset, we demonstrate the vulnerability of conventional CNN training to confounding factors such as propagation channels and

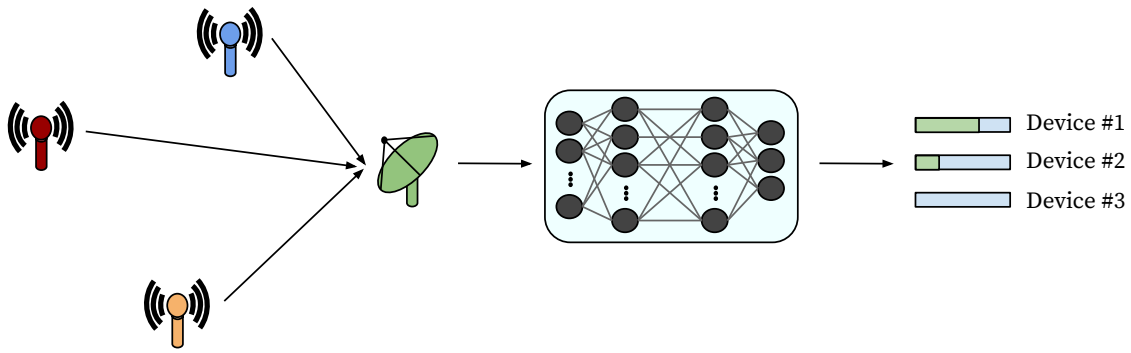


Figure 1.1: Classification of devices using deep neural networks.

frequency offsets, which are far stronger than the nonlinear effects we seek to capture.

- We develop augmentation strategies based on signal models for the impact of confounding factors, and evaluate performance against compensation techniques that explicitly try to undo them. We find that compensation works well if the undesired features are simple enough, like the CFO. However, for more complex effects such as a multipath channel, model-driven augmentation outperforms explicit estimation and compensation for learning robust signatures.

- We make publicly available a simulation-based dataset based on models of some typical circuit-level nonlinearities [21–23]. The results we obtain on this dataset are comparable to those from the measurement-based dataset, enabling reproducibility. The dataset and code are available at [24].

1.2 Robust Self-Supervised Speaker Recognition

Speaker recognition answers the fundamental question "who is speaking" based on a sample of speech, thereby enabling both personalization and authentication in speech-based applications. Most speaker recognition model training is supervised, in the sense

that it relies on both clean and sufficient speaker-labeled data [25–27]. However, labeling data in the quantities required for production-level models is a substantial bottleneck. Recent work [28–34] is turning to unlabeled data for pretraining a speech model first, and then fine-tuning it on a smaller labeled dataset. In this work, we focus on how to pretrain a speech model suitable for speaker recognition tasks.

We propose a contrastive self-supervised method specialized in speaker recognition. To achieve this goal, we leverage unlabeled dialogues between smart-speaker devices and their users. Figure 1.2 shows two dialogue samples, where each dialogue is composed of spoken interactions between customers and a smart speaker. We presume that most dialogues, such as Dialogue 1, are clean in the sense that each involves customer utterances from a single speaker only. Therefore, customer utterances from the same dialogue serve as positive instances, while customer utterances from different dialogues form negative instances. However, a few dialogues, e.g., Dialogue 2, are noisy with respect to speaker identities, i.e., contain customer utterances from more than one speaker. In order to avoid contaminating the model training with positive samples involving different speakers, we develop a rejection module and all-vs-all loss. The rejection module allows the model to effectively learn from clean dialogues and give less weight to the noisy ones, leading to a more accurate and robust speaker recognition model [15]. On the other hand, all-vs-all loss solves the problem of inaccurate centroids resulting from multi-speaker dialogues.

1.2.1 Contributions

Thus, our contribution is a robust self-supervised learning method for speaker recognition systems, demonstrating that

- a dialogue dataset from human-device interactions is an effective unlabeled data source that can be leveraged in self-supervised pretraining of speaker recognition

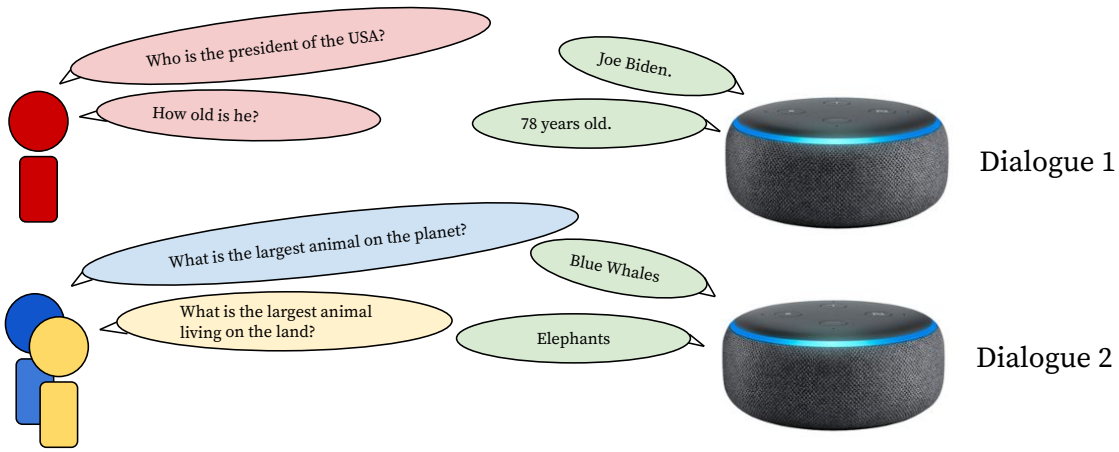


Figure 1.2: Most dialogues contain a single human speaker; however, there are dialogues with more than one human speaker.

models;

- our proposed all-vs-all metric loss improves the performance of the self-supervised training with a highly noisy dataset.
- self-supervised rejection is a very effective tool to deal with false positive pairs caused by multi-speaker dialogues, providing more than 15% equal error rate (EER) improvement even without fine-tuning;
- fine-tuning the pretrained model utilizing our framework can further improve speaker recognition and relative EER improvement is as high as 41.28%.

1.3 Hebbian/Anti-Hebbian Learning

[17] and [16] showed the existence of adversarial examples which can fool the deep neural networks. These adversarial examples are nothing more than natural examples with tiny, well-crafted, and (nearly) imperceptible perturbations that can render state-of-the-art models unusable (see fig. 1.3). Therefore, a plethora of defense measures have

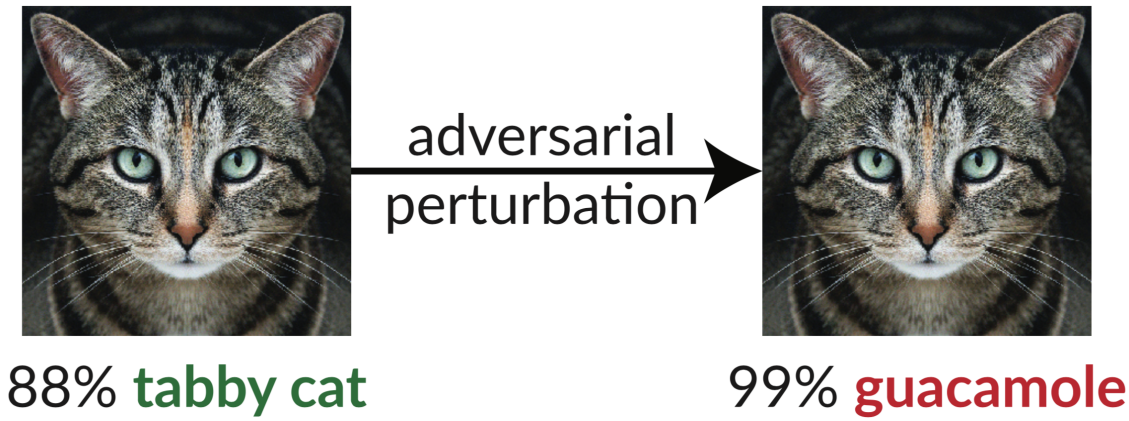


Figure 1.3: Even an imperceivable perturbation can render InceptionV3 [2] classifier into classifying a cat image as "guacamole". Image is taken from the Github repository of [3].

been devised to fight these attacks, only to be defeated by stronger adversaries [3, 35, 36].

The state-of-the-art defense that stands still against adversarial attacks is adversarial training (see [37] and variants thereof), which augments the input data with adversarial perturbations during training, while the cost function in [38] seeks to trade off clean accuracy and attacked accuracy. However, these methods only provide robustness against the adversaries used in the training phase.

In chapter 4, we explore a complementary approach to robustness based on supplementing the end-to-end cost function with layer-wise costs aimed at shaping the features extracted by intermediate layers of the DNN. Specifically, while standard DNNs produce a large fraction of small activations at each layer, we seek architectures which produce a small fraction of strong activations, while continuing to utilize existing network architectures for feedforward inference and existing software infrastructure for stochastic gradient training. To this end, we introduce neuro-inspired mechanisms creating competition between neurons during both training and inference.

1.3.1 Contributions

Specifically, we propose neuro-inspired Hebbian/anti-Hebbian (HaH) learning. Instead of the huge proportion of small activations produced by a typical DNN, we attempt to develop sparse activation patterns with a small fraction of large activations. With HaH costs, neurons which are strongly activated by an input take a step in the direction of the input, while the remaining neurons take a step in the direction opposite to the input. We also introduce changes in the inference framework, replacing a standard DNN layer with a HaH block which includes implicit weight normalization for each neuron, allowing us to interpret its output as a projection, and divisive output normalization, allowing us to suppress weak activations using strong ones. These result in invariance to input and weight scaling.

We demonstrate the effectiveness of our framework against the state-of-the-art gradient-based adversarial attacks [37, 39] as well as common corruptions which include noise injection, weather condition, common blur, and digital corruptions. Therefore, our neuro-inspired work provides a valuable path towards general-purpose robustness against noise, adversarial perturbations, and common corruptions [1, 40, 41].

Chapter 2

Learning stable and robust RF fingerprints

In this chapter, we discuss the importance of being proactive in guiding deep neural networks to extract stable and robust features. We present our work in the radio frequency (RF) fingerprinting setting and explore the effects of confounding factors on learning. We provide the background on RF fingerprinting along with related work in this area in section 2.1. Furthermore, we explain the issue of the confounding factors in the RF context. We then discuss the details of our approach to deal with these confounding factors in section 2.2. Furthermore, we present and share our simulated WiFi dataset with the community to spark further exploration in this direction.

2.1 Introduction

An important tool in wireless security is a "fingerprint" based on physical layer characteristics, capable of distinguishing between different devices even if they are transmitting exactly the same message. This is possible due to subtle hardware imperfections that

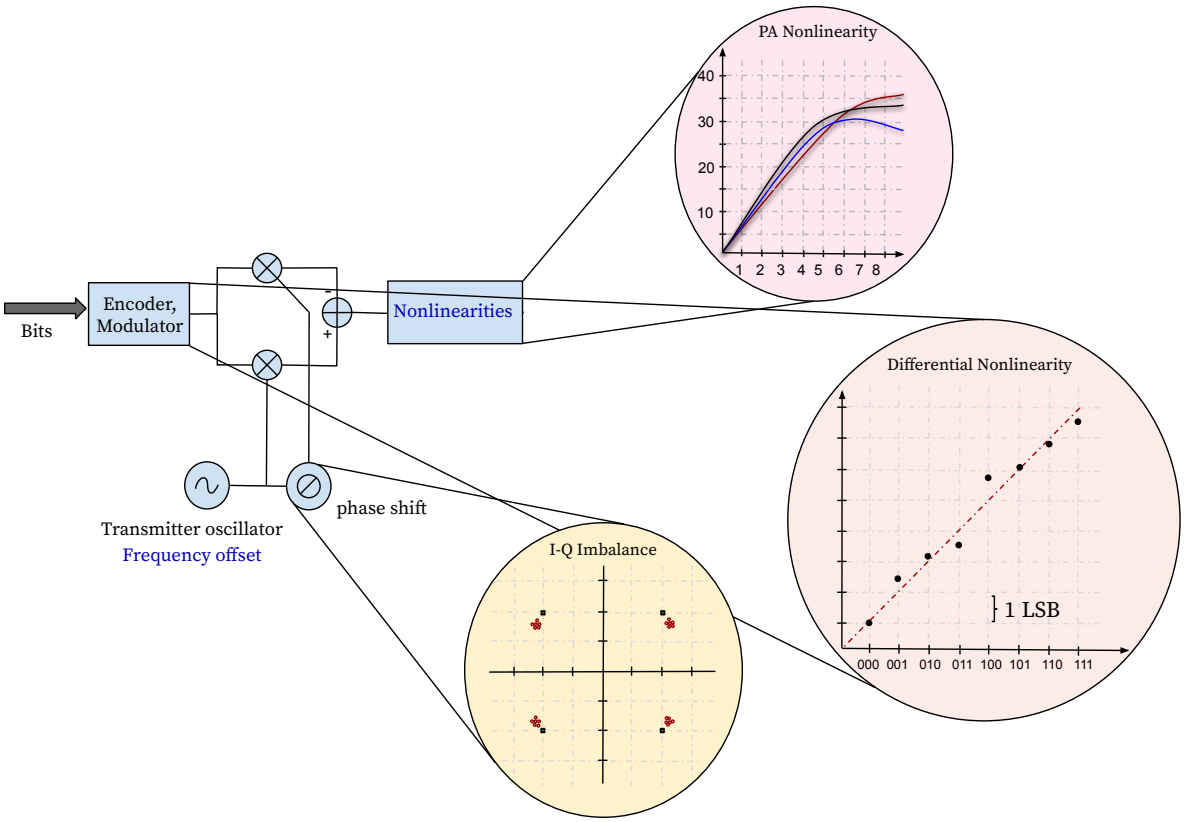


Figure 2.1: Block diagram of a transmitter and several sources of nonlinear features.

occur even in devices made by the same manufacturer [42, 43]. Figure 2.1 depicts the potential sources of nonlinear features such as digital-to-analog converters (DACs) [44], power amplifiers (PAs) [45, 46] and I/Q imbalances [21]. These components deviate slightly from their ideal values (differently for each device) in a typical regime provided by IEEE standards for wireless communications. The goal of RF fingerprinting is to translate these signatures into a device fingerprint.

Fingerprints based on such variations could potentially serve as a powerful authentication tool at the physical layer, complementing conventional security schemes in higher layers of the networking stack [18]. Since these subtle nonlinear effects are difficult to model explicitly, deep learning is a natural approach to teasing out transceiver signatures

based on them. In this work, we investigate the efficacy of extracting fingerprints which are robust to variations across time and location, using one-dimensional complex-valued convolutional neural networks (CNNs) that operate on the complex baseband signal at the receiver.

Our results show that deep learning is a promising tool for wireless fingerprinting, while sounding a cautionary note. The key message is that the network learns the easiest set of features that it can in order to accomplish the desired task (in our case, discriminating between transmitters based on the received wireless signal), hence we must be extremely proactive in promoting robustness across effects that we do not want the network to lock on to, which we term *confounding factors.* For instance, we would like the radio frequency (RF) signature for a transmitter to be robust across time and for different wireless channels. However, if we employ training data collected over a short period of time when the channel and carrier frequency offset (CFO) for a transmitter are relatively constant, the CNN will lock onto these rather than to subtle nonlinear effects. This gives unreasonably excellent accuracy on test data collected over the same time period, but disastrous results for data collected on a different day, when both the channel and the CFO can be different. Figure 2.2 depicts this difference between multiple days.

We develop augmentation strategies based on signal models for the impact of confounding factors, and evaluate performance against classical compensation techniques that explicitly try to undo them. We find that compensation works well if the undesired features are simple enough, like the CFO. However, for more complex effects such as a multipath channel, model-driven augmentation outperforms explicit estimation and compensation for learning robust signatures. A significant finding is that augmentation is useful not just during training, but also during inference: averaging of predictions from multiple augmented versions of the same input provides significant performance gains.

In order to perform controlled experiments, we evaluate our approach on *emulated*

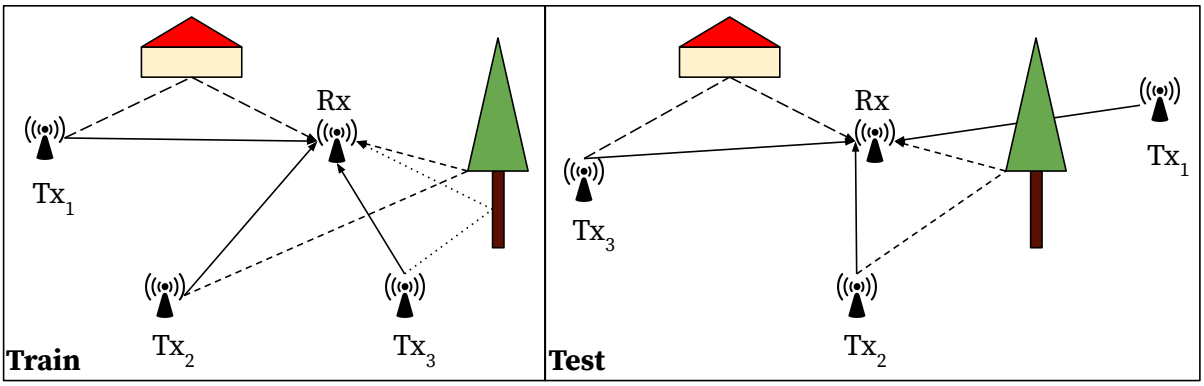


Figure 2.2: Strong confounding factors such as multipath fading channel and carrier frequency offset change over time.

data, in which "clean" measured data is passed through simulated channels and CFO. Since this measured data, obtained as part of the DARPA RFMLS program, cannot be made public, we also make publicly available a *simulation-based* dataset based on models of some typical circuit-level nonlinearities [21–23]. The results we obtain on this dataset are comparable to those from the measurement-based dataset, enabling reproducibility. The dataset and code are available at [24].

