Countering Acoustic Adversarial Attacks in Microphone-equipped Smart Home Devices

SOURAV BHATTACHARYA, Samsung AI Center, Cambridge
DIONYSIS MANOUSAKAS, University of Cambridge
ALBERTO GIL C. P. RAMOS, Samsung AI Center, Cambridge
STYLIANOS I. VENIERIS, Samsung AI Center, Cambridge
NICHOLAS D. LANE, Samsung AI Center, Cambridge and University of Oxford
CECILIA MASCOLO, University of Cambridge

Deep neural networks (DNNs) continue to demonstrate superior generalization performance in an increasing range of applications, including speech recognition and image understanding. Recent innovations in compression algorithms, design of efficient architectures and hardware accelerators have prompted a rapid growth in deploying DNNs on mobile and IoT devices to redefine user experiences. Relying on the superior inference quality of DNNs, various voice-enabled devices have started to pervade our everyday lives and are increasingly used for, e.g., opening and closing doors, starting or stopping washing machines, ordering products online, and authenticating monetary transactions. As the popularity of these voice-enabled services increases, so does their risk of being attacked. Recently, DNNs have been shown to be extremely brittle under adversarial attacks and people with malicious intentions can potentially exploit this vulnerability to compromise DNN-based voice-enabled systems. Although some existing work already highlights the vulnerability of audio models, very little is known of the behaviour of compressed on-device audio models under adversarial attacks. This paper bridges this gap by investigating thoroughly the vulnerabilities of compressed audio DNNs and makes a stride towards making compressed models robust. In particular, we propose a stochastic compression technique that generates compressed models with greater robustness to adversarial attacks. We present an extensive set of evaluations on adversarial vulnerability and robustness of DNNs in two diverse audio recognition tasks, while considering two popular attack algorithms: FGSM and PGD. We found that error rates of conventionally trained audio DNNs under attack can be as high as 100%. Under both white- and black-box attacks, our proposed approach is found to decrease the error rate of DNNs under attack by a large margin.

CCS Concepts: • **Computing methodologies → Machine learning**; **Speech recognition**; • **Security and privacy →** Security services.

Additional Key Words and Phrases: Compressed neural networks, audio adversarial attack, robust training.

ACM Reference Format:
Sourav Bhattacharya, Dionysis Manousakas, Alberto Gil C. P. Ramos, Stylianos I. Venieris, Nicholas D. Lane, and Cecilia Mascolo. 2020. Countering Acoustic Adversarial Attacks in Microphone-equipped Smart Home Devices. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 2, Article 73 (June 2020), 24 pages. https://doi.org/10.1145/3397332

Authors' addresses: Sourav Bhattacharya, Samsung AI Center, Cambridge, sourav.b1@samsung.com; Dionysis Manousakas, University of Cambridge, dionysis.manousakas@cl.cam.ac.uk; Alberto Gil C. P. Ramos, Samsung AI Center, Cambridge, a.gilramos@samsung.com; Stylianos I. Venieris, Samsung AI Center, Cambridge, s.venieris@samsung.com; Nicholas D. Lane, Samsung AI Center, Cambridge and University of Oxford, nic.lane@samsung.com; Cecilia Mascolo, University of Cambridge, cecilia.mascolo@cl.cam.ac.uk.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2020 Association for Computing Machinery.
2474-9567/2020/6-ART73 $15.00
https://doi.org/10.1145/3397332

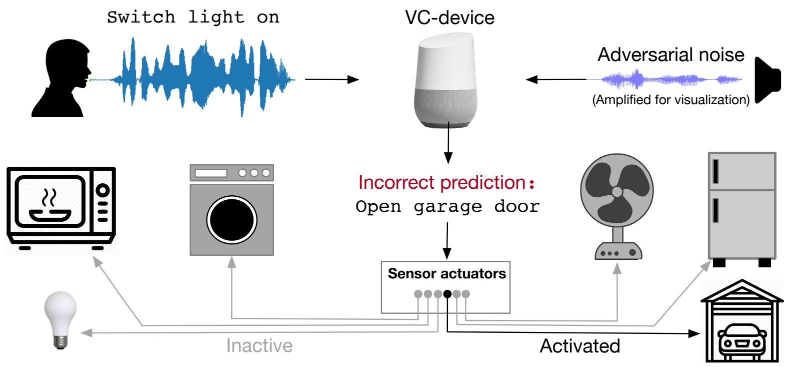


Fig. 1. Example of an everyday scenario where audio adversarial attacks can take place. A user presents a command "*switch light on*" to a voice controllable (VC) device. Without the knowledge of the user, an adversary is injecting adversarial noise, which corrupts the original input signal and the VC now incorrectly interprets the command as "*open garage door*" and actuates the corresponding sensor, which can be harmful.

1 INTRODUCTION

Audio analytics and speech interfaces, powered by deep neural networks (DNNs) [1, 2], continue to proliferate across a wide range of devices present in our everyday environments, e.g., offices, homes, and cars. Devices containing these deep audio models include not just phones and watches – but also alarm clocks, thermostats, vacuum cleaners and coffee machines. *Everything we carry and touch is becoming audio- or speech-aware*. Furthermore, these devices support a multitude of mobile and IoT applications that perform tasks such as opening and closing doors [3], authenticating monetary transactions [4], taking a picture [5], increasing or decreasing temperature of a room [6], and ordering products online [7]. This accomplishment is primarily due to breakthroughs in reduction of the memory, computation, energy and latency required by the state-of-the-art DNNs through techniques commonly known as "model compression" [8–14].