Since we wish to be robust against software spoofing, we focus on extracting fingerprints only from the packet preamble. This is a worst-case approach motivated by prior work [18] demonstrating the tendency of CNNs to lock on to any packet field correlated with the ID field (which is easily spoofed) as the "shortest path" to meeting the classification objective.

2.1.1 Related Work

There are many papers over the past decade using machine learning to derive fingerprints. Much of this work involves significant protocol-specific preprocessing, in contrast to the protocol-agnostic approach considered in this work. An early example is the use

of support vector machine (SVM) in [43] based on demodulation error metrics such as frequency offset and I/Q offset. However, this detection method was defeated in [47, 48], who showed that these modulation features could be impersonated via software-defined radios. Other examples of machine learning based, protocol-specific fingerprints include: a k-nearest neighbor (k-NN) classifier in [49] based on spectral analysis of WiFi preambles; linear discriminant analysis (LDA) in [50] after pilot-aided compensation of RF nonlinearities caused by the receiver; k-means clustering of features based on inter-arrival times of ADS-B messages [51]; a neural network in [52] and k-NN in [53] operating on WiFi inter-arrival times; frequency compensation of ZigBee data, followed by a CNN [54]; and a CNN operating on the error signal obtained after subtracting out an estimated ideal signal from frequency-corrected received data [55]. Section 2.2 evaluates the robustness of our approach against protocol-specific estimation strategies, showing that, while estimation works well for simple phenomena such as CFO variations, the augmentation approach that we study has a clear advantage for more complex effects such as channel variations.

Modern CNNs learning directly from I/Q data include [56, 57] for modulation classification, and [58, 59] for device fingerprinting. This line of work employs real-valued networks, with real and imaginary parts of complex data treated as different channels. Such networks have more degrees of freedom compared to a complex network where the convolution operation is more restricted. Consider a complex convolution operation between input X and weight W, resulting in output Y:

$$\mathrm{Re}(Y) + j\,\mathrm{Im}(Y) = (\mathrm{Re}(W) + j\,\mathrm{Im}(W)) * (\mathrm{Re}(X) + j\,\mathrm{Im}(X))$$

This can be rewritten in the following form [60, 61] with the real and imaginary parts of the input stacked as different channels:

$$\begin{bmatrix} \mathrm{Re}(Y) \\ \mathrm{Im}(Y) \end{bmatrix} = \begin{bmatrix} \mathrm{Re}(W) & -\mathrm{Im}(W) \\ \mathrm{Im}(W) & \mathrm{Re}(W) \end{bmatrix} * \begin{bmatrix} \mathrm{Re}(X) \\ \mathrm{Im}(X) \end{bmatrix} \tag{2.1}$$

Therefore, a complex network with the CReLU activation function ($\mathrm{ReLU}(\mathrm{Re}(x)) + j\mathrm{ReLU}(\mathrm{Im}(x))$) can be considered a regularized form of a real ReLU network, with the weight matrix restricted to the structure in (2.1). This reduction in number of degrees of freedom has been shown to improve generalization performance [62]. We note that this analysis does not hold for complex networks with the ModReLU activation function ($\mathrm{ReLU}(|x|)\exp(j\angle x)$), which we find yields better performance than CReLU for our application; ModReLU-based architectures cannot be realized by a real ReLU network. It has been observed in recent work that complex networks provide advantages over real networks for the tasks of MRI fingerprinting [63], radar-based terrain classification [64], audio source separation [65], music transcription [61] and channel equalization [66]. Our results on the gain provided for the fingerprinting problem are in line with such prior work, and motivate further exploration of neural networks tailored to complex-valued data. It is worth noting that, for real-valued networks, standard DNNs and CNNs are compared with multi-stage training (MST) of simple building blocks for fingerprinting in [67], with MST yielding the best performance. Such work highlights the need for continued architectural experimentation for both real- and complex-valued networks.

The present work builds on our conference paper [18], which considers the impact of ID spoofing and SNR on CNN-based fingerprinting. To our knowledge, [18] was the first to employ complex-valued CNNs for wireless fingerprinting; it precedes and is independent of [68], which also uses complex-valued networks. The main focus of this work is different: we investigate robustness of fingerprints to variations in the CFO and wireless channel. While [18] considers noise augmentation to handle SNR mismatch between training and test data, in the present work, we consider augmentation and compensation strategies for CFO and channel, and introduce the concept of test time augmentation for handling confounding factors. We should note that the concept of test time augmentation proposed here is different from classical ensemble methods such as boosting or bagging [69, 70]:

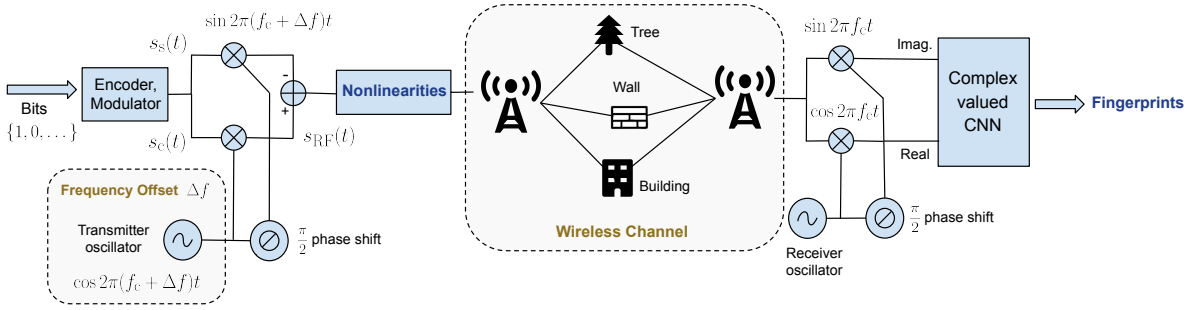


Figure 2.3: Block diagram of a wireless communication system. Subtle nonlinearities unique to each device can provide a fingerprint. However, easy-to-learn features such as the CFO and channel are not stable over time and location, affecting generalization.

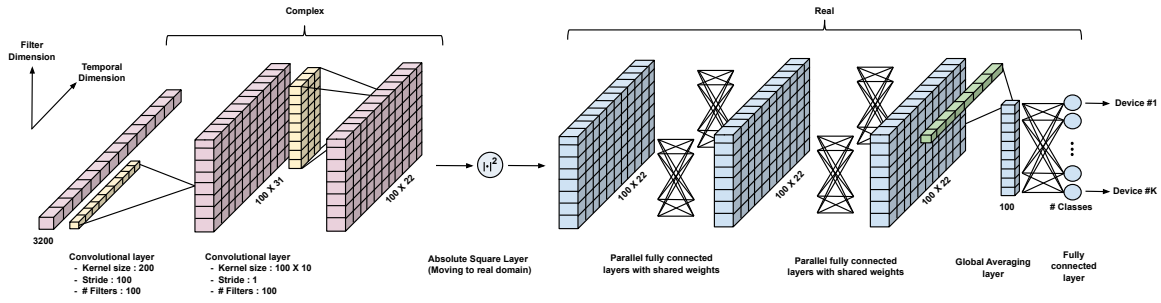


Figure 2.4: Complex-valued 1D CNN architecture for WiFi signals.

rather than averaging over an ensemble of machines, we are averaging over an ensemble of inputs. Given recent promising results on the use of boosting techniques in multilayer settings [71–73], it is of interest to explore comparison and possibly combination of such techniques with our augmentation strategy for deriving RF signatures robust to confounding factors.

In [74], channel-resilient fingerprinting was studied by modifying the transmitter using a finite impulse response (FIR) filter. Our work on channel resilience is based solely on modifying DNN training and does not involve transmitter-side alterations. In recent work, [75, 76] reported a significant degradation in accuracies when training and test data were from different days, with fingerprints extracted using real-valued CNNs. While

equalization was observed to improve performance in the different day scenario, it caused a drop in accuracy when training and test data were from the same day. These results are in line with our observations in Section 2.2.3: while equalization can help, the residual error from this approach appears to swamp out the nonlinear characteristics we are interested in. We find model-based augmentation to be a more effective strategy for learning robust fingerprints.

2.2 Stability to CFO and Channel Variations

In this section, we use a clean WiFi dataset for controlled experiments emulating the effect of frequency drift and channel variations. This dataset contains a mix of IEEE 802.11a ($f_c = 5.8$ GHz) and IEEE 802.11g ($f_c = 2.4$ GHz) packets from 19 commercial-off-the-shelf devices, collected indoors without channel distortion using a Tektronix RSA5126B receiver. When using the preamble alone, we can obtain 99.5% fingerprinting accuracy for 19 WiFi devices using a complex-valued CNN. (Detailed information about CNN architecture and training parameters can be found in section 2.3.) However, as we show below, CFO and channel variations can result in a disastrous effect on performance. We study compensation and augmentation strategies to promote robustness.

2.2.1 Nuisance Parameters, Compensation and Augmentation

Before providing specific results, we lay out our overall framework.

Consider input data $\boldsymbol{x}$ (the packet preamble in our case) fed to a neural network which aims to classify the device ID y. In our present context, we may think of this input data as a transformation of an ideal input $\boldsymbol{x}_{\text{ideal}}$ capturing the desired characteristics of the device, passed through a transformation f_θ, where θ is a nuisance parameter such as a CFO or channel: $\boldsymbol{x} = f_\theta(\boldsymbol{x}_{\text{ideal}})$. A network trained with such inputs would ideally produce

posteriors $p(y|\boldsymbol{x}) = p(y|f_\theta(\boldsymbol{x}_{\text{ideal}}))$ as the softmax outputs. In the scenarios of interest, we define a single "day" of training as a scenario in which θ is fixed during the training period for a given device, but differs across different devices. In this case, it is natural for the DNN to use information in θ to classify devices. Indeed, if the discrimination based on θ is easier than that based on the subtle nonlinear signatures buried in $\boldsymbol{x}_{\text{ideal}}$, then the DNN will focus on using θ rather than the information in $\boldsymbol{x}_{\text{ideal}}$. When we then test on a different "day" when the value of the nuisance parameter θ is different, we understandably get poor performance.

Compensation: If we have detailed protocol-level information and good enough models, then it is possible to try to invert f_θ to recover $\boldsymbol{x}_{\text{ideal}}$ from $\boldsymbol{x}$, and to then train the DNN based on this estimate. For example, we can estimate and undo a CFO, or equalize a channel. For the particular experiments we do, we find that compensation works well for simple nuisance parameters such as the CFO, but that the residual errors after equalization are enough to swamp out the subtle nonlinear effects we are after.

Augmentation: An alternative to protocol-specific compensation strategies is to use models for how the nuisance parameters operate on the input to augment the data. Specifically, we create new inputs of the form $\boldsymbol{x}' = f_{\theta_{\text{aug}}}(\boldsymbol{x})$, where we choose θ_{aug} from a set Θ such that

$$\boldsymbol{x}' = f_{\theta_{\text{aug}}}(\boldsymbol{x}) = f_{\theta_{\text{aug}}}(f_\theta(\boldsymbol{x}_{\text{ideal}})) \approx f_{\theta'}(\boldsymbol{x}_{\text{ideal}}) \ , \ \theta' \in \Theta$$

where θ' is an "effective" nuisance parameter. Now, if we train the DNN using multiple augmentations of $\boldsymbol{x}$, then we hope that the network learns to use $\boldsymbol{x}_{\text{ideal}}$ to a greater extent than before, since we are varying θ' for a given device. Nevertheless, standard training does not *guarantee* marginalization over θ'. Rather, it allows the network to produce posteriors of the form $p(y|\boldsymbol{x}') = p\big(y|f_{\theta_{\text{aug}}}(f_\theta(\boldsymbol{x}_{\text{ideal}}))\big) \approx p(y|f_{\theta'}(\boldsymbol{x}_{\text{ideal}}))$, where hopefully the information from $\boldsymbol{x}_{\text{ideal}}$ is being used to a greater extent because of training

augmentation. When we are now presented with a fresh test input $\boldsymbol{x} = f_\theta(\boldsymbol{x}_{\text{ideal}})$, we are not guaranteed that this particular realization of the nuisance parameter θ is comfortably far from the decision boundaries that the network has learnt. On the other hand, test time augmentation allows us to generate multiple effective nuisance parameter realizations which we can average over.

$$\frac{1}{|\Theta_{\text{test}}|} \sum_{\theta_{\text{aug}} \in \Theta_{\text{test}}} p\big(y | f_{\theta_{\text{aug}}}(f_\theta(\boldsymbol{x}_{\text{ideal}}))\big) \tag{2.2}$$

Thus, we are effectively averaging over $|\Theta_{\text{test}}|$ realizations of the "effective" nuisance parameters θ'.

Residual approach: An interesting way to combine the above two strategies is by excising a reconstruction of the transmitted message based on a linear model to obtain a residual signal containing device nonlinearities. Using the known preamble sequence and estimated CFO and channel, we can compute an ideal noiseless reconstruction $\widehat{\boldsymbol{x}}$ of the received signal $\boldsymbol{x}$. The residual noise $\boldsymbol{x} - \widehat{\boldsymbol{x}}$ can then fed as input to a neural network. Since this residual signal still contains CFO and channel effects, we find that this technique does not work well on its own. However, it can be used in combination with augmentation to confer robustness.

In the following sections, we assess performance using the average of five different runs, with different random realizations of CFOs and channels used for emulation and augmentation, as well as different random seeds for CNN weight initialization. In all graphs, error bars denote one standard deviation from the mean over different runs.

2.2.2 Carrier Frequency Offset

The carrier frequency offset, caused by frequency mismatch in the crystal oscillators at the transmitter and receiver, could potentially be used as a feature to fingerprint devices

[43, 77]. However, we treat it here as a confounding factor for our goal of obtaining a fingerprint which is stable over time. Oscillator frequencies are affected by a few parts per million (ppm) for every 1°C change in temperature [78], and therefore drift daily, and are also affected by aging [79]. The CFO can also be spoofed by a sophisticated enough adversary manipulating baseband signals [47, 48]. While the CFO could still be a useful feature as a defense against simpler attacks (e.g., for systems with relatively frequent transmissions, its slow drift could be tracked across packets to detect abrupt transitions), its role as a confounding factor in our study enables us to benchmark augmentation against compensation for an effect which can be accurately modeled. We investigate this by inserting offsets in data, emulating an oscillator frequency tolerance of ± 20 parts per million as specified in the IEEE 802.11 standard [23]. We begin with an example where only the test data is offset.

Offset in test data alone: We find that networks trained on clean data do *not* generalize to offset data, even when the offset is very small: as shown in the first row of Table 2.1, accuracy drops to 4.6% at an offset of 20 ppm. In order to alleviate this, we augment the training set with randomly chosen CFOs and report results in the second and third rows of Table 2.1. We consider two types of random offsets: Bernoulli $\{-20, 20\}$ ppm and uniform $(-20, 20)$ ppm, augmenting the size of the training set by 5x in each scenario.

Table 2.1: Performance when only the test data is offset, with CFOs in the range (-20, 20) ppm. The first row shows that this results in poor accuracies if we do not modify our training strategy. Rows 2 and 3 then demonstrate that augmenting training data with uniformly distributed CFOs helps confer robustness.

Type of data augmentation	CFO in test set		
	None	Bernoulli	Uniform
None	99.50	4.63	13.58
Bernoulli	3.32	99.32	13.53
Uniform	96.21	90.79	95.37

This strategy can significantly help in learning robust fingerprints, but the type of augmentation matters: in particular, it is insufficient to augment with worst-case offsets alone. When we train with Bernoulli offsets, the network becomes robust to Bernoulli test offsets (99.3%), but fails to generalize to any offset smaller than 20 ppm, including an offset of zero. In contrast, when we augment data with uniformly chosen offsets, we obtain resilience (>90%) to all test set offsets in the desired range.

"Different day" scenario (no augmentation or compensation): We now emulate collecting training data on one day and testing on another: given clean data $\boldsymbol{x}_{\text{ideal}}$, we add CFOs θ to emulate the effect of different days: $f_\theta(\boldsymbol{x}_{\text{ideal}})$. We insert different "physical" offsets for each device, but fix the offset for all packets from a particular device. The offsets are randomly chosen in the range $(-40, 40)$ ppm (since both the transmitter and receiver oscillators can vary by ± 20 ppm). Oscillator drift across days is realized via different random seeds for training and test offsets.

This "different day" setting makes it particularly easy for the network to focus on the

Table 2.2: Comparison of augmentation, compensation and the residual approach in the "different day" CFO scenario. The training and test datasets are augmented by 20 and 100 times respectively.

Training strategy	Test accuracy
Baseline (no augmentation or compensation)	9.68
Augmentation	91.47
Residual + Augmentation	93.21
Compensation	96.37

CFO as a fingerprint: since each device has a different offset on each day, training on a single day leads to the DNN focusing on using the CFO as a means of distinguishing between devices. This results in artificially high training accuracies (94.2%), but poor test set performance (9.7%) on a different day when the devices have different CFOs. We now explore two strategies to restore performance: data augmentation with randomly chosen CFOs, and frequency compensation.

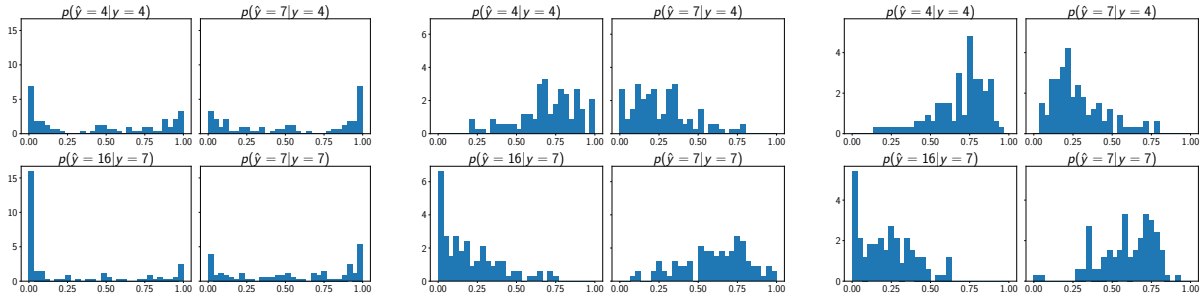
"Different day" scenario with augmentation: In order to promote robustness, we add new, randomly chosen CFOs θ_{aug} *on top* of the CFOs used for different day emulation: $f_{\theta_{\text{aug}}}(f_\theta(\boldsymbol{x}_{\text{ideal}}))$. Table 2.3 reports on the efficacy of various CFO augmentation strategies, capable of increasing test accuracy to 91.5%. For training data, we find that the best augmentation technique is to use a different augmentation offset for each packet from a device, but the same set of offsets across devices, which discourages the network from learning the CFO as a means of distinguishing between devices. We term this an "orthogonal" strategy: we are trying to train in a direction "orthogonal" to the tendency to lock onto the "physical" CFO as a signature.