Although DNNs continue to show good generalization performance, recent studies identified that DNNs can be extremely unreliable under adversarial attacks [15–19]. Thus, as the popularity of DNN-based audio-aware devices and applications increases, so does the risk of malicious attacks targeted to these devices by exploiting the known vulnerabilities of DNNs. *Adversarial attacks* aim to inject carefully crafted tiny vectors of perturbations, imperceptible to humans, which can systematically trick an otherwise strong and accurate deep audio model to believe it is overhearing an audio class, often as desired by the attacker. In this work, we have discovered that deep neural networks for audio – *when compressed* – remain open to adversarial threats that until very recently were mainly studied for deep models that target image inputs. An illustrative attack scenario could be as follows (see Figure 1): Alice is interacting with a voice-controllable device (VC) that can trigger a number of actuators in the house, e.g., lights, microwave oven, washing machine, fan, refrigerator and garage door. Without Alice's knowledge, Bob places a malicious device, close to the VC, which is listening to the environment and injecting intermittent adversarial perturbations. Now, when Alice gives the command "*switch light on*" to the VC, it receives a corrupted version of the audio due to Bob's device. Bob carefully generated the perturbation vector such that the VC device interprets the command as "*open the garage door*" and actuates the corresponding sensor. Adversarial attacks in general can take a wide variety of forms – and the extent of vulnerabilities that can be built on such technology is still being fully investigated.

Contrary to adversarial research on images, which is well studied within the broader machine learning community, audio adversarial research is gaining popularity very recently and the process of performing a real-world audio adversarial attack brings new challenges. For instance, (i) an attacker needs to maintain *time synchronization* to conduct an *over-the-air* attack, as the audio and perturbations are now generated from two different sources, (ii) the adversarial perturbation vector can suffer from non-linear distortions during over-the-air transmission and may not remain adversarial, (iii) audio input is often subjected to various *pre-processing* for input feature representation, e.g., MFCC or *log-mel spectrogram*, which can diminish the effect of adversarial distortions, and (iv) attacks relying on gradient estimations need to navigate complex *loss-function*, e.g., CTC-Loss for *sequence-to-sequence* modeling, where a non-linear transformation is applied on top of the *softmax* activations [18]. This work makes early steps in understanding and evaluating adversarial threats of *uncompressed* and *compressed* audio models. To the best of our knowledge, we are the first to study adversarial vulnerabilities of compressed audio deep models, measure the transferability of audio attacks across compressed and uncompressed model architectures, input-representations, and across a range of current and new training-time countermeasures. The main scientific contributions of the paper can be broadly divided across three central themes:

- **Susceptibility of Compressed Models.** (i) We study the adversarial vulnerability of compressed audio models in real-world settings (see §5.1). This is an important problem, as most adversarial machine learning research today is taking place around uncompressed and full-precision models, which may have different robustness properties than their compressed counterparts. (ii) We evaluate the transferability of adversarial attacks between uncompressed and compressed models. The existence of such transferability directly relates to the success of an attack, when an attacker can only query the model and has otherwise no information about its architecture or parameters (see §5.2). (iii) We focus on two different common compression techniques: *low-rank approximation* and *quantization*, and two *adversarial threat-models*: white- and black-box attacks (see §4.4), spanning both *untargeted* and *targeted* scenarios (see §2.1).
- **Training-time Countermeasures.** We build on adversarial training methodologies and low-rank factorizations to increase the robustness and security of light-weight deep neural networks for audio signals. In particular, our work presents new training techniques for generating compressed models with increased resilience to adversarial attacks. Contrary to existing approaches of model compression, we propose an iterative *stochastic low-rank* factorization of the convolution and fully connected parameters to achieve both compression and increased robustness, and propose a new consistency loss to account for the randomness introduced in our low-rank factorizations, which we combine with traditional adversarial training (see §3). We present extensive evaluation results on their effectiveness, providing further insight into the performance differences between models tailored for small devices rather than for data centers (see §5.3). Our experiments show that the adversarial training can mitigate current vulnerabilities of deep audio models significantly.
- **Core Adversarial Feasibility.** We investigate to which extent adversarial attacks can be realized in real-world settings, where many digital-to-analog actuators and analog-to-digital sensors can easily corrupt carefully designed adversarial perturbations, rendering them ineffective. We study the effect of (i) microphone quality, and (ii) different relative positions between (a) the source of adversarial attack, (b) the user giving audio commands and (c) the device running the audio recognition system. We further show that robustly trained models have improved error rate in audio recognition tasks under attack (see §5.4).

2 ADVERSARIAL AUDIO AND MODEL COMPRESSION: A PRIMER

In this section, we provide a brief primer on adversarial audio attacks and techniques for generating adversarial examples. We also provide an overview of two popular model compression techniques used in this work for reducing the memory and computational cost of DNNs. For illustrations, we summarize a deployment pipeline of an audio recognition system in Figure 2, which will be referred to throughout this paper. In the pipeline presented,

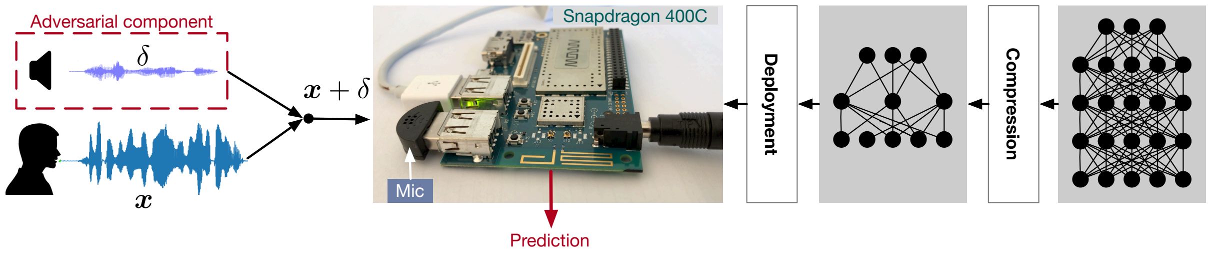


Fig. 2. Deployment pipeline of an audio recognition system on a resource-constrained embedded platform. The platform executes a compressed version of a model and accepts audio signals as input.

the embedded run-time executes a compressed version of an audio model, which takes as input audio samples that are potentially corrupted by adversarial perturbations.