A novel finding is that *data augmentation for testing* leads to significant performance gains when we add up soft outputs across augmented versions of each test packet. The best result is obtained when we insert a different randomly chosen CFO for each of a 100 copies of each test data packet, and then sum up the softmax outputs across the augmented data. We find that averaging of logits also improves performance, but not to the extent of the softmax average.

Table 2.3: Effect of augmentation in the "different day" CFO setting, with CFOs in the range (-40, 40) ppm. "Random" training augmentation uses a new randomly chosen CFO for each packet, while the "orthogonal" type uses the same set of offsets across devices. In both cases, the offsets are drawn from a uniform distribution.

Training augmentation		Test time augmentation			
		None	5	20	100
None	–	9.68	7.84	8.74	8.47
Random	5	74.21	71.84	74.21	77.37
	20	72.79	75.84	78.05	80.05
Orthogonal	5	69.58	75.11	81.05	83.63
	20	82.37	82.32	86.21	91.47

"Different day" scenario with frequency compensation: We can also estimate and correct the offset using knowledge of the periodic structure of the preamble. Consider a periodic signal $s[n]$ with period L, and frequency offset θ resulting in $r[n] = s[n]\ \exp(j2\pi n\,\theta)$. Since we know that $s[n] = s[n + L]$, the CFO can be estimated by correlating r with its



(a) No test augmentation. (b) 10 test augmentations. (c) 100 test augmentations.

Figure 2.5: Plots showing how test augmentation affects the histogram of softmax outputs $p(\hat{y})$ (averaged over augmentations) for data from two specific devices ($y = 4$ and $y = 7$), in the "different day" channel setting. Histograms are normalized to be probability densities. As the number of test augmentations increases, the probability of correct prediction $p(\hat{y} = 4|y = 4)$ and $p(\hat{y} = 7|y = 7)$ shifts towards 1.

shifted version:

$$\widehat{\theta} = \frac{1}{2\pi L} \angle\left(\sum_n r[n]\, r^*[n+L] \right).$$

We follow a two-step approach [80] involving a coarse estimate from the 802.11 short training sequence ($L = 16$) and then a fine estimate from the long training field ($L = 64$). This method restores accuracy to 96.4%, and, as shown in table 2.2, its accuracy is about 4.9% better than that with augmentation.

Residual approach: We could also use the estimated CFO to compute a residual signal that can be fed as input to a CNN, as described in Section 2.2.1. This approach can be combined with augmentation to obtain a performance improvement over pure augmentation, as shown in Table 2.2. Stripping out the message in this manner makes it easier for the network to learn nonlinear signatures.

2.2.3 Multipath Channels

The wireless channel is another important source of distribution shift between training and test data. Since multipath components in the channel depend on propagation geometry, a network that locks on to the channel will fail to generalize to test data collected on a different day or location. If the training data does not span a sufficiently diverse set of geometries, it could contain channels that are highly correlated with the transmitter ID, necessitating the use of channel augmentation or equalization strategies to improve robustness.

Table 2.4: Power-delay profile for the EPA multipath fading model. Tap amplitudes A_k are Rayleigh distributed with variance P_k.

k	1	2	3	4	5	6	7
τ_k (ns)	0	30	70	90	110	190	410
P_k (dB)	0.0	-1.0	-2.0	-3.0	-8.0	-17.2	-20.8

Table 2.5: Performance in the "different day" channel setting when we train on 2 days and test on a third day. "Random" augmentation uses a randomly drawn channel for each packet, while the "orthogonal" type uses the same set of channels across devices.

Training augmentation		Test time augmentation				
		None	1	5	20	100
None	–	5.74	6.74	7.26	7.21	7.26
Random	5	39.58	39.79	54.05	59.84	62.68
	20	54.05	52.84	63.21	67.68	68.47
Orthogonal	5	41.16	42.16	52.89	56.68	58.68
	20	56.16	54.74	66.47	71.00	71.84

We study the impact of multipath on fingerprinting using a Rayleigh fading model [81] with L multipath components: $h(t) = \sum_{k=1}^{L} A_k e^{j\phi_k} \delta(t - \tau_k)$, where $A_k \sim \text{Rayleigh}\,(P_k)$, $\phi_k \sim \text{Uniform}\,(0, 2\pi)$ and $\delta(\cdot)$ is the Dirac delta function. We use the Extended Pedestrian A (EPA) profile, a well-known statistical channel model used in LTE system testing [82]. As shown in Table 2.4, this profile quantifies the delays τ_k and relative powers P_k of the multipath components.

"Different day" scenario (no augmentation or equalization): We investigate training and testing on different emulated days similar to prior CFO experiments. Using the EPA profile, we use different realizations of the channel vector for each day and for each device. Each realization has 7 multipath components chosen from a Rayleigh distribution with relative powers and delays specified in Table IV. We do not vary the channel realization for a given device on a given day, hence we are modeling quasi-static environments. With single day training, we get excellent performance when testing on

the same day (98%), but very poor accuracy if we test on a different day (5.8%). This clearly indicates a lack of robustness to channel variations, with the network involuntarily locking on to the channel as a means of discriminating between devices.

"Different day" scenario with augmentation: Assuming the received data is $f_\theta(\boldsymbol{x}_{\text{ideal}})$, we study the effect of channel augmentation θ_{aug} *on top* of the emulated data: $f_{\theta_{\text{aug}}}(f_\theta(\boldsymbol{x}_{\text{ideal}}))$. We find that augmentation helps, but accuracy increases only to 47.8% in the "train on one day, test on another" setting. We can boost performance to 71.8% if we are allowed access to training data over 2 emulated days (without increasing the size of the training set) and test on a third day, as shown in Table 2.5. Note that accuracy without augmentation is still low. If training data spans 3 days, augmentation improves accuracy even further to 79.7%.

This phenomenon can be understood by modeling channel variations in the frequency domain. Suppose transmitter i sends message X_i over "physical" channel H_i: $Y_i(f) = H_i(f)\,X_i(f)$, and we augment with randomly chosen channels G: $\tilde{Y}_i(f) = G(f)\,Y_i(f) = G(f)\,H_i(f)\,X_i(f)$. The effective channel $G(f)\,H_i(f)$ will still contain all the nulls of H_i, which could potentially be correlated with the transmitter ID. Thus, augmentation alone cannot completely remove the effect of the underlying physical channel. Access to more varied training data, when combined with augmentation, increases the diversity of the overall channel that the network sees.

The preceding results are achieved using 20 training and 100 test augmentations (with soft outputs added up over 100 augmented copies of each test packet). As before, we find that the "orthogonal" approach works the best for training: using the same set of channels across devices discourages the network from learning to use the channel as a fingerprint. Fig. 2.5 illustrates the impact of test time augmentation on the distribution of soft outputs $p(\hat{y})$ for two sample devices. If we do not augment the test set, many

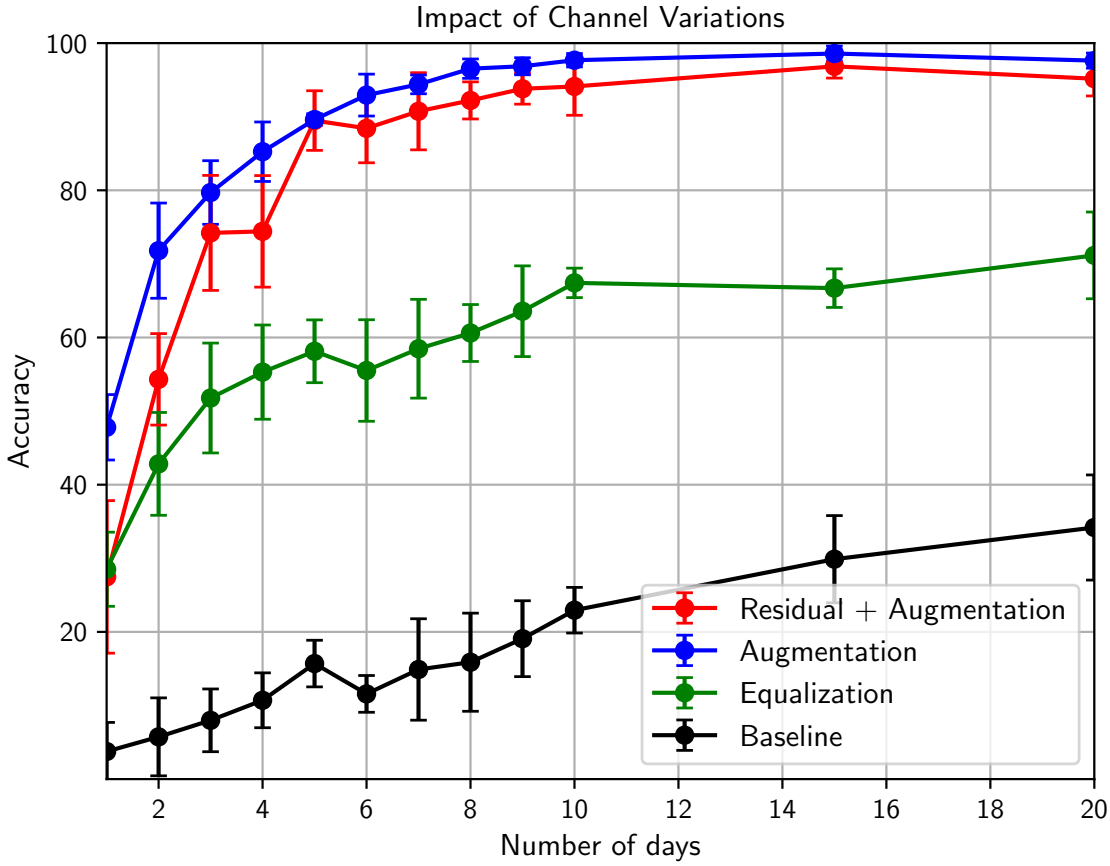


Figure 2.6: Comparison of channel equalization and augmentation as we increase the number of emulated days for training (with the size of the training set kept constant). Baseline accuracies are reported for a network trained without augmentation or equalization.

samples from device 4 are misclassified as device 7 (shown in the first row of Fig. 2.5a). As the number of test augmentations increases (Fig. 2.5b, 2.5c), we get increasingly precise estimates of the desired prediction (2.2), causing $p(\hat{y} = 7|y = 4)$ to shift towards 0, and $p(\hat{y} = 4|y = 4)$ towards 1.

"Different day" scenario with equalization: Another strategy to remove channel influence would be to equalize signals using the long training field of the WiFi preamble. We equalize data in the frequency domain and compare results with augmentation in Fig. 2.6. Each experiment is performed with 5 different seeds, with error bars denoting one standard deviation from the mean. We find that equalization performs much poorer than channel augmentation, with a performance gap of 26.5% even with 20 training days. It appears that the residual distortion after equalization is large enough to swamp out the nonlinear characteristics that we are interested in.

Residual approach: As previously described (Section 2.2.1), we can use the estimated channel to obtain residual noise and use it as CNN input. When combined with augmentation, we obtain accuracies that are competitive with, but not better than, pure augmentation, as shown in Fig. 2.6. We speculate that errors in channel estimation prevent the residual method from offering a clear advantage in accuracy, in contrast to the simpler setting of CFO uncertainty considered in Section 2.2.2.

Overall, augmentation is the best of the three considered strategies for making networks insensitive to channel effects: with 10 training days, it can restore accuracy to 97.7%.

2.2.4 Combination of Channel and Carrier Offsets

Lastly, we focus on a combination of channel and carrier offsets across different days. This is a harsher and more realistic setting than prior experiments, with test set accuracy without augmentation or compensation no better than random guessing (5%) even if

training data spans 20 emulated days.

Augmentation: We explore data augmentation with randomly generated channels and CFOs, and report results in Figures 2.7 and 2.8. We find an equal number of augmented CFOs and channels to work well: when using 20 training days, performance improves from 5% to 84.4% with training augmentation alone, and to 90.1% with both training and test augmentation. We observe that the amount of test augmentation is important: as shown in Fig. 2.9, if we only augment test data 2 times, we observe a drop in accuracy. This is because the Bayesian average (2.2) requires a large number of realizations of the two nuisance parameters (CFO, channel) in order to be accurate.

Estimation: Table 2.6 reports on comparisons with estimation strategies, the residual approach and also a mix of estimation and augmentation. We find that equalization, when combined with either CFO compensation or augmentation, results in poor accuracies and therefore do not include it in the comparison. The best result is obtained by a combination of CFO compensation and channel augmentation for both training and test sets, with competitive performance from pure augmentation when the number of days of training is large.

Table 2.6: Comparison of augmentation, estimation and the residual approach when both the CFO and channel vary.

Training strategy	Number of days			
	2	5	10	20
Residual + augmentation	19.11	26.21	67.50	78.95
Pure augmentation	24.90	49.36	77.83	90.10
CFO comp. + channel aug.	33.96	62.63	88.96	91.40

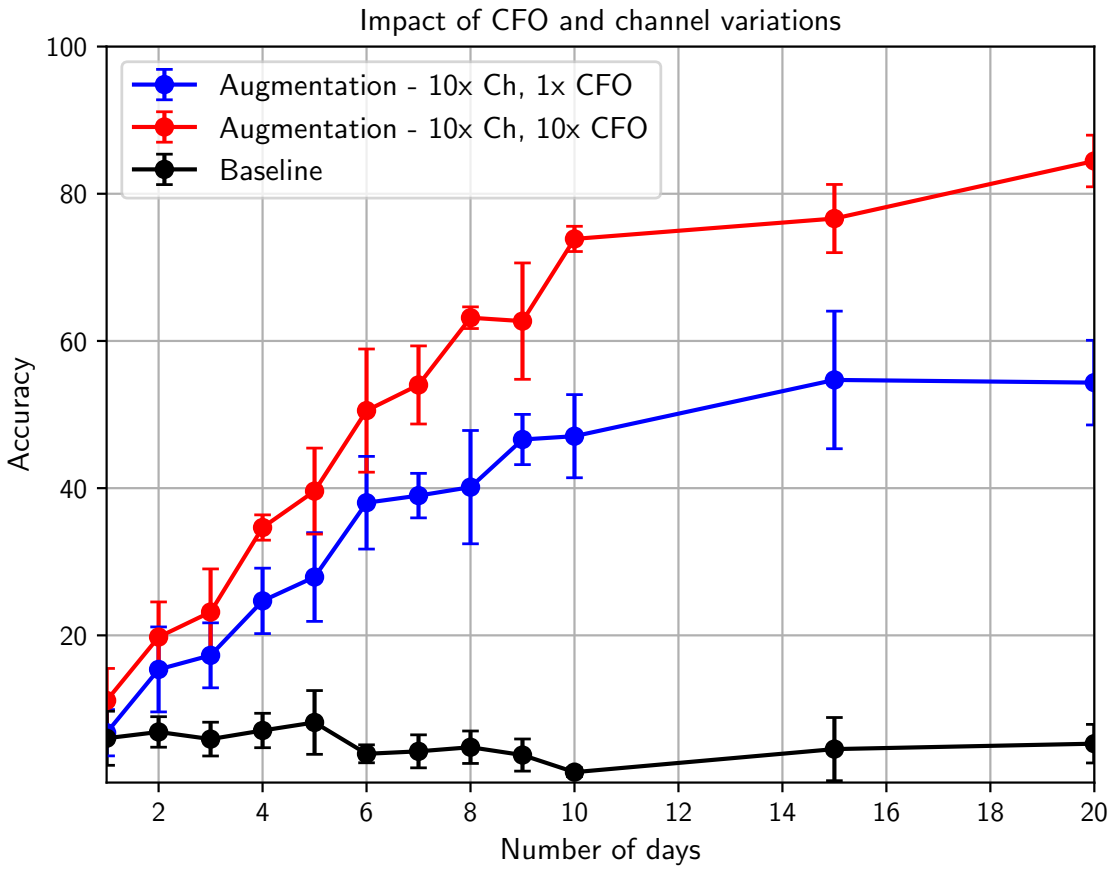


Figure 2.7: Performance of training augmentation across days when there is a combination of CFO and channel variations. We use the orthogonal augmentation approach for channels and the random method for CFOs.

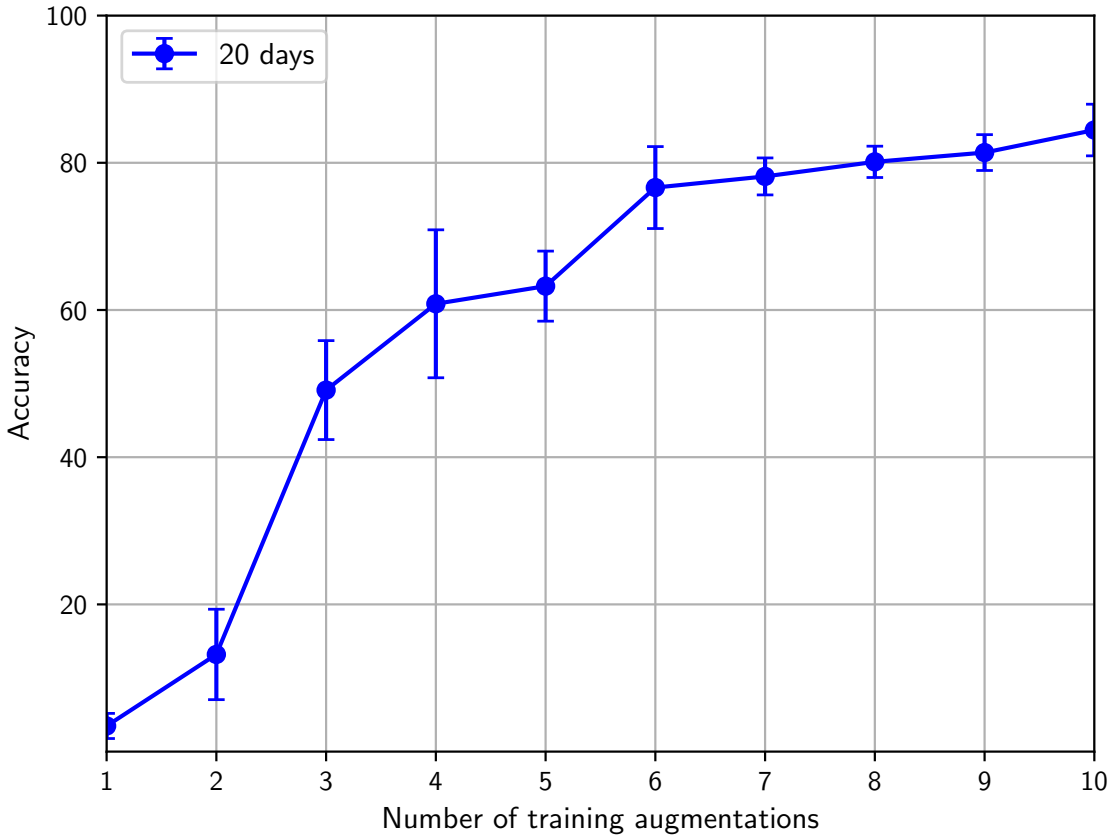


Figure 2.8: Accuracy as a function of the amount of training augmentation when both the CFO and channel fluctuate. We augment the CFO and channel by equal amounts, with the x-axis denoting the number of augmentations for each.

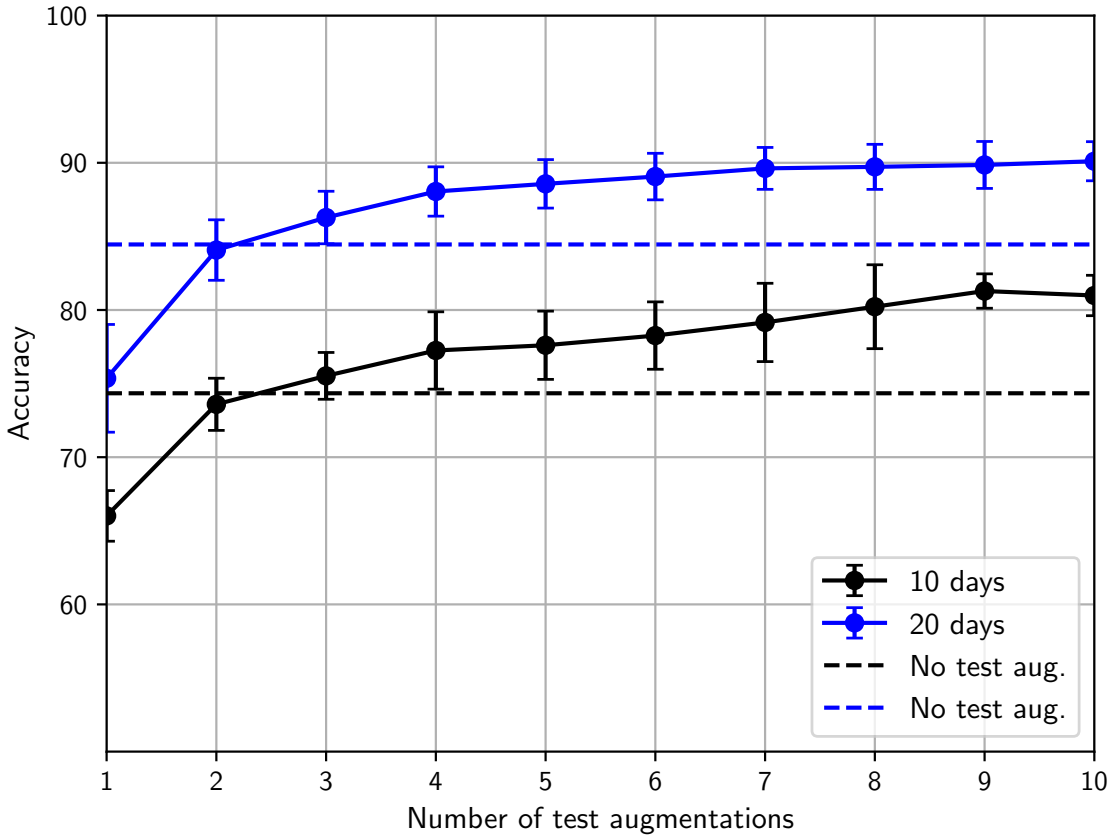


Figure 2.9: Accuracy as a function of the amount of test augmentation when both the CFO and channel fluctuate. We augment the CFO and channel by equal amounts, with the x-axis denoting the number of augmentations for each.

2.2.5 Simulation-Based Dataset

Since the datasets used in the previous sections are not publicly available, in the interest of reproducibility and as a contribution to the community, we have created a simulation-based WiFi dataset [24] based on models of some typical nonlinearities [21–23]. We implement two different kinds of circuit-level impairments: I/Q imbalance and power amplifier nonlinearity with Figure 2.10 depicting the order in which the nonlinear effects were added. We skip effects of the digital to analog converter such as DNL and INL.

In a manner similar to prior sections, we perform experiments to study the effect of channel and CFO variations on fingerprinting performance. We now discuss the models and parameters used to generate the nonlinear effects.

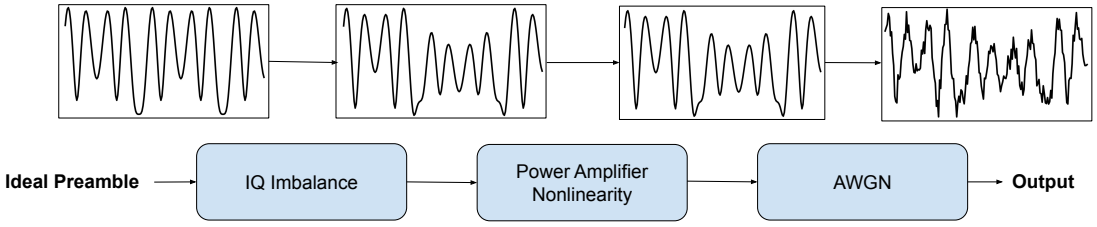


Figure 2.10: Block diagram for generation of the simulation-based dataset.

Table 2.7: Fingerprinting performance on the simulated dataset in the "different day" scenario for both CFOs and channels, when using 20 days for training.

Training strategy	Test time augmentation		
	None	1	100
No augmentation or compensation	7.61	6.68	8.30
Pure augmentation	81.38	77.56	86.24
CFO comp. + channel aug.	81.59	81.98	91.80

I/Q Imbalance: The I/Q imbalance [21] can be modeled as follows, with parameters ϵ and ϕ representing gain and phase mismatch respectively:

$$\tilde{s}_{\mathrm{RF}}(t) = s_{\mathrm{c}}(t)\left(1+\frac{\epsilon}{2}\right)\cos\left(2\pi f_{\mathrm{c}}t+\frac{\phi}{2}\right) - s_{\mathrm{s}}(t)\left(1-\frac{\epsilon}{2}\right)\sin\left(2\pi f_{\mathrm{c}}t-\frac{\phi}{2}\right) \tag{2.3}$$

Since the IEEE 802.11 WiFi standard [23] specifies an error vector magnitude (EVM) of -19 dB, we set $\epsilon \leq 0.2$ and $|\phi| \leq \pi/30$. In order to simulate 19 different devices (similar to original dataset) we choose distinct ϵ values for each device from the set $[0, 0.2]$ uniformly,

i.e. $\{0, 0.2/19, 0.4/19...\}$. Similarly, we pick ϕ from the set $[-\pi/30, \pi/30]$ uniformly. We note that all the values are shuffled randomly before matching to each device, hence extreme cases for both parameters are most likely not on the same device. An example of I/Q imbalance for 4-QAM modulation is provided on figure 2.11

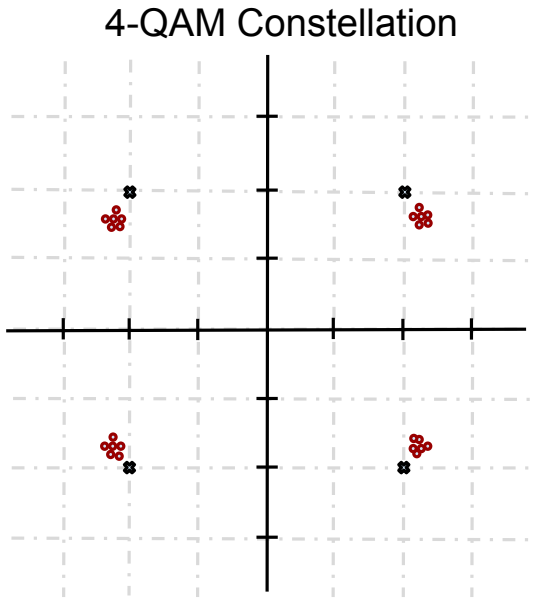


Figure 2.11: Scatterplots of noisy 4-QAM constellation points with and without I-Q imbalance.

Power Amplifier Nonlinearity: The power amplifier (PA) is another source of circuit-level nonlinearity that varies across devices. There are a number of different models for this nonlinearity [45, 46, 83, 84]. We model PA nonlinearities as a saturated third-order polynomial function [22]:

$$y(t) = \begin{cases} x(t) \cdot \left(1 - \dfrac{0.44|x(t)|^2}{3P_{\text{1dB}}}\right) & \text{if } |x(t)|^2 \leq \dfrac{P_{\text{1dB}}}{0.44}, \\ \dfrac{x(t)}{|x(t)|}\sqrt{P_{\text{1dB}}} & \text{if } |x(t)|^2 > \dfrac{P_{\text{1dB}}}{0.44}. \end{cases}$$

This function is parametrized by the 1 dB compression point P_{1dB}, defined as the output power level at which the gain decreases 1 dB from its constant value. Similar

to I/Q imbalance, we determine the range of the values for $P_{1\text{dB}}$ that satisfy the EVM specifications. We choose $P_{1\text{dB}}$ values for each device uniformly from the set $[8.45, 20]$. The corresponding transfer functions are depicted on figure 2.12.

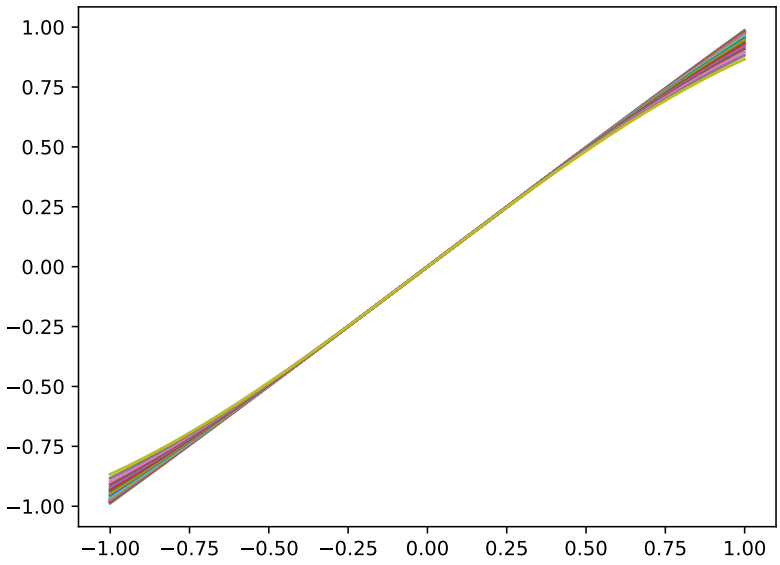


Figure 2.12: Simulated power amplifier nonlinearities for different devices.

Adding AWGN: After obtaining preamble signals with nonlinear features for 19 different devices, we create training, validation and test datasets by adding additive white Gaussian noise (AWGN) such that SNR = 20 dB for each dataset. For training, we use 200 signals per device from 19 devices. The validation and test sets contain 100 signals per device. Overall, the dataset contains 3800 signals for training, 1900 signals for validation and 1900 signals for the test set.

Results: We use the same CNN and training hyperparemeters as before, except for the number of epochs, which we set to 100. We observe trends similar to our results on emulation of "different days" with the measured WiFi data: model-based augmentation can significantly help improve performance when training over multiple emulated days

and testing on a different day. We report on these results in Table 2.7.

2.3 Implementation details

Network architecture details can be found in Table 2.8. We use a complex-valued ModReLU architecture similar to [18], and perform complex backpropagation using the framework of [61], taking partial derivatives of the cost with respect to the real and imaginary parts of each parameter. Networks are trained for 200 epochs with a batch size of 100, using the Adam optimizer with learning rate $\eta = 0.001$ and weight decay constant $\lambda = 0.0001$. We normalize all signals to unit power. For weight initialization, we use the complex-valued Glorot initialization from [61] for complex layers, and the real-valued Glorot [85] for real layers. For all experiments, we use Keras [86] with Theano backend, since complex-valued layers are implemented in Keras.

To assess performance, we have used the average of 5 different runs with different random seeds for initial weights and with different random realizations of CFOs and channels used for emulation and augmentation. In all the graphs in Section 2.2, error bars denote one standard deviation from the mean over different runs. We have also carried out 5-fold cross validation, where we use 5 different randomly chosen partitions of the data for training and testing, with the result that there is very little variation in performance.

Table 2.8: Architecture details for the CNN used in WiFi fingerprinting. Kernel sizes follow the notation [convolution size, input channels, output channels] for convolutional layers, and [input size, output size] for fully connected layers.

Layer	Kernel size	Bias size	Output shape
Complex Input	–	–	[3200, 1]
Complex Conv.	[200, 1, 100]	[100]	[31, 100]
ModRelu	–	[100]	[31, 100]
Complex Conv.	[10, 100, 100]	–	[22, 100]
ModRelu	–	[100]	[22, 100]
Absolute Value	–	–	[22, 100]
Real Fully Conn.	[100, 100]	[100]	[22, 100]
Real Fully Conn.	[100, 100]	[100]	[22, 100]
Global Avg. Pool	–	–	[100]
Real Fully Conn.	[100, 19]	[19]	[19]

Chapter 3

Robust Learning in a self-supervised setting

In this chapter, we study how to train a speaker recognition system utilizing an unlabeled dialogue dataset. We demonstrate the potential pitfalls caused by the dataset and how to handle them carefully. Section 3.1 presents the motivation along with a discussion of the prior work on self-supervised training of speech models. We then present our method in section 3.2. Finally, we present our empirical findings demonstrating the effectiveness of our proposed techniques in handling corrupt data while also teaching the model the features important to speaker identity.

3.1 Introduction

With the proliferation of mobile and smart devices in everyday life, the total amount of data created and captured has significantly increased. However, only a tiny fraction of this data finds its use in training the machine learning models. The facts that just a small portion of this data is labeled and that supervised learning require high-quality labeled

datasets are the major factors in this. All of these considered, it is not a big surprise that self-supervised learning gains significant attention from the deep-learning community [87, 88].

While self-supervised learning offers to employ unlabeled datasets, the lack of quality checks on data poses a problem for learning quality features. In this work, we take a look at this problem in a self-supervised speaker recognition setting. We propose to employ a large unlabeled yet well-structured dialogue dataset that contains human-machine dialogues from thousands of households. In a typical setting, a dialogue session is between a human speaker and a smart device. Therefore each utterance from a single dialogue corresponds to a data point with the same hypothetical label. While this assumption holds for a large part of the dialogues, there is a non-negligible amount of dialogues corrupt in some way. These include background human speakers, multiple speakers in multiple turns, and dialogues without intelligible human sounds. For example, table 3.1 gives an example of a good quality dialogue and a bad quality dialogue in our setting.

In this work, we propose two techniques to handle low-quality dialogues. First, we propose a novel metric learning loss function specifically designed to deal with multi-speaker dialogues. Secondly, we introduce a self-supervised rejection mechanism similar to the ones used in noisy data problem [89, 90]. We observe great benefits from both the loss function and rejection mechanism.

3.1.1 Related Work

In general, there are two types of self-supervised methods to pretrain speech models, namely, based on reconstruction or based on contrastive learning. For the former, one or a few consecutive frames are masked and then models are trained to reconstruct or predict the original features, such as APC [30], MockingJAY [31], DeCoAR [91] and HuBERT [33].

Table 3.1: Sample dialogues. Dialogue A corresponds to good quality dialogue containing a conversation of a single human speaker with a smart device. On the other hand, Dialogue B is a low-quality dialogue in which there are multiple human speakers.

Dialogue ID	Device type	Time	Source	Utterance
A	Google Home	2021-07-03 12:12:02	Customer	Hey Google, what's the weather like tomorrow?
		2021-07-03 12:12:12	Device	In New York city, it will be mostly sunny with the highest 77 and lowest 64.
		2021-07-03 12:12:20	Customer	Thanks Google.
		2021-07-03 12:12:27	Device	No problem.
B	Echo	2020-07-01 09:10:01	Customer	Alexa, add eggs to my shopping list.
		2020-07-01 09:10:08	Device	A dozen of eggs of organic large brown eggs have been added into your cart.
		2020-07-01 09:10:15	Customer	Alexa, I also want dark chocolate. Can I have that, Daddy?
		2020-07-01 09:10:25	Device	Sorry, I did not recognize your voice. Would you like to get enrolled?

As the masked features are reconstructed based on context, reconstruction-based methods are more suitable to speech recognition tasks and less effective on speaker recognition tasks. For the latter, positive and negative instances are constructed and models are optimized by conducting comparisons, which aim to group positive instances together while separating negative instances, such as COLA [28], CPC [34], and wav2vec [92]. Therefore, to effectively distinguish utterances from different speakers, contrastive learning methods are more appropriate.