2.1 Attacking Audio Recognition Systems

Audio adversarial attacks refer to generating a carefully crafted and often almost inaudible noise vector δ that, when superimposed on an input audio $\boldsymbol{x}$, causes the result $\boldsymbol{x} + \delta$ to trigger unexpected behavior of the underlying DNN, which forms the core of various on-device audio recognition systems. A typical example of unexpected behaviour is inaccurate predictions with very high confidence.

Triggering inaccurate predictions by injecting nefariously crafted noise δ is referred to as *evasion attacks* in the adversarial machine learning literature [20, 21]. Interestingly, the corrupted signal $\boldsymbol{x} + \delta$ sounds innocuous to a human listener and thus raises concerns on the reliability of state-of-the-art DNNs, especially in security applications. Audio evasion attacks can take two forms: *untargeted* and *targeted* attacks. The main objective in untargeted attack is to increase the error rate in prediction and can be used to perform *denial-of-service* (DoS) attacks. In the example scenario presented in Figure 1, an untargeted attack would never allow triggering the light by the VC, thus severely limiting the usability of such systems. Targeted adversarial audio attacks on the other hand are more malicious [18], as the attacker can trigger the VC to perform any VC-supported activity, irrespective of the input command. In Figure 1, the attacker performs a targeted attack to open a door, which was not intended by the user and can be potentially harmful.

Adversarial vulnerabilities of deep audio models can be computed by measuring the extent to which the error rate of the model under benign conditions increases with adversarial attacks. Adversarial audio examples can be generated by adding adversarial noise to the input in two possible ways: allowing natural superposition *over-the-air* or through *software-simulation*. Over-the-air attacks are more realistic but challenging, whereas software simulations to add perturbations immediately provide an upper-bound on the model vulnerability, as no distortion of δ takes place before the superposition. The software-simulation is the most prevalent approach in adversarial machine learning [2] and counter factual explanations [22].

2.2 Generating Adversarial Examples

An adversarial noise $\delta \in \mathbb{R}^d$, with a norm bound, i.e., $||\delta||_\infty \leq \epsilon$ for some small $\epsilon > 0$, is defined as any vector, which when added to an input audio $\boldsymbol{x} \in \mathbb{R}^d$, causes a DNN $h_\theta(\cdot)$ to produce an incorrect prediction. The bound on the norm, i.e., ϵ, makes sure that $\boldsymbol{x}+\delta$ remains very close to $\boldsymbol{x}$ as a means to decrease the perceptibility of the change. For untargeted attacks, the task is to find a δ, such that $\arg\max_i h_\theta(\boldsymbol{x})_i \neq \arg\max_i h_\theta(\boldsymbol{x}+\delta)_i$. However, for targeted attacks an attacker selects δ such that $y_{target} = \arg\max_i h_\theta(\boldsymbol{x} + \delta)_i$ and $\arg\max_i h_\theta(\boldsymbol{x} + \delta)_i \neq \arg\max_i h_\theta(\boldsymbol{x})_i$, where y_{target} is the prediction desired by the attacker.

The process of finding δ can be viewed as the opposite optimization task as used during DNN training. For example, given a loss function $\ell(\cdot)$, e.g., *cross entropy loss*, the gradient term $\nabla_\theta \ell(h_\theta(\boldsymbol{x}), y)$, determines how to adjust the model parameters to minimize overall loss. Interestingly, the gradient computation can be extended to the input $\boldsymbol{x}$ easily to compute $\nabla_{\boldsymbol{x}} \ell(h_\theta(\boldsymbol{x}), y)$, which captures how small changes to the input affect the loss function. This gradient information is commonly exploited by attack algorithms to generate adversarial examples. For instance, small changes in the input are carried out to maximize the loss function used during the training process and the perturbed input becomes an adversarial example for the network when $\arg\max_i h_\theta(\boldsymbol{x} + \delta)_i \neq \arg\max_i h_\theta(\boldsymbol{x})_i$. Thus for untargeted attacks, adversarial perturbations can be found by solving the following optimization problem:

$$\underset{\delta:\ ||\delta||_\infty \leq \epsilon}{\text{maximize}}\ \ell(h_\theta(\boldsymbol{x} + \delta), y), \tag{1}$$

and for targeted attacks the following optimization problem:

$$\underset{\delta:\ ||\delta||_\infty \leq \epsilon}{\text{maximize}}\ \ell(h_\theta(\boldsymbol{x} + \delta), y) - \ell(h_\theta(\boldsymbol{x} + \delta), y_{target}). \tag{2}$$

Next, we describe two state-of-the-art adversarial attack algorithms that we consider in this work.

2.2.1 Fast Gradient Sign Method (FGSM). Goodfellow *et al.* [15] introduced the FGSM to compute adversarial examples efficiently. FGSM solves the above optimization problems by performing one gradient step and estimating the perturbation δ as:

$$\delta \longleftarrow \epsilon \cdot sign(\nabla_\delta \ell(h_\theta(\boldsymbol{x} + \delta), y)).$$

2.2.2 Projected Gradient Decent (PGD). One potential way to improve the quality of the attack is to consider several gradient computation steps, contrary to the single step used by the FGSM. To decrease the sensitivity of the update rule to the absolute scale of the computed gradient, a common technique is to apply gradient normalization. For ℓ_∞ bounded perturbations, the normalized gradient update rule can be summarized as follows: initialize the perturbation vector $\delta \longleftarrow \mathbf{0}$ and then apply N gradient steps as:

```
Repeat ($N$):
    $\delta \longleftarrow \mathcal{P}(\delta + \alpha \cdot sign(\nabla_\delta \ell(h_\theta(\boldsymbol{x} + \delta), y)))$,
```

where $\mathcal{P}(\cdot)$ is the projection operator on the ℓ_∞-ball defined by $||\delta||_\infty \leq \epsilon$. PGD is considered a very strong attack algorithm that utilizes first-order information about the network [23].