3.2 Method

The unlabeled dialogue data is noisy because the customer utterances in the same dialogue can come from different speakers. It follows that the positive instances constructed by pairing utterances from the same dialogue are not reliable all the time. To alleviate this issue, we propose a new all-versus-all loss function and a rejection mechanism. Unlike angular prototypical loss [25] and GE2E loss [93], all-versus-all loss avoids using a centroid to represent a dialogue, but rather conducts comparisons for each utterance in a dialogue.

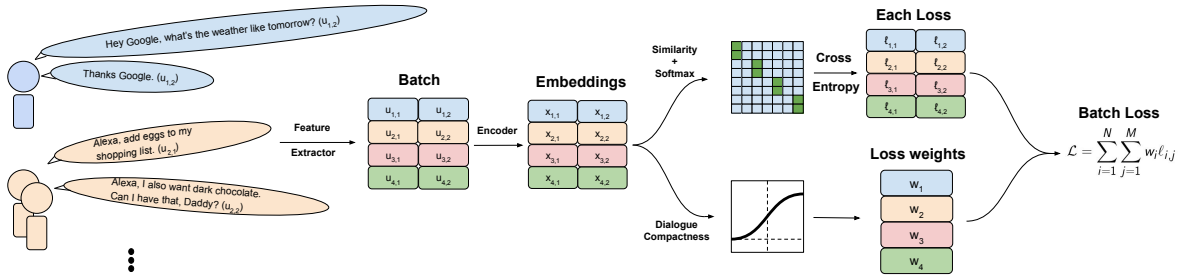


Figure 3.1: Each batch contains $M = 2$ utterances from $N = 4$ different dialogues, $M \cdot N$ utterances in total. We multiply the loss from each utterance depending on the compactness of the dialogue the utterance is extracted from.

In this way, the model will suffer less from centroids aggregating multiple speakers while learning from all utterances in a dialogue. In addition, the rejection mechanism guides the model in learning more from clean dialogues instead of noisy ones by weighting their loss contributions differently.

Figure 3.1 shows the proposed modeling framework. Given N dialogues ($N = 4$ in our case), we randomly sample M customer utterances per dialogue ($M = 2$ here) to construct a batch of $M \cdot N$ utterances. An encoder is employed to extract an embedding for each utterance in the batch. Then, a loss for each utterance is calculated based on its similarities to other utterances in the same dialogue and those in other dialogues, which are stored in a similarity matrix. At the same time, a compactness score is calculated for each dialogue, expressing the speaker purity of the dialogues. The overall batch loss is defined by a weighted sum of utterance losses considering the dialogue compactness scores.

3.2.1 All-versus-All Loss

The presence of the multiple speaker dialogues in the dataset causes the class centroids of GE2E loss to be flawed, as different speakers will have completely different embeddings.

Specifically, if there are multiple speakers in a dialogue, the negative pair centroids would not be ideal as the centroids are not reliable and the aggregation step leads to information loss, causing wrong gradient directions. In this work, we propose all-versus-all (AvA) loss function to alleviate the negative pair centroid problem. We compare the embedding to all the other embeddings without relying on centroids, not only avoiding the flawed centroid problem but also increasing the effective number of negative pairs. For positive pairs, we compare the query embedding with the centroid of the embeddings coming from the same dialogue, excluding the query utterance. The similarity between the utterances coming from same dialogue is promoted whereas the similarity between the utterances coming from different dialogues is penalized. If we form our batch with M utterances coming from N different dialogues, then the number of comparison pairs will be $M \cdot (N-1) + 1$ with only one of them being positive. GE2E and angular prototypical loss, on the other hand, have $N-1$ negative pairs and a single positive pair. [94] demonstrated that larger batch sizes, meaning more negative pairs, help improve the performance of self-supervised learning; therefore AvA gives a better loss function for our problem. Figure 3.2 visualizes all the loss functions considered and explains the main differences among them. Formally,

$$x_{i,j} = f(u_{i,j}) \tag{3.1}$$

$$s(x_{i,j}, x_{k,l}) = \frac{x_{i,j}^T x_{k,l}}{||x_{i,j}||_2 ||x_{k,l}||_2} \tag{3.2}$$

$$c_{i,j} = \frac{1}{M-1} \sum_{k=1, k\neq j}^{M} x_{i,k} \tag{3.3}$$

$$\ell_{i,j} = -\log \frac{e^{s(x_{i,j}, c_{i,j})}}{\sum_{k,l,k\neq i} e^{s(x_{i,j}, x_{k,l})} + e^{s(x_{i,j}, c_{i,j})}} \tag{3.4}$$

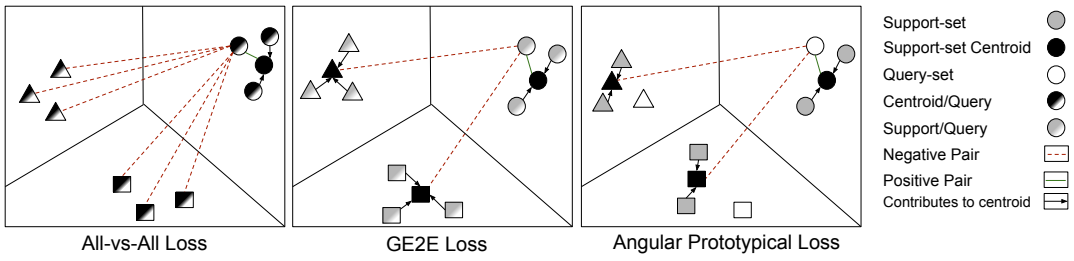


Figure 3.2: Comparison between the effect of all-versus-all loss (AvA), generalized end-to-end loss (GE2E), and angular prototypical loss (AP). Dashed lines represent distances encouraged to increase, while solid lines represent distances being decreased. Centroids denoted by black nodes are computed as the mean of the support set during training.

where f is an encoder model that produces embedding representation $x_{i,j}$ from an utterance $u_{i,j}$, where $u_{i,j}$ is the j-th utterance from dialogue i. s calculates the cosine similarity between two embeddings. $c_{i,j}$ is the positive centroid of a given utterance $u_{i,j}$ based on dialogue i. $\ell_{i,j}$ is the cross entropy loss for a given utterance $u_{i,j}$.

3.2.2 Self-supervised Rejection

Although AvA loss is solving part of the negative pair centroid problem, it does not offer a solution for the positive pair errors caused by the multiple speaker dialogues. Since the model will try to distinguish all the provided training data as well as possible, multi-speaker examples may push the neural network to learn some non-robust and SID-unrelated features. This is similar to the problem of noisy labels in supervised learning [89, 90]. We employ the idea of loss reweighing to decrease the contributions from noisy dialogues. Our method works on the fly, providing single-pass training with a significant performance improvement. We believe this framework also works for speaker recognition with datasets with noisy labels. Furthermore, our rejection mechanism works without significant additional computational cost since the similarities are already computed for

the loss functions.

Given the dialogue embeddings, we compute the average of the pairwise cosine similarities, which we call the *compactness* of the given dialogue. Then, we pass the compactness values through a sigmoid function with two hyper-parameters: temperature and threshold.

$$C_i = \frac{1}{M(M-1)} \sum_{j=1}^{M} \sum_{k=1, k \neq j}^{M} s(x_{i,j}, x_{i,k}) \tag{3.5}$$

$$w_i = \sigma(T * (C_i - t)) \tag{3.6}$$

where T is the temperature controlling the steepness of the sigmoid function, and t is a predefined threshold that determines the center of the sigmoid $\sigma(\cdot)$. We allow T to be learned by the model; however, we do not propagate derivatives through compactness C_i, letting it function simply as a scaling factor for the loss values.

Our final loss function becomes the weighted sum of the per-utterance losses:

$$\mathcal{L} = \sum_{i,j} w_i \cdot \ell_{i,j} \tag{3.7}$$

When the similarity between two utterances coming from the same dialogue is relatively small the weight of that particular dialogue will also be small. The idea is decreasing the loss contribution from multi-speaker dialogues, since their compactness will be much lower than that of single-speaker dialogues.

3.3 Experiments

Across all our experiments we employ a model consisting of a multi-layer unidirectional LSTM followed by a fully connected layer network. The dimensionality of each LSTM layer is 768, whereas the last linear layer has 256 units.

Table 3.2: Pretraining results. For each loss function, improvements relative to batch size 32 without rejection are shown.

Loss	Batch Size			
	32	64	128	256
All-vs-All	0.00%	+2.91%	+6.56%	**+8.20%**
Rejection + All-vs-All	**+3.76%**	+7.65%	+18.76%	**+19.00%**
A-Proto	0.00%	+7.32%	+8.52%	+12.93%
Rejection + A-proto	+7.55%	+12.58%	+16.76%	+25.85%
GE2E	0.00%	+3.24%	+3.06%	+6.36%
Rejection + GE2E	+10.64%	+17.75%	+17.99%	+13.83%

We pretrain our models on the AWS platform using 8 NVIDIA V100 GPUs with 16GB memory for 200 epochs. We employ the Adam optimizer with an initial learning rate of 0.0004, decreasing by 2% every 10000 iterations. For all of the experiments, we save the model giving the best validation EER value on a small subset of the labeled dataset.

The pretraining is conducted on deidentifed speech dialogues. The dataset is composed of 927,000 dialogues, comprising about 1800 hour of speech data. Since the number of dialogues is large, the chance of having multiple dialogues from the same speaker is very low per batch. As a dialogue contains at least two customer utterances, we form each batch by collecting two utterances from N different dialogues. We conduct our experiments using three different loss functions: GE2E, all-versus-all, and angular prototypical. Moreover, in order to investigate the effect of the rejection mechanism we conduct a number of

Table 3.3: Fine-tuning results. For all experiments we take the model trained from scratch as our baseline and report the relative improvement.

Pretraining	Loss	Episodes	Labeled Dataset Speaker Count			
			1,024	2,048	4,096	8,192
-	GE2E	1000	0.00%	0.00%	0.00%	0.00%
COLA	GE2E	300	-8.81%	-23.57%	-37.07%	-44.21%
APC	GE2E	300	+24.34%	+23.13%	+19.48%	+15.35%
VoxCeleb2	GE2E	300	+31.38%	+25.91%	+20.95%	+15.61%
Dialogue+AvA (ours)	GE2E	300	+40.18%	+34.19%	**+31.10%**	**+27.10%**
Dialogue+A-Proto (ours)	GE2E	300	**+41.28%**	**+34.77%**	+30.03%	+26.57%
Dialogue+GE2E (ours)	GE2E	300	+40.12%	+32.86%	+27.49%	+23.42%

experiments with varying batch sizes.

The evaluation dataset is constructed by first randomly sampling de-identified utterances from a year's traffic. Then each sampled utterance and the enrollment data of speakers are sent to multiple annotators to obtain ground-truth labels independently. To reduce annotation errors, we select utterances that have consistent annotation labels for the final evaluation dataset.

3.3.1 Model Performance with Rejection Mechanism

We first investigate how the rejection mechanism helps us learn from the noisy unlabeled dialogue data. Table 3.2 reports relative EER improvements by taking a batch size of 32, without using rejection, as a baseline.

There are two observations. First, the rejection mechanism helps improve EER performance on all three loss functions and different batch sizes. For example, when

all-versus-all loss is applied and the batch size is 32, we observe 3.76% relative EER improvement. This demonstrates the effectiveness of the rejection mechanism, helping the model focus on clean dialogues rather than noisy ones. Second, a large batch size also contributes to better EER performance, especially when the rejection mechanism is applied. For example, when all-versus-all loss is applied, the EER improves by 8.2% by increasing the batch size from 32 to 256. It improves by 19.0% if the rejection mechanism is involved. A large batch involves utterances from more dialogues and forces the model to learn harder tasks, distinguishing more speakers in a batch. This results in more accurate speaker recognition.

3.3.2 Model Performance before Fine-tuning

In this section, we investigate the performance of speaker recognition without fine-tuning the pretrained models. To compare with other self-supervised methods, we also pretrain a reconstruction-based APC model [30], and COLA [95] based on contrastive learning. As there are several public labeled datasets, we also train a supervised model based on the VoxCeleb2 dataset [26] to serve as an additional pretrained model. Here the supervised pretrained model based on the VoxCeleb2 dataset serves as the reference. In addition, we further train four fully supervised models based on labeled Alexa datasets with varying number of speakers.

We highlight three observations based on Table 3.4. First, we note that the pretrained models COLA and APC are worse than the supervised model trained on the VoxCeleb2 dataset. These two methods aim to learn general audio features and they strongly depend on fine-tuning steps in order to achieve comparable performance for a downstream task. Therefore, they perform poorly on speaker recognition task without fine-tuning. Second, the proposed model and its variants consistently outperform the reference model trained

Table 3.4: Comparison of pretrained models, our method outperforms reference model trained on the VoxCeleb2 labeled dataset without fine-tuning. Its performance is even comparable to fully supervised models trained on labeled Alexa datasets.

Training Data	Method type	Loss	EER
Alexa Dialogue	Self-supervised	COLA	-129.56%
Alexa Dialogue	Self-supervised	APC	-108.32%
VoxCeleb2	Supervised	GE2E	0%
Alexa (1024 spk)	Supervised	GE2E	+12.75%
Alexa (2048 spk)	Supervised	GE2E	+27.11%
Alexa (4096 spk)	Supervised	GE2E	+**34.79**%
Alexa (8192 spk)	Supervised	GE2E	+39.17%
Alexa Dialogue	Self-supervised	AvA	+28.81%
Alexa Dialogue	Self-supervised	A-Proto	+**30.84**%
Alexa Dialogue	Self-supervised	GE2E	+28.49%

on the VoxCeleb2 labeled dataset, with EER reduced by as much as 30.84% relative. This clearly demonstrates the effectiveness of the proposed model in exploiting implicit speaker information in human-machine dialogues. The utilization of Alexa human-machine dialogues helps us overcome the domain mismatch between Alexa users and speech from other sources, such as the YouTube excerpts assembled in VoxCeleb. Third, the proposed model achieves EER reductions comparable to the models trained from scratch on Alexa labeled datasets. For example, our best performing model achieves 30.84% EER reduction while the fully supervised model trained on the Alexa labeled 4096-speaker dataset achieves 34.79% reduction. This shows that the proposed model trained with unlabeled dialogue data is effective in learning speaker identity features.

3.3.3 Model Performance after Fine-tuning

We fine-tune the pretrained network on different labeled Alexa datasets with varying number of speakers, where the total utterance duration for a speaker is around 150 seconds on average. All fine-tuning results based on the various pretrained models are summarized in Table 3.3. Here the four models trained from scratch with 1024, 2048, 4096, and 8192 labeled speakers serve as the reference baselines. Due to limited space, we show the fine-tuned model performance for GE2E loss only.

There are four key observations. First, the pretrained COLA model is not effective at learning speaker identities on the dialogue data, as we observe performance drop compared to the model trained from scratch for all four fine-tuning datasets. The utterances in dialogues are very short (one to two seconds duration). COLA further separates each utterance into two segments in order to form positive instances. Moreover, the background environment tends to be identical within the same utterance. Without massive and effective data augmentations, COLA tends to perform poorly on speaker recognition tasks.

Second, we notice that the pretrained APC model [96] helps improve the recognition performance with fine-tuning. For example, compared with the model trained with 1024 speakers from scratch, fine-tuning the APC model with the same labeled dataset improves EER by 24.34%.

Third, fine-tuning the supervised model pretrained on the VoxCeleb2 dataset also helps improve the EER performance, in spite of the domain mismatch between VoxCeleb2 (YouTube recordings) and Alexa traffic. We observe 31.38% relative EER improvement when the model is fine-tuned with 1,024 speakers.

Fourth, the proposed method achieves the largest relative EER improvements on all four fine-tuning datasets compared to COLA, APC, and the supervised model trained

on the VoxCeleb2 dataset. The best results are highlighted in bold in Table 3.3. This demonstrates the superiority of the proposed method for our speaker recognition scenario, learning to distinguish speakers by selectively learning from the unlabeled human-machine dialogues.

Chapter 4

A Neuro-inspired Approach To Robustness

This chapter discusses how incorporating appropriately engineered neuro-inspired principles into DNN architectures and training yields promising results to combat ℓ_p bounded adversarial perturbations and common corruptions. We provide the background in adversarial attacks and common corruptions in section 4.1 along with related work in adversarial machine learning. We then present the details of our neuro-inspired Hebbian/anti-Hebbian framework to enhance the robustness of the deep neural networks in section 4.4. We provide another perspective to evaluate the robustness of a model in section 4.5. Finally, we show our framework's effectiveness against various corruptions and adversarial attacks, supported by extensive experimental results in section 4.5.

4.1 Background

Since their original breakthrough in image classification performance, DNNs trained with backpropagation have attained outstanding performance in a wide variety of fields

[8–10, 97]. However, fundamental concerns remain regarding the lack of robustness in DNNs (e.g, to adversarial perturbations and common corruptions).

While deep learning models accurately classify natural data points, they suffer when encountering a carefully designed adversarial example. In traditional DNNs, even a slight change in the input image gets amplified, propagating through the layers and eventually causing a misclassification. A game of cat and mouse between attackers and defenders was set in motion after the susceptible nature of deep neural networks was first pointed out by [16], and [17]. We now detail state-of-the-art adversarial attacks.

4.1.1 Adversarial Attacks

In this work, we only report on inference time attacks, also called evasion attacks [16]. The goal of evasion attacks is to change the classification outcome of DNN with the smallest possible perturbation. In general, some constraints limit adversarial perturbations to avoid changing the input's ground truth label (human prediction). The most commonly used constraints for adversarial attacks are ℓ_p distance measures, ℓ_∞ being the most popular amongst them. Furthermore, attacks can be grouped into two classes, whether they have access to the inner structure and the parameters of the model or not: respectively, white-box attacks and black-box attacks. Since white-box attacks are stronger than black-box ones by definition, we limit our focus to white-box attacks, specifically gradient-based attacks. Gradient-based attacks, including the fast gradient sign method (FGSM) [98], the iterative version of FGSM known as the basic iterative method [99], employ the normalized gradients with respect to the inputs to generate the perturbation. These methods are also referred to as projected gradient descent (PGD) [37], a well-known technique to solve a constrained optimization problem. Formally, ℓ_∞ bounded PGD computes the adversarial perturbation as follows:

$$\boldsymbol{e}_{i+1} = \text{clip}_{\epsilon}\big[\boldsymbol{e}_i + \delta \cdot \text{sign}(\nabla_{\mathbf{e}}\mathcal{L}(\boldsymbol{f}(\boldsymbol{x} + \boldsymbol{e}_i), \boldsymbol{y}))\big] \tag{4.1}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ correspond to input and label respectively, $\boldsymbol{e_i}$ corresponds to the perturbation at iteration i, ϵ corresponds to the attack budget, δ is step size, $\mathcal{L}$ corresponds to the loss function, and $\boldsymbol{f}$ corresponds to the neural network. Most recently, an ensemble of parameter-free versions of PGD and state-of-the-art black-box attacks are benchmarked as AutoAttack [100] to evaluate the empirical robustness of a defense reliably. In this work, we use AutoAttack to evaluate the robustness of our framework.

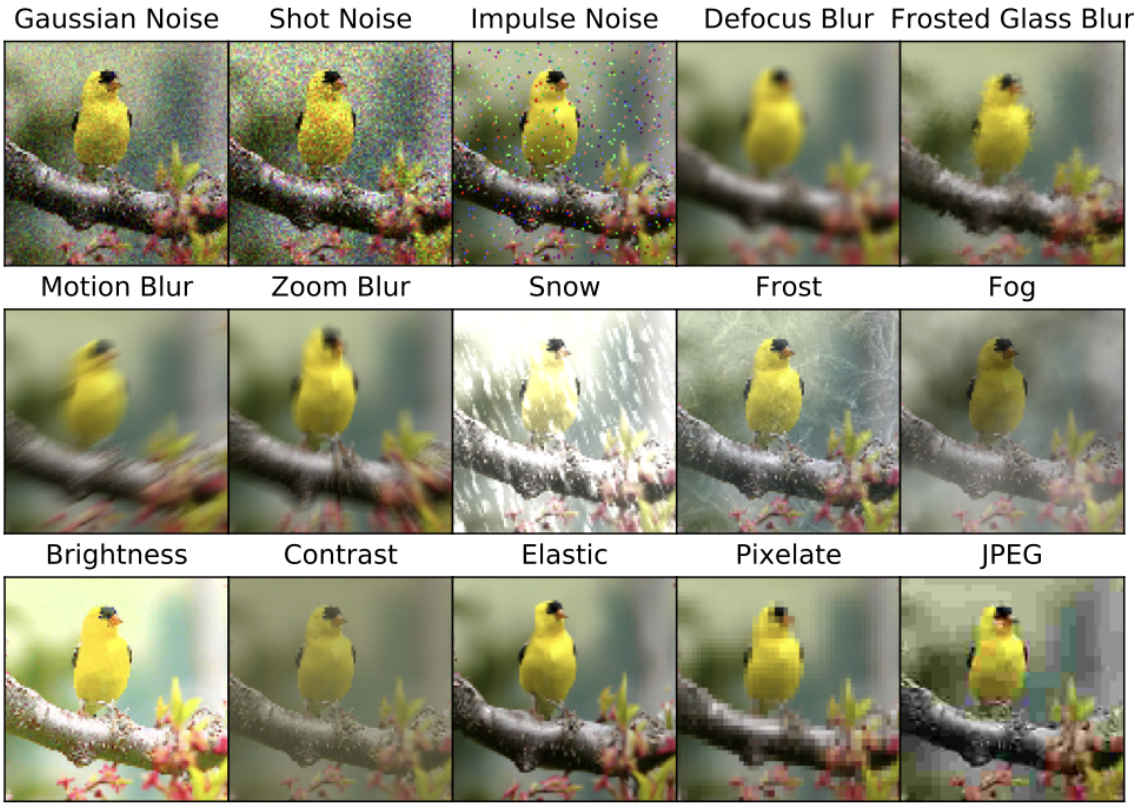


Figure 4.1: Common corruptions suggested by [4]. Figure is taken from [4]. Although these corruptions change the outcome of the most successful deep neural networks, humans are not confused with these corruptions.

4.1.2 Common Corruptions

The vulnerability of deep neural networks are not limited to adversarial examples; [4] showed that common corruptions that does not affect the decision of human vision system affects the DNNs' outcomes significantly. Figure 4.1 shows these corruptions that include noise, blur, weather, and digital corruptions all of which are not uncommon in real life. Therefore, getting robust against these corruptions are also essential. Here in this work, we aim to get robust against both adversarial attacks and common corruptions unlike the other defense techniques against adversarial examples which aim to be robust against a specific attack type.

4.2 Approach and Contributions

Conventional top-down training employs a cost function based on the DNN output, thus providing little insight and no guarantees on the features extracted by the layers of the DNN. The use of the resulting "black box" DNNs in many safety- and security-critical applications is blocked by concerns about their lack of interpretability and robustness. While data augmentation has been shown to enhance robustness to some extent (e.g, the use of adversarial examples generated on the fly during adversarial training), it is only a partial solution. In this work, we explore the thesis that a first step to alleviating these problems is to exert more control on the features being extracted by DNNs. We develop a training framework aimed at shaping the features generated by a DNN layer, by supplementing end-to-end costs with costs that depend on the activations at each layer. We seek to generate sparse activation patterns with a small fraction of large activations, instead of the large proportion of small activations produced by a standard DNN.

In order to attain sparse, strong activations at each layer, we employ the following neuro-inspired strategy for modifying standard DNN training and architecture:

Hebbian/anti-Hebbian (HaH) Training: We supplement a standard end-to-end discriminative cost function with layer-wise costs at each layer which promote neurons producing large activations and demote neurons producing smaller activations. The goal is to develop a neuronal basis that produces a distributed sparse code, without requiring a reconstruction cost as in standard sparse coding [101].

Neuronal Competition via Normalization: We further increase sparsity by introducing Divisive Normalization (DN), which enables larger activations to suppress smaller activations. In order to maintain a fair competition among neurons, we introduce Implicit ℓ_2 Normalization of the neuronal weights, so that each activation may be viewed as a geometric projection of the layer input onto the "direction" of the neuron. (Using implicit rather than explicit weight normalization in our inference architecture simplifies training.)

We report on experiments with CIFAR-10 image classification, comparing a baseline VGG-16 network trained end-to-end against the same architecture with HaH training and DN. Both architectures employ implicit weight normalization, which we have verified does not adversely impact accuracy. We demonstrate that the activations in our proposed architecture are indeed more sparse than for the baseline network. In order to isolate the impact of our training approach and inference architecture, we do not employ noise augmentation or adversarial training in these initial experiments. For CIFAR10 classification, we show that our model is significantly more robust than a baseline model against both noise and adversarial perturbations. Against the broader set of corruptions in the CIFAR10-C dataset (Common corruptions dataset), our model is generally more resilient than both the baseline model and an adversarially trained model.

4.3 Related Work

Hebbian learning has a rich history in artificial neural networks, dating back to the neocognitron [102], and including recent attempts at introducing it into deep architectures [103]. However, to the best of our knowledge, ours is the first paper to clearly demonstrate gains in robustness from its incorporation in DNNs. Divisive normalization is a widely accepted concept in neuroscience [104, 105], and versions of it have been shown to be competitive with other normalization techniques in deep networks [106]. Our novel contribution is in showing that divisive normalization can be engineered to enhance sparsity and robustness. Finally, sparse coding with a reconstruction objective was shown to lead to neuro-plausible outcomes in a groundbreaking paper decades ago [101]. In contrast to the iterative sparse coding and dictionary learning in such an approach, our HaH-based training targets strong sparse activations in a manner amenable to standard stochastic gradient training.

Recent work showing potential robustness gains by directly including known aspects of mammalian vision in DNNs includes [107], which employs Gabor filter blocks and stochasticity, and [108], which employs neural activity measurements from mice for regularization in DNNs. Rather than incorporating specific features from biological vision, we use neuro-inspiration to extract broad principles that can be folded into data-driven learning and inference in DNNs.

4.4 Model

We now describe how we incorporate HaH training and divisive normalization into a standard CNN for image classification. We consider a "classical" CNN for our experiments– VGG-16 [109] applied to CIFAR-10, rather than variants of ResNet [110], because residual

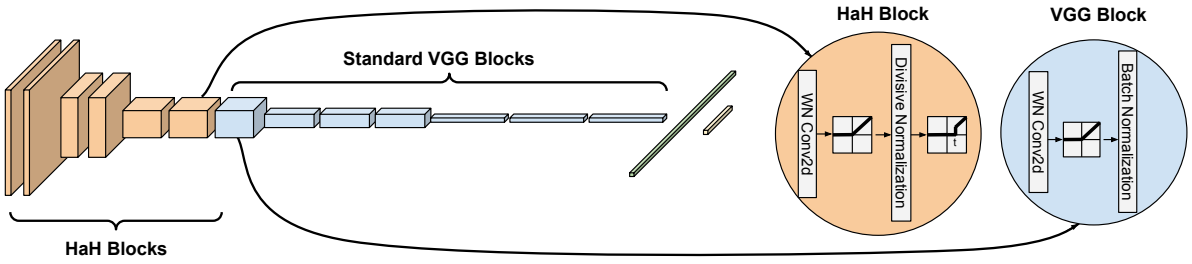


Figure 4.2: Our model consists of two different types of blocks: first 6 blocks are Hebbian-anti-Hebbian (HaH) while the rest are regular VGG blocks. HaH blocks use a weight normalized convolutional layer, followed by ReLU, divisive normalization and thresholding. Regular VGG blocks use a weight normalized convolutional layer followed by ReLU and batch norm.

connections complicate our interpretation of building models from the bottom-up using HaH learning. Since we wish to build robustness from the bottom up, we modify the first few convolutional blocks to incorporate neuro-inspired principles. We term these modified blocks "HaH blocks."

Each HaH block employs convolution with implicit weight normalization, followed by ReLU, then divisive normalization, and then thresholding. Implicit weight normalization enables us to interpret the convolution outputs for each filter as projections, and we have verified that employing it in all blocks of a baseline VGG-16 architecture does not adversely impact accuracy (indeed, it slightly improves it). Each standard (non-HaH) block in our architecture therefore also employs convolution with implicit weight normalization, followed by ReLU, but uses batch norm rather than divisive normalization. Each HaH block contributes a HaH cost for training, so that the overall cost function used for training is the standard discriminative cost and the sum of the HaH costs from the HaH blocks.

We now describe the key components of our architecture, shown in Figure 4.2.

4.4.1 Inference in a HaH block

Implicit weight normalization: Representing the convolution output at a given spatial location from a given filter as a tensor inner product $\langle \cdot, \cdot \rangle$ between the filter weights $\boldsymbol{w}$ and the input $\boldsymbol{x}$, the output of the ReLU unit following the filter is given by

$$y = \text{ReLU}\left(\frac{\langle \boldsymbol{w}, \boldsymbol{x} \rangle}{||\boldsymbol{w}||_2}\right) \tag{4.2}$$

This effectively normalizes the weight tensor of each filter to unit ℓ_2 norm, without actually having to enforce an ℓ_2 norm constraint in the cost.

Divisive normalization: If we have N filters in a given HaH block, let $y_1(loc), ..., y_N(loc)$ denote the corresponding activations computed as in (Equation 4.2) for a given spatial location loc. Let $M(loc) = \frac{1}{N}\sum_{k=1}^{N} y_k(loc)$ denote the mean of the activations at a given location, and let $M_{max} = \max_{loc} M(loc)$ denote the maximum of this mean over all locations. We normalize each activation using these terms as follows:

$$z_k(loc) = \frac{y_k(loc)}{\sigma M_{max} + (1-\sigma)M(loc)} \ , \ k = 1, ..., N \tag{4.3}$$

where $0 \leq \sigma \leq 1$ is a hyperparameter which can be separately tuned for each HaH block. Thus, in addition to creating competition among neurons at a given location by dividing by $M(loc)$, we also include M_{max} in the denominator in order to suppress contributions at locations for which the input is "noise" rather than a strong enough "signal" well-aligned with one or more of the filters. This particular implementation of divisive normalization ensures that the output of a HaH-block is scale-invariant (i.e., we get the same output if we scale the input to the block by any positive scalar).

Adaptive Thresholding: Finally, we ensure that each neuron is producing significant outputs by neuron-specific thresholding after divisive normalization. The output of the

kth neuron at location loc is given by

$$o_k(loc) = \begin{cases} z_k(loc) & \text{if } z_k(loc) \geq \tau_k \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

where the threshold τ_k is neuron and image specific. For example, we may set τ_k to the 90th percentile of the statistics of $z_k(loc)$ in order to get an activation rate of 10% for each neuron for every image. Another simple choice that works well, but gives higher activation rates, is to set τ_k to the mean of $z_k(loc)$ for each image.

4.4.2 HaH Training

For an N-neuron HaH block with activations $y_k(loc)$, $k = 1, ..., N$ at location loc, the Hebbian/anti-Hebbian cost seeks to maximize the average of the top K activations, and to minimize the average of the remaining $N - K$ activations, where K is a hyperparameter. Thus, sorting the activations $\{y_k(loc)\}$ so that $y^{(1)}(loc) \geq y^{(2)}(loc) \geq ... \geq y^{(N)}(loc)$, the

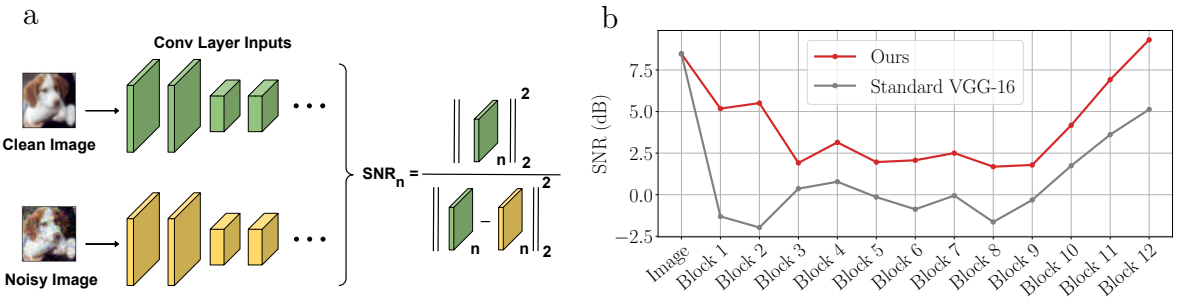


Figure 3: **a**: To compute the SNR at the n^{th} block inputs, we divide the ℓ_2 norm of the block input corresponding to clean image by the ℓ_2 norm of the difference of block corresponding to clean and noisy images. **b**: Comparison of SNR values of the block inputs for the standard base model (gray) and ours (red).

contribution to the HaH cost (to be maximized) is given by

$$L_{block}(loc) = \frac{1}{K}\sum_{k=1}^{K} y^{(k)}(loc) - \lambda \frac{1}{N-K}\sum_{k=K+1}^{N} y^{(k)}(loc) \tag{4.5}$$

where $\lambda \geq 0$ is a hyperparameter determining how much to emphasize the anti-Hebbian component of the adaptation. The overall HaH cost for the block, L_{block}, which we wish to maximize, is simply the mean over all locations and images.

The overall loss function to be minimized is now given by

$$L = L_{disc} - \sum_{\text{HaH blocks}} \alpha_{block} L_{block} \tag{4.6}$$

where L_{disc} is the standard discriminative loss, and $\{\alpha_{block} \geq 0\}$ are hyper-parameters determining the relative weight of the HaH costs across blocks.

4.4.3 Insight into HaH updates

In this section, we provide quick analytical and geometric insight into the HaH framework. By rewarding large activations y, the HaH cost targets learning weight tensors more aligned with input tensors. To see this, consider an activation $y = \langle \boldsymbol{w}, \boldsymbol{x} \rangle / ||\boldsymbol{w}||_2$ which among the top K, and is therefore receiving a Hebbian update. The gradient along which $\boldsymbol{w}$ is to be updated can be computed as

$$\frac{\partial y}{\partial \boldsymbol{w}} = \frac{\langle \boldsymbol{w}, \boldsymbol{w} \rangle \boldsymbol{x} - \langle \boldsymbol{w}, \boldsymbol{x} \rangle \boldsymbol{w}}{||\boldsymbol{w}||_2^3} = \frac{\mathcal{P}_{\boldsymbol{w}}^{\perp} \boldsymbol{x}}{||\boldsymbol{w}||_2} \tag{4.7}$$

where $\mathcal{P}_{\boldsymbol{w}}^{\perp} \boldsymbol{x}$ denotes the projection of the input $\boldsymbol{x}$ orthogonal to the one-dimensional subspace spanned by $\boldsymbol{w}$. The update $\Delta \boldsymbol{w} = \eta \frac{\partial y}{\partial \boldsymbol{w}}$ is therefore proportional to this orthogonal component, and moving in this direction reduces the angle between $\boldsymbol{w}$ and $\boldsymbol{x}$, provided that η is small enough. We skip details due to lack of space, but note the following geometric interpretation. Because of implicit normalization, the original activation can be written as

$$y = \langle \boldsymbol{w}, \boldsymbol{x} \rangle / ||\boldsymbol{w}||_2 = ||\boldsymbol{x}||_2 \cos\theta \tag{4.8}$$

where θ is the angle between $\boldsymbol{w}$ and $\boldsymbol{x}$. By reducing θ via the weight update, we increase the implicitly normalized activation. Thus,

$$y_{new} = \langle \boldsymbol{w} + \Delta \boldsymbol{w}, \boldsymbol{x} \rangle / ||\boldsymbol{w} + \Delta \boldsymbol{w}||_2 > y_{old} = \langle \boldsymbol{w}, \boldsymbol{x} \rangle / ||\boldsymbol{w}||_2 \tag{4.9}$$

Note that exactly the opposite phenomenon occurs for an anti-Hebbian update: those weight vectors become less aligned with the input.