2.3 Deep Model Compression

State-of-the-art DNNs can generate highly accurate predictions, however, they often have over millions of parameters and suffer from high computational demands, even just for running a single forward pass [14]. High resource demands make parameter- and computation-heavy DNNs unsuitable for on-device deployments. To overcome the resource challenges, a number of model compression approaches have been proposed lately that allow generating a highly memory- and computation-efficient model, without losing much on the performance accuracy. Examples of popular model-compression techniques include *convolution separation* [24], *depth-wise convolution* [25], *model distillation* [26], *low-rank factorization and quantization* [27]. The ability of shaping resource demands, without losing on performance, allowed a rapid adoption of compressed DNNs on embedded and IoT platforms to redefine users' experiences, especially in noisy sensor-inference tasks. As a result, we are witnessing an increasing number of devices in our everyday life becoming voice- and vision-aware. Other benefits of locally running DNNs include *privacy preservation*, quick response time, and non-reliance on the availability of Internet connectivity among others [28, 29].

The model compression techniques mentioned above have the only goal of maintaining accuracy while reducing resource demands, and very little is known on their adversarial vulnerabilities. In this work, we bridge this gap by presenting the first systematic performance evaluation of compressed deep audio models under adversarial attacks. For this purpose, in this work we consider *low-rank factorization* and *quantization* as the two popular model compression schemes to reduce the memory and computational footprints of DNNs. In the following we describe both model compression techniques briefly.

2.3.1 Low-rank Approximation. It allows to simultaneously decrease the number of parameters and overall number of floating-point operations needed for inference by a DNN [14]. For example, in a fully connected layer, low-rank approximations are often realized by replacing the matrix $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ by surrogates of the form $\boldsymbol{U}\text{diag}(\boldsymbol{s})\boldsymbol{V}$, with $\boldsymbol{U} \in \mathbb{R}^{m \times k}$, $\boldsymbol{s} \in \mathbb{R}^{k}$ and $\boldsymbol{V} \in \mathbb{R}^{k \times n}$ leading to an approximation of the form:

$$\boldsymbol{W} \cdot \boldsymbol{z_i} \approx \boldsymbol{U} \cdot (\text{diag}(\boldsymbol{s}) \cdot (\boldsymbol{V} \cdot \boldsymbol{z_i})),$$

which results in a decrease of the number of parameters from $O(mn)$ to $k \cdot O(m+n)$, if k is chosen small enough [14, 30–32]. A typical approach is to: (i) train the original network until a good classification performance is achieved, (ii) perform *singular value decomposition* (SVD) on the trained weights $\boldsymbol{W}$ as a means to find quantities $\boldsymbol{U}$, $\boldsymbol{s}$ and $\boldsymbol{V}$ as above, (iii) initialising a surrogate network with $\boldsymbol{U} \cdot \text{diag}(\boldsymbol{s}) \cdot \boldsymbol{V}$ instead of $\boldsymbol{W}$ (see [30]), and (iv) re-training the surrogate network to recover any classification degradation that may have been introduced by the compression. Finally, for a convolutional layer with filters $\tilde{\boldsymbol{W}} \in \mathbb{R}^{a \times b \times c \times d}$ a similar procedure can be carried out by first reshaping the filters into a matrix $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ with $m = a \times b$ and $n = c \times d$, then producing a low-rank factorization and finally reshaping back to an appropriate four dimensional tensor [14, 33].

2.3.2 Quantization. It aims to reduce memory and computations by representing parameters and/or activations of a DNN with fewer number of bits than the conventional 32-bit floating-point format (FP32) [8]. When reducing the bit-depth of representation, one common strategy is to first train a DNN in full-precision and then apply fixed-point quantization to the trained network parameters and activations. This strategy is commonly known as *post-training quantization* and has been shown to work well [34]. To maintain accuracy, a dynamic fixed-point scheme with uniform bit-depth and different scaling factors across layers is often employed. Model re-training can be further performed to fine-tune the fixed-point parameters of the quantized model and gain back accuracy drops due to reduced precision [11, 35]. The training algorithm can also be modified to perform quantization-aware training, where the bit-depth of model parameters throughout the training procedure is kept low [9, 12, 36]. Both quantization-aware training and re-training of post-training quantization approaches need access to the training dataset, which is often difficult, especially in transfer learning approaches aimed to reduce the amount of required data.

In this work a post-training strategy is employed using dynamic fixed-point quantization without re-training. For a given bit-depth, the per-layer scaling factors are tuned based on the dynamic range of its parameters and activations, estimated using a small set of input sampled from the target domain.

3 MAKING COMPRESSED NEURAL NETWORKS ROBUST

A high degree of resilience to adversarial examples is a desirable property of DNNs, especially when deployed to support security applications on constrained devices in the wild. In order to make light-weight DNNs robust to adversarial examples, it is essential to: i) minimize their *empirical adversarial risk*, and ii) decrease their *size* and *compute requirements*. In this paper, we decrease adversarial risk by extending adversarial training techniques [23, 37] with an iterative noisy factorization of convolution and fully connected layer parameters. Our approach adds stochasticity to the model by injecting noise to the hidden variables in low-rank factorization [14, 30] during model compression and applies consistency training. The result is a robust light-weight ensemble method, which averages softmax probabilities of randomized models during an inference. We improve the generalizability of the model via consistency training, i.e., by augmenting the cross entropy and adversarial training objectives with the average of the Kullback-Leibler divergences between the different softmax outputs whose average yields the ensemble. By taking advantage of the structure of low-rank factorizations, and modifying only a linear number of diagonal elements, we reduce ensemble costs and observe an improvement on the effect of adversarial training.

There are *three* main lines of research related to our approach. (i) Applying *adversarial training* to deterministic uncompressed models. Closest to our approach is the work by Kannan *et al.* [38], which leverages adversarial logit pairing to improve the robustness of DNNs operating in the image domain. In particular, the authors propose a ℓ_2 penalization term in the loss function to encourage the logits generated by an original input $\boldsymbol{x}$ and corresponding attacked input $\boldsymbol{x} + \delta$ to become closer during training. (ii) Applying *consistency training* between the outputs of a deterministic model computed from an input and a perturbed (non-adversarial) version of that input. The work in [39] (and references therein) encourage softmax outputs of both input and its perturbation to be perceived similar. (iii) Introducing randomness to decrease adversarial risk by adding noise to the inputs of the layers. Notable work includes [40] in the image domain, which injects noise to all layers of a DNN.

The remainder of this section is organized as follows: we define our adversarially robust low-rank ensemble method in §3.1, and describe the three main terms in the total loss function, namely the cross entropy loss in §3.2, the consistent training loss in §3.3, and the adversarial training loss in §3.4. These losses are then aggregated into a weighted total loss in §3.5. Finally in §3.6, we provide a description of our proposed algorithm for training robust light-weight low-rank stochastic ensemble model.

3.1 Stochastic Compression

We present an efficient stochastic ensemble method that leverages the structure of commonly used low-rank factorization techniques to decrease the adversarial risk of a wide-range of light-weight models designed to run on microphone-equipped constrained devices. At a high level, greater resilience to adversarial attacks comes from the randomness introduced in the singular values of the low-rank approximations, which makes gradient estimation difficult. For this purpose we add Gaussian noise $\eta \sim \mathcal{N}(\mu, \sigma)$ to the diagonal matrix $\boldsymbol{s}$, computed using SVD (see §2.3.1), after every gradient update step during training for all fully connected and convolution layers. The parameters μ, σ, controlling the extent of noise, are kept fixed for a layer during the entire training procedure. The stochasticity thus introduced generates different softmax output probabilities for different forward passes through the network for the same input. To overcome this variation, during an inference, we execute the model multiple times and ensemble individual outputs to compute the final prediction by the stochastic model. Memory footprint reduction of the model comes from the low-rank approximations themselves, which can be combined with a range of reduced-precision techniques such as quantization (see §2.3.2). Improvement in latency comes from the fact that to compute the ensemble only a small number of parameters, namely the singular values, need to be updated for each different instantiation of the model for a given input.

3.2 Cross Entropy Loss

As customary for supervised training based on labeled data (x, y), we consider the cross entropy or negative log likelihood loss. Specifically, we denote by $h_\theta(\boldsymbol{x})$ the deterministic output of an uncompressed model, and by

$$\overline{c}_\theta(\boldsymbol{x}) := \frac{1}{n} \sum_{\substack{i=1 \\ \theta^{(i)} \sim \theta}}^{n} c_{\theta^{(i)}}(\boldsymbol{x}),$$

the stochastic ensemble average output of compressed models – each instantiated from the same architecture but with different perturbations of the singular values. Next, we denote the cross entropy of deterministic models by

$$\ell(h_\theta(\boldsymbol{x}), y) := -\log([h_\theta(\boldsymbol{x})]_y) \tag{3}$$

and define the cross entropy loss of a stochastic model over n ensemble by $\ell(\bar{c}_\theta(\boldsymbol{x}), y)$.

3.3 Consistency Training Loss

Averaging the probabilities across the ensemble of compressed models – with the same architecture but with different perturbations of the singular values – helps training more robust predictors. However, it is important to encourage each ensemble element to be different but similar, by augmenting the objective function to reward consistency between the probabilities given by each compressed model variant. To this end, we propose the following consistency loss:

$$\frac{1}{n(n-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^{n} \mathrm{KLD}\left(c_{\theta^{(i)}}(\boldsymbol{x}) \| c_{\theta^{(j)}}(\boldsymbol{x})\right) \tag{4}$$

where $\theta^{(1)}, \ldots, \theta^{(n)}$ denote different samples from θ, corresponding to random perturbations of the singular values of the low-rank factorizations.

3.4 Adversarial Training Loss

We employ adversarial training [20, 37, 41] to increase the robustness of our light-weight ensemble of low-rank models, which consists of a two-stage procedure: for each sample, an adversarial attack is first computed with the current parameter configuration (see §3.4.1); then the resulted attack is used to update the parameter configuration to minimize the discrepancy of the model outputs when applied to the original and perturbed inputs (see §3.4.2).

3.4.1 Min-max Optimization. Robust models can be defined as those, which achieve small error rate on a test dataset $\mathcal{D}$ corrupted with ℓ_∞-bounded adversarial perturbations. In this respect, robust training can be performed by optimizing

$$\underset{\theta}{\text{minimize}} \frac{1}{|\mathcal{D}|} \sum_{(\boldsymbol{x},y)\in\mathcal{D}} \underset{\delta\in\Delta(\boldsymbol{x})}{\text{maximize}}\ \ell(\bar{c}_\theta(\boldsymbol{x}+\delta), y),$$

where $\Delta(\boldsymbol{x})$ denotes a small ℓ_∞-ball around $\boldsymbol{x}$. For gradient-based optimization methods, the Danskin's theorem[1] [42] is often employed and the gradients are computed as follows

$$\nabla_\theta \underset{\delta\in\Delta(\boldsymbol{x})}{\text{maximize}}\ \ell(\bar{c}_\theta(\boldsymbol{x}+\delta), y) = \nabla_\theta \ell(\bar{c}_\theta(\boldsymbol{x}+\tilde{\delta}), y),$$

where

$$\tilde{\delta} = \underset{\delta\in\Delta(\boldsymbol{x})}{\text{argmax}}\ \ell(\bar{c}_\theta(\boldsymbol{x}+\delta), y). \tag{5}$$

Since it is often not possible to compute Eq. (5) exactly, approximate algorithms such as FGSM and PGD are used instead to yield approximate solutions.