This procedure trains the neurons such that, during inference, the highly activated neurons at a given location tend to be better aligned with the input. Not only does this make these top activations more resilient to the impact of noise or perturbations, but larger activations also help attenuate the impact of noise on smaller activations by virtue of divisive normalization. In fact, many of these smaller noisy activations get eliminated via the thresholding applied after divisive normalization.

4.5 Experiments

We consider VGG-16 with the first 6 blocks (each block includes conv, ReLU, batch norm) replaced by HaH blocks (each block includes conv, ReLU, divisive norm, thresholding). In our training, we use Adam optimizer [111] with an initial learning rate of 10^{-3}, multiplied by 0.1 at epoch 60 and again at epoch 80. We train all models for 100 epochs on CIFAR-10. We choose τ_k in Equation 4.4 to keep 20% of activations. We use $[4.5 \times 10^{-3}, 2.5 \times 10^{-3}, 1.3 \times 10^{-3}, 1 \times 10^{-3}, 8 \times 10^{-4}, 5 \times 10^{-4}]$ for α in Equation 4.6. We use 0.1 for λ and set K to 10% of number of filters in each layer in Equation 4.5 and set $\sigma = 0.1$ in Equation 4.3. Details about other hyper-parameters can be found in our code in supplementary materials.

Sparser activations: To ensure that HaH blocks are operating as intended and achieving the sparse and strong activations we test the sparsity levels of intermediate representations

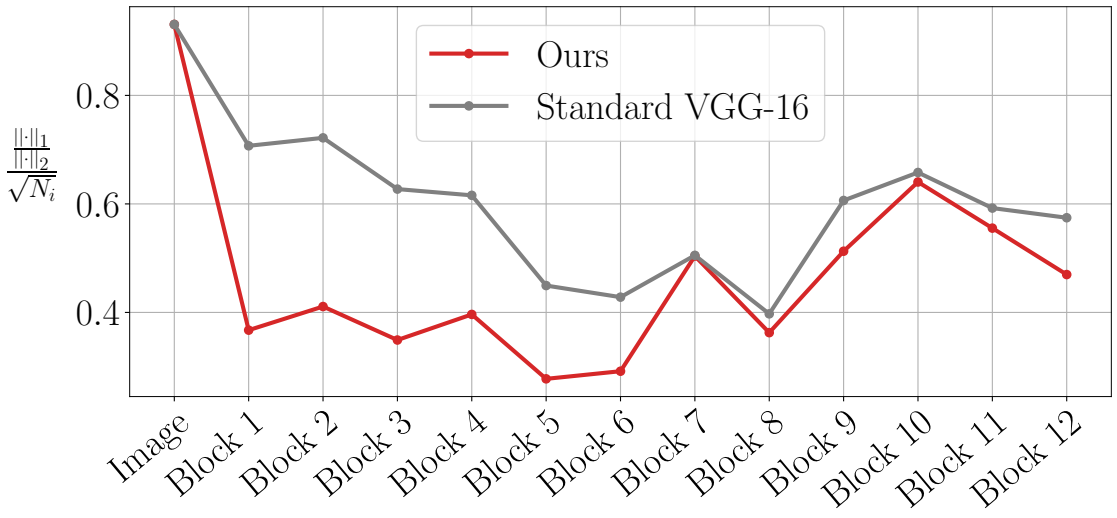


Figure 2: HaH blocks yield sparser activations than baseline. The measure of sparsity is the Hoyer ratio [5] of ℓ_1 norm to ℓ_2 norm of activations across channels, averaged across spatial locations, and then normalized to lie in [0,1] (lower values correspond to more sparsity).

and plot them in Figure 2. Sparsity is computed by the ratio of ℓ_1 norm to ℓ_2 norm (also known as Hoyer term [5]) of each spatial location's representation across the channel dimension. We then linearly normalize the values to lie in [0,1]. Lower values represent sparser representations. The activations in these first 6 blocks are indeed more sparse for our architecture than for baseline VGG.

Enhanced robustness to noise: We borrow the concept of signal-to-noise-ratio (SNR) from wireless communication to obtain a block-wise measure of robustness. Let $f_n(x)$ denote the input tensor at block n in response to clean image x, and $f_n(x+w)$ the input tensor when the image is corrupted by noise w. As illustrated in Figure 3a, we define SNR as

$$\text{SNR}_n = 10\log_{10}\left(\mathbb{E}_{x\sim\mathcal{D}_{test}}\left[\frac{||f_n(x)||_2^2}{||f_n(x+w)-f_n(x)||_2^2}\right]\right)dB \quad (4.10)$$

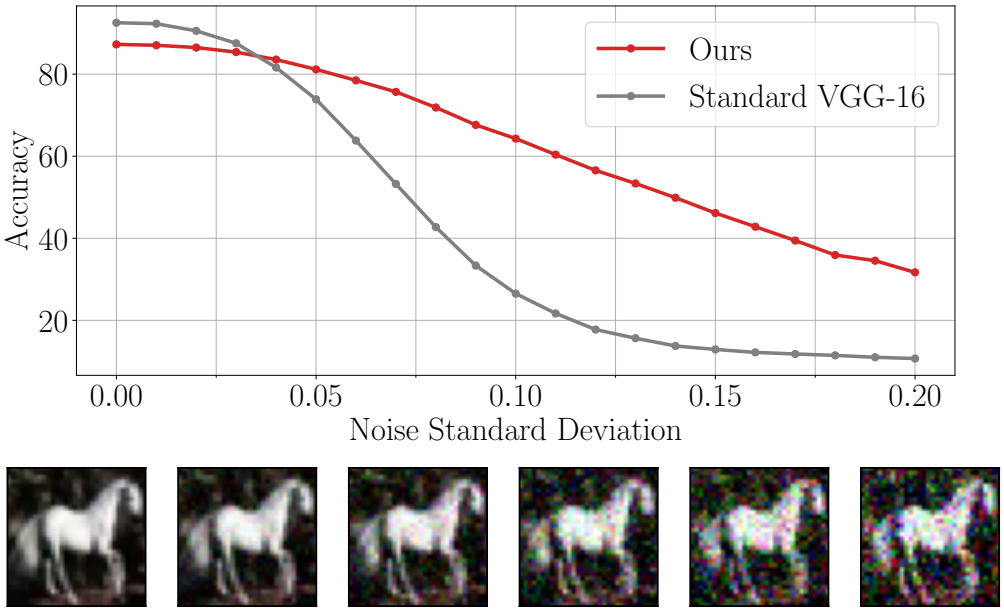


Figure 4: Comparison of classification accuracies as a function of noise σ. To provide a concrete sense of the impact of noise, noisy images at increasing values of σ are shown below the graph.

converting to logarithmic decibel (dB) scale as is common practice. Figure 3b shows that the SNR for our model comfortably exceeds that of the standard model, especially in the first 6 HaH blocks.

These higher SNR values also translate to gains in accuracy with noisy images: Figure 4 compares the accuracy of our model and the base model for different levels of Gaussian noise. There are substantial accuracy gains at high noise levels: 64% vs. 26% at a noise standard deviation of 0.1, for example.

Enhanced robustness to adversarial attacks: While we have not trained with adversarial examples, we find that, as expected, the noise rejection capabilities of the HaH blocks also translates into gains in adversarial robustness relative to the baseline VGG model. This holds for state-of-the-art gradient-based attacks [37, 39], as well as AutoAttack, an ensemble of parameter-free attacks suggested by RobustBench [100].

We observe no additional benefit of using gradient-free attacks, and conclude that the robustness provided by our scheme is not because of gradient-masking. Because of space constraints, we only report on results from minimum-norm adversarial attacks and AutoAttack.

Figure 5 shows that the minimum distortion needed to flip the prediction of our model (computed using the recently proposed fast minimum norm computation method [39]) is higher for our model for all the ℓ_p attacks considered.

We have also obtained substantial gains in adversarial accuracy against all four ℓ_p norm attacks ($p = 0, 1, 2, \infty$) used as benchmarks in adversarial machine learning. Table 1 displays a subset of results demonstrating accuracy gains against noise and adversarial perturbations, at the expense of a slight decrease in clean accuracy.

Enhanced robustness to common corruptions: Finally, we evaluate our neuro-inspired framework for common corruptions suggested by [112]. These corruptions include noise injection, weather condition, common blur, and digital corruptions. Table 2 compares

Corruptions →	Clean	Noise				Weather				Blur				Digital				Mean
Models ↓		Gauss.	Shot	Speckle	Impulse	Snow	Fog	Frost	Bright.	Defocus	Gauss.	Motion	Zoom	Contrast	Elastic	Pixelate	Spatter	of all
Standard	**92.5**	32.4	40.0	45.5	27.5	72.9	**64.5**	61.6	**87.4**	45.5	34.8	59.7	58.9	23.0	**74.8**	51.0	68.6	53.0
Adv(8/255)	78.7	**74.2**	**74.4**	**73.4**	**62.9**	62.3	29.7	59.0	60.4	**69.8**	**67.5**	**67.2**	**72.0**	18.0	72.5	**75.4**	71.5	63.1
HaH (Ours)	87.3	64.7	63.9	61.2	50.2	**74.4**	63.3	**73.3**	83.3	65.9	59.9	65.8	69.5	**76.3**	73.8	62.1	**76.3**	**67.7**

Table 2: Common corruption accuracies across different models. While standard and adversarially trained models are VGG16, HaH (ours) uses the aforementioned modified version of VGG16. Adversarially trained models perform poorly on fog and contrast corruptions while excelling on high-frequency corruptions like noise. On the other hand, the HaH framework consistently improves the robustness against all sorts of corruptions. Bright. stands for brightness, Gauss. stands for Gaussian, Elastic stands for elastic transformation

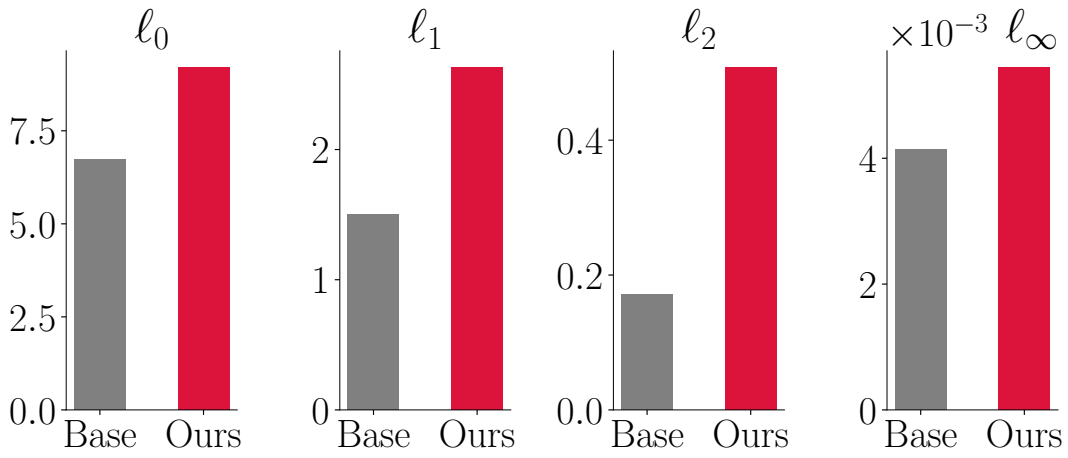


Figure 5: The average norm of minimum-norm adversarial attacks is higher for our model for all ℓ_p norms considered.

the accuracies obtained by our model with those for a standard model and an adversarially trained model. We see that our neuro-inspired design is effective in increasing robustness against these common corruptions. It is worth noting that, while adversarially trained models perform well against noise type corruptions, they perform drastically worse against more complex corruptions like fog and contrast [113, 114]. In contrast, our HaH framework not only performs relatively well (performing substantially better than the standard model) for noise corruptions but also performs significantly better on more complex corruptions such as fog and contrast. Furthermore, the HaH-VGG16 outperforms both the standard model and adversarially trained model in terms of mean corruption accuracy. Given that such corruptions barely affect human vision, these results indicate that neuro-inspiration provides a valuable path towards general-purpose robustness against noise, adversarial perturbations, and common corruptions.

Ablation: Since we have different components in our HaH blocks, we explore the effectiveness of each component by doing an ablation study. Table 3 summarizes the contribution from each of the components. We see that all of the components (HaH

Table 1: Enhanced accuracy against noise and adversarial attacks.

	Clean	Noisy ($\sigma = 0.1$)	Adv (ℓ_∞) ($\epsilon = 2/255$)	Adv (ℓ_2) ($\epsilon = 0.25$)
Standard	**92.5%**	26.6%	10.4%	13.9%
Ours	87.3%	**64.0%**	**21.5%**	**27.6%**

training, divisive normalization, adaptive thresholding) play an important role in obtaining the reported gains in robustness to noise and adversarial attacks.

Furthermore, the number of HaH blocks plays a crucial role in obtaining robustness.

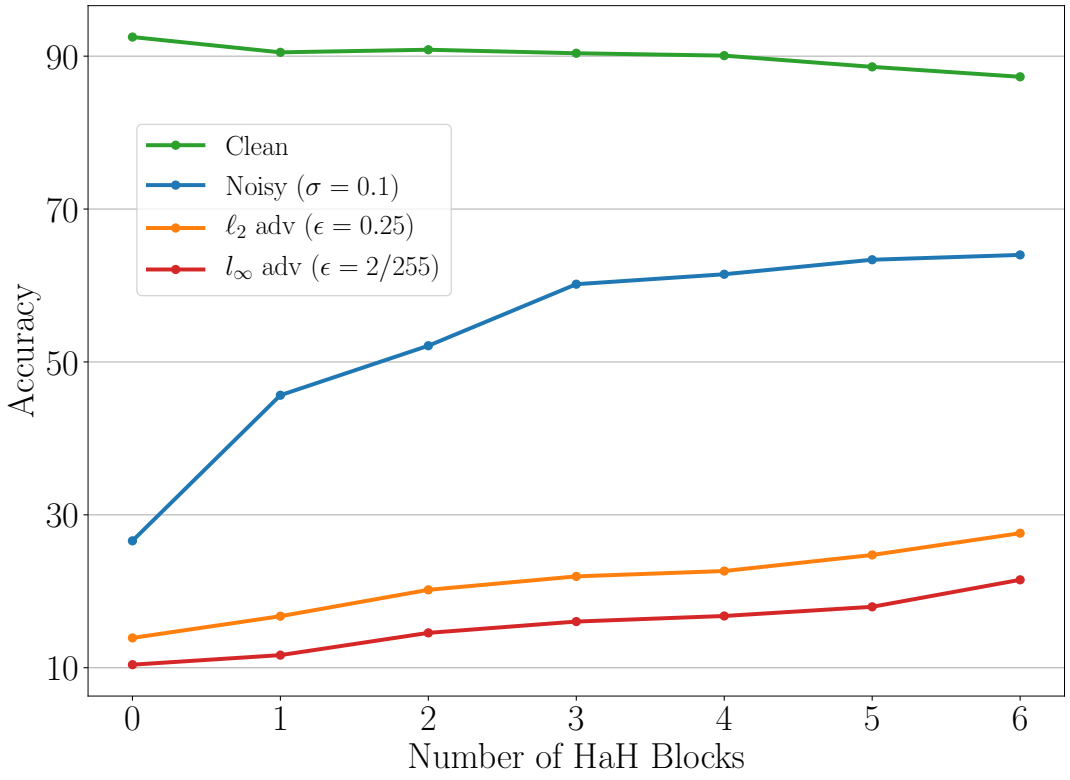


Figure 6: Ablation study for number of HaH blocks. Every additional HaH block contributes to the robustness of the model with a slight compromise on clean accuracy.

Table 3: Accuracies for ablation study.

	Clean	Noisy ($\sigma = 0.1$)	Adv (ℓ_∞) ($\epsilon = 2/255$)	Adv (ℓ_2) ($\epsilon = 0.25$)
All included	87.3%	**64.0%**	**21.5%**	**27.6%**
No HaH loss	89.7%	50.4%	8.8%	11.7%
Batch norm instead of divisive norm	**90.4%**	46.7%	12.3%	17.4%
No thresholding	89.9%	37.5%	3.7%	2.5%

Figure 6 shows the trade-off between clean accuracy and robust accuracy when the number of HaH blocks changes. Note that we successfully trained a model at most with 6 HaH blocks. Like earlier bio-inspired defenses, robustness through the HaH blocks also comes with a slight compromise on clean accuracy.

4.6 Software Release

We release our framework as a software toolbox to motivate the community to explore these ideas further. Table 4 shows the metadata for our software framework. We implement our ideas via a publicly available extension module to Pytorch [115] called HaH (Hebbian/Anti-Hebbian): a neuro-inspired DNN toolbox, with aforementioned new components such as Cost, Regularizer, Divisive normalizer, and Threshold.

We also provide additional utility functions to extract and use the layer outputs. We provide a wrapper class for *torch.nn.Module* which utilizes forward hooks from PyTorch to extract all the outputs and inputs of a specific layer type from a DNN while training.

C1	Current code version	v0.0.5
C2	Permanent link to code/repository used for this code version	https://github.com/metehancekic/HaH
C3	Permanent link to Reproducible Capsule	https://codeocean.com/capsule/0731065/tree/v1
C4	Legal Code License	MIT License
C5	Code versioning system used	Git
C6	Software code languages, tools, and services used	Python
C7	Compilation requirements, operating environments & dependencies	Python >= 3.8.2 with the following dependencies PyTorch 1.10.2; Numpy 1.19.2
C8	If available Link to developer documentation/manual	https://github.com/metehancekic/HaH
C9	Support email for questions	metehancekic@ucsb.edu

Table 4: Software metadata

The wrapper expects a model and the layer type and makes the layer outputs and inputs accessible whenever an input is fed to the neural network.

Chapter 5

Conclusions and Future Work

One of the promises of deep neural networks is to supersede domain expertise by extracting features automatically and learning complicated tasks from extensive data. State-of-the-art DNNs deliver this promise; however, they are far less stable and robust than the traditional systems they replace. They operate like large statistical machines looking for the strongest features that explain the data fed to them, ignoring whether they are spurious correlations or not; therefore, their performance and robustness heavily depend on the training data. In this thesis, through different case studies, we tackle this problem by proposing domain-specific measures to improve their stability and robustness. Our main point is that the impetuous employment of DNNs might lead to disastrous security, stability, and performance issues. In particular, we demonstrate the importance of being proactive in identifying and preventing potential problems due to the data and given task. Now we detail our concluding remarks and the future directions.

5.1 Robust RF Fingerprinting

While complex-valued CNNs are a promising tool for learning RF signatures, we conclude that blind adoption of these networks is dangerous due to confounding factors that impede generalization across space and time. We show that model-based augmentation is a useful tool for handling such confounding factors; a novel finding is that augmentation is helpful not just for training, but also during inference. A lower-complexity alternative to augmentation is to estimate and undo the effects of confounding factors using detailed, protocol-specific models, but, depending on the phenomenon of interest, the residual errors (e.g., from channel estimation) may swamp out the weaker nonlinear effects that we wish to learn. A judicious combination of estimation and augmentation can confer robustness, but augmentation alone is a competitive approach when we seek protocol-agnostic strategies.

Our results highlight the promise and pitfalls of deep learning for RF signatures, rather than providing definitive answers. There are a number of open issues for further investigation, including alternative DNN architectures and fundamental detection-theoretic limits to provide benchmarks for robust fingerprinting, Another important area for future work is exploration of the robustness of DNN-based RF signatures to adversarial attacks. Adversarial attacks and defenses are a topic of intensive investigation in the context of standard image datasets [37, 98, 116], but it is of interest to explore threat models that are specifically tailored to wireless physical layer security. Finally, it is important to investigate RF and mixed signal circuit design issues associated with the concept of RF signatures, including the potential for deliberately introducing manufacturing variations to enable discrimination, and characterization of the stability of device nonlinearities to environmental variations (e.g., in temperature and moisture).

5.2 Robust Self-Supervised Learning

Extracting quality features from an unlabeled dataset is a challenging task that requires careful task definition and cautious handling of the data. In this work, we present a self-supervised learning method for speaker recognition tasks designed to exploit implicit speaker identity information in unlabeled human-machine dialogues. We propose an effective soft rejection mechanism to deal with dialogues containing multiple speakers. Experiments on de-identified smart-speaker production data show that the proposed algorithm is effective at handling unsupervised speaker information, giving performance comparable to supervised models. When used for model pretraining before supervised training, our method reduces EER by up to 41% relative, compared to no pretraining, and is superior both to other self-supervised pretraining methods and to pretraining on a large labeled (but domain-mismatched) dataset.

These findings open up new research directions for learning from corrupt datasets. While we reject the corrupt data points with our soft rejection mechanism, we sacrifice valuable hard examples. Therefore, the potential next step is to explore ways to distinguish hard examples from corrupt examples.

5.3 Neuro-Inspired Robustness

Given the shortcomings of standard DNNs in terms of both interpretability and robustness, we believe it is time to explore how to control better the features generated within a DNN. We present our HaH framework as the first step in this direction. Our preliminary results demonstrate the promise of enhancing the end-to-end training paradigm in DNNs with layer-wise HaH costs in order to control the features extracted by intermediate layers. In particular, our neuro-inspired approach to neuronal competition during training and in-

ference demonstrably results in sparser, stronger activations and robustness against noise, common corruptions, and adversarial perturbations than baseline models. Indeed, based on our experiments with the CIFAR10-C (common corruptions) dataset, the robustness provided by our approach, trained in these preliminary results without any augmentation, appears to be more general-purpose than that obtained by adversarial training. We note that recent work on bio-inspired adversarial defenses appears to yield similar observations [113].

We hope these results motivate a systematic inquiry into enhancing end-to-end training with layer-wise cost functions for various architectures (specifically the ones using skip connections), training techniques (including unsupervised and semi-supervised learning, and data augmentation), and applications. In particular, for robust machine learning, a natural next step is to explore the combination of data augmentation strategies (including adversarial training) with HaH architectures. Another critical area for future exploration is the effectiveness of the HaH framework in different domains, such as speech, audio, and natural language processing.

Bibliography

[1] M. Cekic, C. Bakiskan, and U. Madhow, "Towards robust, interpretable neural networks via hebbian/anti-hebbian learning: A software framework for training with feature-based costs," *Software Impacts*, vol. 13, p. 100347, 2022.

[2] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[3] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, Jul. 2018. [Online]. Available: https://arxiv.org/abs/1802.00420

[4] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *arXiv preprint arXiv:1903.12261*, 2019.

[5] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints." *Journal of machine learning research*, vol. 5, no. 9, 2004.

[6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[7] S. Targ, D. Almeida, and K. Lyman, "Resnet in resnet: Generalizing residual architectures," *arXiv preprint arXiv:1603.08029*, 2016.

[8] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[9] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[10] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[12] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*. PMLR, 2016, pp. 173–182.

[13] M. Cekic, S. Gopalakrishnan, and U. Madhow, "Wireless fingerprinting via deep learning: The impact of confounding factors," in *2021 55th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2021, pp. 677–684.

[14] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?" in *International Conference on Machine Learning*. PMLR, 2019, pp. 7164–7173.

[15] M. Cekic, R. Li, Z. Chen, Y. Yang, A. Stolcke, and U. Madhow, "Self-supervised speaker recognition training using human-machine dialogues," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6132–6136.

[16] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 387–402.

[17] 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 (ICLR)*, 2014.

[18] S. Gopalakrishnan, M. Cekic, and U. Madhow, "Robust wireless fingerprinting via complex-valued neural networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[19] M. Du, X. He, X. Cai, and D. Bi, "Balanced neural architecture search and its application in specific emitter identification," *IEEE Transactions on Signal Processing*, vol. 69, pp. 5051–5065, 2021.

[20] K. Huang, J. Yang, H. Liu, and P. Hu, "Channel-robust specific emitter identification based on transformer," *Highlights in Science, Engineering and Technology*, vol. 7, pp. 71–76, 2022.

[21] T. Schenk, *RF Imperfections in High-rate Wireless Systems: Impact and Digital Compensation*. Springer Science & Business Media, 2008.

[22] B. Razavi and R. Behzad, *RF Microelectronics*. Prentice Hall New York, 2012, vol. 2.

[23] I. C. S. L. M. S. Committee *et al.*, "Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications: High-speed physical layer in the 5ghz band," *IEEE std 802.11 a-1999*, 1999.

[24] M. Cekic, S. Gopalakrishnan, and U. Madhow, "GitHub Repository for 'Wireless Fingerprinting via Deep Learning: The Impact of Confounding Factors'," https://github.com/metehancekic/wireless-fingerprinting, 2020.

[25] J. S. Chung, J. Huh, S. Mun, M. Lee, H. S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han, "In defence of metric learning for speaker recognition," in *Proc. Interspeech*, 2020, pp. 2977–2981.

[26] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," in *Proc. Interspeech*, 2018, pp. 1086–1090.

[27] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5791–5795.

[28] A. Saeed, D. Grangier, and N. Zeghidour, "Contrastive learning of general-purpose audio representations," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*. IEEE, 2021, pp. 3875–3879.

[29] S. Ling and Y. Liu, "DeCoAR 2.0: Deep contextualized acoustic representations with vector quantization," *CoRR*, vol. abs/2012.06659, 2020. [Online]. Available: https://arxiv.org/abs/2012.06659

[30] Y. Chung and J. R. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2020, pp. 3497–3501. [Online]. Available: https://doi.org/10.1109/ICASSP40776.2020.9054438

[31] A. T. Liu, S. Yang, P. Chi, P. Hsu, and H. Lee, "Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders," in *Proc. IEEE ICASSP*, Barcelona, 2000.

[32] A. T. Liu, S. Li, and H. Lee, "TERA: self-supervised learning of transformer encoder representation for speech," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 29, pp. 2351–2366, 2021. [Online]. Available: https://doi.org/10.1109/TASLP.2021.3095662

[33] W. Hsu, B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *CoRR*, vol. abs/2106.07447, 2021. [Online]. Available: https://arxiv.org/abs/2106.07447

[34] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018. [Online]. Available: http://arxiv.org/abs/1807.03748

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

[36] J. Uesato, B. O'donoghue, P. Kohli, and A. Oord, "Adversarial risk and the dangers of evaluating against weak attacks," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5025–5034.

[37] 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.

[38] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," *arXiv:1901.08573*, 2019.

[39] M. Pintor, F. Roli, W. Brendel, and B. Biggio, "Fast minimum-norm adversarial attacks through adaptive norm constraints," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[40] M. Cekic, C. Bakiskan, and U. Madhow, "Neuro-inspired deep neural networks with sparse, strong activations," *arXiv preprint arXiv:2202.13074*, 2022.

[41] M. Cekic, C. Bakiskan, and U. Madhow, "Layerwise hebbian/anti-hebbian (hah) learning in deep networks: A neuro-inspired approach to robustness."

[42] K. A. Remley, C. A. Grosvenor, R. T. Johnk, D. R. Novotny, P. D. Hale, M. D. McKinley, A. Karygiannis, and E. Antonakakis, "Electromagnetic signatures of WLAN cards and network security," in *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology*, December 2005, pp. 484–488.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] M. Edman and B. Yener, "Active attacks against modulation-based radiometric identification," *Rensselaer Institute of Technology, Technical report*, pp. 09–02, 2009.

[48] B. Danev, H. Luecken, S. Capkun, and K. El Defrawy, "Attacks on physical-layer identification," in *Proceedings of the Third ACM Conference on Wireless Network Security*, 2010, pp. 89–98.

[49] 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.

[50] W. Wang, Z. Sun, S. Piao, B. Zhu, and K. Ren, "Wireless physical-layer identification: Modeling and validation," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 2091–2106, 2016.

[51] 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.

[52] 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.

[53] Y. Luo, H. Hu, Y. Wen, and D. Tao, "Transforming device fingerprinting for wireless security via online multitask metric learning," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 208–219, 2019.

[54] J. Yu, A. Hu, G. Li, and L. Peng, "A robust RF fingerprinting approach using multisampling convolutional neural network," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6786–6799, 2019.

[55] 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.

[56] 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.

[57] 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.

[58] 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.

[59] J. M. McGinthy, L. J. Wong, and A. J. Michaels, "Groundwork for neural network-based specific emitter identification authentication for IoT," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6429–6440, 2019.

[60] N. Guberman, "On complex valued convolutional neural networks," *arXiv preprint arXiv:1602.09046*, 2016.

[61] 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.

[62] 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.

[63] P. Virtue, X. Y. Stella, and M. Lustig, "Better than real: Complex-valued neural nets for MRI fingerprinting," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3953–3957.

[64] Z. Zhang, H. Wang, F. Xu, and Y.-Q. Jin, "Complex-valued convolutional neural network and its application in polarimetric SAR image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 12, pp. 7177–7188, 2017.

[65] Y.-S. Lee, C.-Y. Wang, S.-F. Wang, J.-C. Wang, and C.-H. Wu, "Fully complex deep neural network for phase-incorporating monaural source separation," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 281–285.

[66] S. Scardapane, S. Van Vaerenbergh, A. Hussain, and A. Uncini, "Complex-valued neural networks with nonparametric activation functions," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2018.

[67] K. Youssef, L. Bouchard, K. Haigh, J. Silovsky, B. Thapa, and C. Vander Valk, "Machine learning approach to RF transmitter identification," *IEEE Journal of Radio Frequency Identification*, vol. 2, no. 4, pp. 197–205, 2018.

[68] I. Agadakos, N. Agadakos, J. Polakis, and M. R. Amer, "Deep complex networks for protocol-agnostic radio frequency device fingerprinting in the wild," *arXiv preprint arXiv:1909.08703*, 2019.

[69] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *European conference on computational learning theory*. Springer, 1995, pp. 23–37.

[70] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[71] M. Moghimi, S. J. Belongie, M. J. Saberian, J. Yang, N. Vasconcelos, and L.-J. Li, "Boosted convolutional neural networks." in *British Machine Vision Conference*, vol. 5, 2016, p. 6.

[72] J. Feng, Y. Yu, and Z.-H. Zhou, "Multi-layered gradient boosting decision trees," in *Advances in neural information processing systems*, 2018, pp. 3551–3561.

[73] C. Chen, Z. Xiong, X. Tian, and F. Wu, "Deep boosting for image denoising," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–18.

[74] F. Restuccia, S. D'Oro, A. Al-Shawabka, M. Belgiovine, L. Angioloni, S. Ioannidis, K. Chowdhury, and T. Melodia, "DeepRadioID: Real-time channel-resilient optimization of deep learning-based radio fingerprinting algorithms," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 2019, pp. 51–60.

[75] A. Al-Shawabka, F. Restuccia, S. D'Oro, T. Jian, B. C. Rendon, N. Soltani, J. Dy, K. Chowdhury, S. Ioannidis, and T. Melodia, "Exposing the fingerprint: Dissecting the impact of the wireless channel on radio fingerprinting," in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, July 2020, p. 10.

[76] T. Jian, B. C. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis, "Deep learning for RF fingerprinting: A massive experimental study," *IEEE Internet of Things Magazine*, vol. 3, no. 1, pp. 50–57, 2020.

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

[78] B. Razavi, *Fundamentals of Microelectronics*. Wiley, 2008.

[79] H. Zhou, C. Nicholls, T. Kunz, and H. Schwartz, "Frequency accuracy & stability dependencies of crystal oscillators," *Carleton University, Systems and Computer Engineering, Technical Report SCE-08-12*, 2008.

[80] E. Sourour, H. El-Ghoroury, and D. McNeill, "Frequency offset estimation and correction in the IEEE 802.11a WLAN," in *IEEE 60th Vehicular Technology Conference*, vol. 7. IEEE, 2004, pp. 4923–4927.

[81] T. S. Rappaport *et al.*, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall PTR New Jersey, 1996.

[82] 3GPP TS 36.101, *LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) Radio Transmission and Reception.* Version 11.2.0, release 11, 2012.

[83] 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.

[84] 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.

[85] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[86] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[87] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, "S4l: Self-supervised semi-supervised learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1476–1485.

[88] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[89] S. Sukhbaatar and R. Fergus, "Learning from noisy labels with deep neural networks," arXiv:1406.2080, 2014.

[90] R. Wang, T. Liu, and D. Tao, "Multiclass learning with partially corrupted labels," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2568–2580, 2017.

[91] S. Ling and Y. Liu, "Decoar 2.0: Deep contextualized acoustic representations with vector quantization," *arXiv preprint arXiv:2012.06659*, 2020.

[92] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Proc. Interspeech*, G. Kubin and Z. Kacic, Eds., 2019, pp. 3465–3469.

[93] L. Wan, Q. Wang, A. Papir, and I. Moreno, "Generalized End-to-end Loss For Speaker Verification," in *Proc. IEEE ICASSP*, 2018, pp. 4879–4883.

[94] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. International Conference on Machine Learning*, 2020, pp. 1597–1607.

[95] A. Saeed, D. Grangier, and N. Zeghidour, "Contrastive learning of general-purpose audio representations," in *Proc. IEEE ICASSP*, 2021, pp. 3875–3879.

[96] Y.-A. Chung and J. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *Proc. IEEE ICASSP*, 2020, pp. 3497–3501.

[97] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. Nelson, A. Bridgland *et al.*, "Improved protein structure prediction using potentials from deep learning," *Nature*, vol. 577, no. 7792, pp. 706–710, 2020.

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

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

[100] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, "Robustbench: a standardized adversarial robustness benchmark," *arXiv preprint arXiv:2010.09670*, 2020.

[101] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision research*, vol. 37, no. 23, 1997.

[102] K. Fukushima, S. Miyake, and T. Ito, "Neocognitron: A neural network model for a mechanism of visual pattern recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 826–834, 1983.

[103] G. Amato, F. Carrara, F. Falchi, C. Gennaro, and G. Lagani, "Hebbian learning meets deep convolutional neural networks," in *Image Analysis and Processing – ICIAP 2019*, E. Ricci, S. Rota Bulò, C. Snoek, O. Lanz, S. Messelodi, and N. Sebe, Eds. Cham: Springer International Publishing, 2019, pp. 324–334.

[104] M. Carandini and D. J. Heeger, "Normalization as a canonical neural computation," *Nature Reviews Neuroscience*, vol. 13, no. 1, pp. 51–62, 2012.

[105] M. F. Burg, S. A. Cadena, G. H. Denfield, E. Y. Walker, A. S. Tolias, M. Bethge, and A. S. Ecker, "Learning divisive normalization in primary visual cortex," *PLOS Computational Biology*, vol. 17, no. 6, p. e1009028, 2021.

[106] M. Ren, R. Liao, R. Urtasun, F. H. Sinz, and R. S. Zemel, "Normalizing the normalizers: Comparing and extending network normalization schemes," *arXiv preprint arXiv:1611.04520*, 2016.

[107] J. Dapello, T. Marques, M. Schrimpf, F. Geiger, D. Cox, and J. DiCarlo, "Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations," *bioRxiv*, 2020. [Online]. Available: https://www.biorxiv.org/content/early/2020/06/17/2020.06.16.154542

[108] Z. Li, W. Brendel, E. Walker, E. Cobos, T. Muhammad, J. Reimer, M. Bethge, F. Sinz, Z. Pitkow, and A. Tolias, "Learning from brains how to regularize machines," *Advances in neural information processing systems*, vol. 32, 2019.

[109] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[110] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[111] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[112] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and surface variations," *arXiv preprint arXiv:1807.01697*, 2018.

[113] H. Machiraju, O.-H. Choung, M. H. Herzog, and P. Frossard, "Empirical advocacy of bio-inspired models for robust image recognition," *arXiv preprint arXiv:2205.09037*, 2022.

[114] K. Kireev, M. Andriushchenko, and N. Flammarion, "On the effectiveness of adversarial training against common corruptions," *arXiv preprint arXiv:2103.02325*, 2021.

[115] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[116] 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.