3.4.2 Kullback-Leibler Loss. Adversarial training can be viewed as learning parameters for which the loss function is smooth in the neighborhood of each input data point. Given adversarial examples break this tenet, the underlying idea is then for every legitimate input $\boldsymbol{x}$ and adversarial attack $\boldsymbol{x}+\delta$ to update the parameters of the model so that $\ell(\bar{c}_\theta(\boldsymbol{x}), y)$ becomes closer to $\ell(\bar{c}_\theta(\boldsymbol{x}+\delta), y)$, or that $\bar{c}_\theta(\boldsymbol{x}) \approx \bar{c}_\theta(\boldsymbol{x}+\delta)$. This can be encouraged by adding the following Kullback–Leibler regularization to the minimization objective

$$\mathrm{KLD}\left(\frac{1}{n}\sum_{i=1}^{n} c_{\theta^{(i)}}(\boldsymbol{x}) \middle\| \frac{1}{n}\sum_{i=1}^{n} c_{\theta^{(i)}}(\boldsymbol{x}+\tilde{\delta})\right), \tag{6}$$

where $\theta^{(1)}, \ldots, \theta^{(n)}$ correspond to different perturbations of the low-rank weights.

[1] Danskin's theorem requires the inner maximization problem to be convex, which is often not the case in practice.

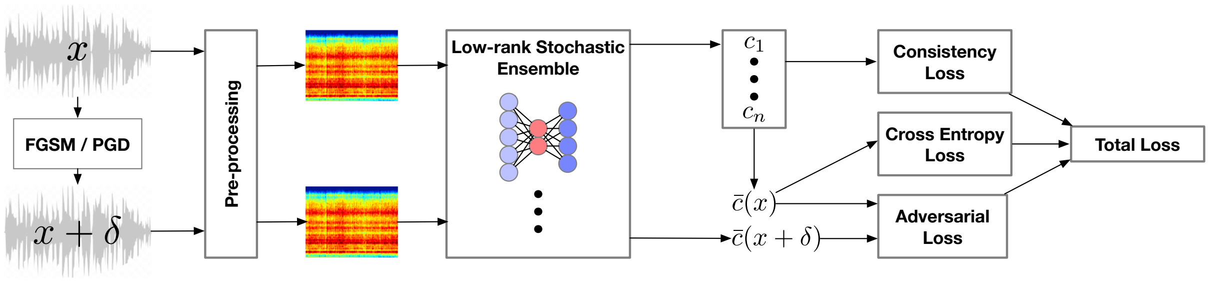


Fig. 3. Overview of the proposed robust audio training framework. The arrows indicate the direction of information flow during the forward pass. For each audio input $\boldsymbol{x}$, a copy is generated, which gets corrupted by adversarial noise constructed via FGSM or PGD, which yields an attack $\boldsymbol{x} + \delta$. Both signals are then pre-processed and put through the low-rank stochastic ensemble, which produces individual predictors $c_{\theta^{(1)}}, \ldots, c_{\theta^{(n)}}$ from the same architecture but with different perturbations of the singular values of the low-rank approximations. These individual predictors are then averaged to produce the ensemble softmax output $\bar{c}_\theta$. The proposed total loss is composed of three parts: a consistency loss term which depends on the KL divergences between individual predictors applied to the original signal $\boldsymbol{x}$, a cross entropy loss term which is determined by the ensemble softmax applied to the clean signal $\boldsymbol{x}$, and an adversarial loss which depends on the KL divergence between the ensemble softmax applied to the original $\boldsymbol{x}$ and corrupted $\boldsymbol{x} + \delta$ inputs. The gradients from the losses are combined together and back-propagated through the network during training.

3.5 Total Training Loss

As depicted in Figure 3, given a labeled sample (x, y) and a particular instantiation of $\theta^{(1)}, \ldots, \theta^{(n)} \sim \theta$, we compute an adversarial perturbation (Eq. 5) based on a FGSM or PGD approximation as:

$$\tilde{\delta} := \underset{\delta \in \Delta(\boldsymbol{x})}{\operatorname{argmax}} \, \ell\left(\frac{1}{n}\sum_{i=1}^{n} c_{\theta^{(i)}}(\boldsymbol{x} + \delta), y\right)$$

Finally, we propose the total training loss defined as the weighted sum of the cross entropy (Eq. 3), consistency (Eq. 4) and adversarial (Eq. 6) losses as:

$$\mathcal{L}_{\text{Total}} := \alpha\ell\left(\frac{1}{n}\sum_{i=1}^{n} c_{\theta^{(i)}}(\boldsymbol{x}), y\right) + \frac{\beta}{n(n-1)}\sum_{\substack{i,j=1 \\ i\neq j}}^{n} \text{KLD}\left(c_{\theta^{(i)}}(\boldsymbol{x}) \| c_{\theta^{(j)}}(\boldsymbol{x})\right) + \gamma\text{KLD}\left(\frac{1}{n}\sum_{i=1}^{n} c_{\theta^{(i)}}(\boldsymbol{x}) \middle\| \frac{1}{n}\sum_{i=1}^{n} c_{\theta^{(i)}}(\boldsymbol{x} + \tilde{\delta})\right)$$

where $\alpha \geq 0$, $\beta \geq 0$ and $\gamma \geq 0$ determine the relative weighting between the three loss components subject to the constraint $\alpha + \beta + \gamma = 1$. Overall, at training time, we minimize the total loss with respect to the unperturbed singular value decompositions and the model's trainable parameters.

3.6 Robust Audio Training Pipeline

In this section, we present an overview of our proposed robust audio training pipeline, which is also outlined in Algorithm 1. The input to the training framework is the audio dataset, the architecture of the DNN, the target ranks and ϵ which is the radius of the ℓ_∞-ball. We begin by splitting the dataset into non-overlapping training (80%), validation (10%) and test (10%) sets (line 5). During this process, we perform stratified sampling to take into account any class imbalances present in the data. Next, model parameters are initialized either randomly or from a previous checkpoint. We adopt mini-batch SGD and use the *Adam* optimizer in all our experiments with default hyperparameters. We perform a fixed number of parameter update steps for all the algorithms ($S_{max} = 50,000$, line 7). Inside the training loop, we fetch an audio training batch and perform any necessary pre-processing or

Algorithm 1 Robust Audio Model Training

```
1: Input: (i) Dataset $\mathcal{D}$ of samples $(x, y)$, (ii) DNN architecture to train ($\bar{c}_\theta$), iii) target ranks, and (iv) the $\ell_\infty$-ball radius $\epsilon$
2: Output: Trained model
3: $\bar{c}_\theta$ := modelArchitecture(ranks)
4: $\theta$ := init()                                  ▷ Initializing the model parameters
5: $\mathcal{D}_{train}, \mathcal{D}_{val}, \mathcal{D}_{test} := splitDataset(\mathcal{D}, 80, 10, 10)$
6: $S_{max}$ := 50,000                                 ▷ The maximum number of steps
7: for $i$ := 1 to $S_{max}$ do
8:     $\mathcal{B}$ := getNextBatch($\mathcal{D}_{train}$)
9:     $\boldsymbol{x}, y$ := preProcess($\mathcal{B}$)
10:    $\tilde{\delta}$ := AdversarialAttack($\boldsymbol{x}, \bar{c}_\theta, \epsilon$)          ▷ Maximization
11:    $\mathcal{L}_{Total} := \alpha\ell\left(\bar{c}_\theta(\boldsymbol{x}), y\right) + \frac{\beta}{n(n-1)}\sum_{i\neq j} \text{KLD}\left(c_{\theta^{(i)}}(\boldsymbol{x}) \| c_{\theta^{(j)}}(\boldsymbol{x})\right) + \gamma\text{KLD}(\bar{c}_\theta(\boldsymbol{x}) \| \bar{c}_\theta(\boldsymbol{x} + \tilde{\delta}))$
12:    $\theta := \theta - \frac{\eta}{|\mathcal{B}|}\nabla_\theta \mathcal{L}_{Total}$           ▷ Minimization
13:    if modulo(i, 100) == 0 then
14:        accuracy = computeAccuracy($\mathcal{D}_{val}, \bar{c}_\theta$)
15:        if bestPerformance(accuracy) then
16:            saveModel($\bar{c}_\theta$)
17: return readModel($\bar{c}_\theta$)
```

feature extraction step (line 9). In our evaluation experiments, we consider both *end-to-end* models that operate directly on raw audio samples and conventional audio recognition models that operate on *log-mel-frequency spectrograms* [43]. For all audio measurements in the batch, we respectively find adversarial examples and generate corrupted audio samples (line 10). While generating the adversarial example, the perturbation vector δ conforms to the ℓ_∞ neighborhood. Next, we compute the total loss and its gradient (line 11-12) and perform a parameter update step (line 12). During the training process, we sporadically monitor performance of the trained model on the uncorrupted validation set and save the best model found thus far (line 15-18). At the end of the training, the best model found on the validation set is returned as the final robustly trained model.

4 EXPERIMENTAL SETUP

In this section, we present the details of the datasets we use in our experiments, summarize the baseline algorithms and outline the various threat models considered in this work.

4.1 Datasets

As representative mobile and embedded audio sensing tasks, we focus on on-device *speech-command* recognition and *ambient audio* classification problems. In the following, we briefly describe the datasets we use for both tasks.

4.1.1 Speech Command (SC). The first audio dataset we consider is comprised of audio recordings of word utterances from a small vocabulary [44]. Specifically, the objective is to recognize among ten spoken words: *yes, no, up, down, left, right, on, off, stop* and *go*. To test the performance of the trained classifier on *unrecognized* classes, the dataset also contains recordings of a relatively large number of words outside of the target vocabulary, including silence. Overall, the dataset contains about 105, 000 audio clips, each 1 second in duration sampled at 16KHz. Moreover, the dataset is relatively balanced with roughly 3, 750 recordings available for each individual word from a large number of participants.

Table 1. A summary of DNN architectures used to study the performance vulnerability of models under adversarial attacks. On the Speech command (SC) dataset, we train three model architectures: end-to-end (ETE), log-mel (LM), and log-mel small (LMS). The ETE model accepts a second-long raw (or PCM) audio as the input, whereas, both LM and LMS models accept inputs after extracting log-mel spectrograms. Similarly, on the Ambient (A) dataset we train two models: end-to-end (ETE) and log-mel (LM). The ETE accepts raw (or PCM) inputs of three seconds duration, while the LM model extracts log-mel spectrograms before running an inference.

Model	Dataset	Input	Size (MB)	Architecture
ETE_{SC}	*SC*	Raw (PCM)	11.7	*c*:5[ɩ]; *p*:3[‡]; *bn*:5[†]; *fc*:2[★]
LM_{SC}	*SC*	log-mel freq.	20.2	*c*:3[ɩ]; *p*:3[‡]; *bn*:3[†]; *fc*:2[★]
LMS_{SC}	*SC*	log-mel freq.	1.8	*c*:2[ɩ]; *p*:2[‡]; *bn*:2[†]; *fc*:1[★]
ETE_{A}	*A*	Raw (PCM)	19.7	*c*:5[ɩ]; *p*:3[‡]; *bn*:5[†]; *fc*:2[★]
LM_{A}	*A*	log-mel freq.	33.3	*c*:3[ɩ]; *p*:3[‡]; *bn*:3[†]; *fc*:2[★]

[ɩ] *convolution layers*; [‡] *pooling layers*; [†] *batch normalization*; [★] *fully connected layers*

4.1.2 Ambient (A). The second audio dataset we consider is the LITIS Rouen audio scene dataset [45], referred in this paper as the ambient dataset, which contains over 1, 500 minutes of audio scenes, which were captured using Samsung Galaxy S3 smartphones. The dataset is composed of 19 different ambient scenes such as, ‘plane’, ‘busy street’, ‘bus’, ‘cafe’, ‘student hall’ and ‘restaurant’. Several 30-seconds long audio files from each ambient environment are provided. Every recording is originally sampled with frequency of 22.05KHz, which we downsampled to 16KHz for consistency with the speech command dataset described above. To permit continuous inference in practice, we split the files into three second chunks. Contrary to the speech command dataset, the ambient dataset exhibits a high degree of class imbalance.

4.2 Deep Neural Network Architectures

To present a representative set of experiments, we train a number of CNNs on the Speech command (SC) and Ambient (A) datasets that vary in size and input layer type. In particular, we consider end-to-end models operating on pulse code modulation (PCM) data, as well as models operating on log-mel (LM) data. The CNN architectures operating on PCM measurements, are adapted from the SoundNet5 [46] architecture. A summary of the model architectures for the two different tasks is given in Table 1.

4.3 Baseline Algorithms

In this section, we describe three popular approaches within the adversarial machine learning community to build robust models. In our evaluation experiments, we compare our proposed solution with these approaches.

4.3.1 Defensive Distillation. The defensive distillation approach is used by Carlini *et al.* [19], where the authors use repeated self-distillation while using a high temperature in the softmax layer, namely:

$$\text{softmax}(\boldsymbol{x}, T)_i := \frac{\exp(\boldsymbol{x}_i/T)}{\sum_j \exp(\boldsymbol{x}_j/T)}.$$

The use of high temperature enforces the output probability distribution to have higher entropy, which is beneficial for training the student network. The overall procedure can be summarized as follows:

- Train a teacher network using hard labels (e.g., one hot vectors) by setting a temperature $T > 1$.
- Compute soft labels of the training dataset using the trained teacher network (using the same T).
- Train a student network (same architecture) on the soft-labeled dataset using the temperature T.
- Repeat the self-distillation process if necessary.
- Final predictions are performed using the distilled network and using temperature $T = 1$.

4.3.2 Adversarial Data Augmentation. Augmenting the training dataset with adversarial examples is a common technique in adversarial training to (i) increase training data amount and (ii) improve the generalization performance of classifiers, e.g., a DNN. The basic idea is to generate several copies of the input and perform some data transformations, e.g., rotation and cropping in the case of images, and adding adversarial noise in the audio. Once the additional data are generated, the model is trained on the combined original and augmented dataset.

4.3.3 Random Self-ensemble. Lastly, we consider a stochastic self-ensemble technique proposed by Liu *et al.* [40]. Similarly to our work, the paper also adds randomness to the model by injecting noise, but whereas we inject noise in the singular values of low-rank decompositions, Liu *et al.* instead inject noise to the input and intermediate-layer activations. Unlike us, they do not propose any consistency loss to alleviate the effect of the ensemble and therefore need a greater number of elements in the ensemble.

4.4 Threat Models

The defence solutions presented in this work are evaluated under two threat models: *white-* and *black-box* attacks. Threat models represent capabilities of the adversary and in all our experiments we assume that the adversarial perturbations are limited by ℓ_∞ norm.

- **White-box (WB) attack:** Under white-box attacks an attacker is assumed to have a complete knowledge of the underlying DNN deployed for audio recognition tasks, i.e., knowledge of the exact architecture, the input, the type of input pre-processing to be applied and all the parameters of the model [19].
- **Black-box (BB) attack:** On the contrary, in black-box attacks an attacker is assumed to have no such knowledge about the underlying model, but can only observe the predicted class probability vector and have query access to the model. Under the black-box assumption, the process of generating adversarial examples cannot rely on the exact gradient of the loss function, with respect to the input, but rather uses approximations of it via efficient stochastic estimates. In our experiments, we use a derivative-free iterative optimization technique based on Natural Evolution Strategies (NES) [47, 48], which maximizes the loss under a search distribution in a small neighborhood around an input data point.

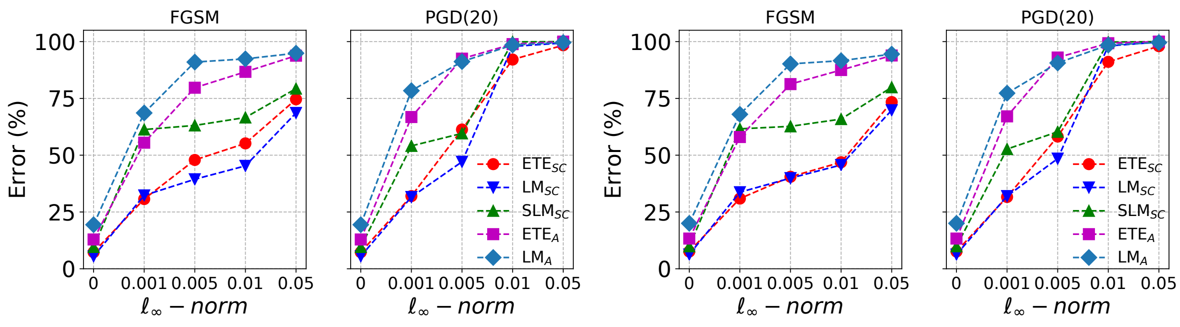
5 EVALUATION

In this section, we summarize our results from a number of experiments to understand the performance of DNNs under adversarial audio attacks. We begin by presenting results that show severe vulnerabilities of audio DNNs (§5.1) and then study the transferability of adversarial audio examples across different model architectures and compression techniques (§5.2). Next, we present results of the robust training approach proposed in this work (§5.3) and we conclude our evaluation by presenting a real-world case study (§5.4).

5.1 Vulnerability of Deep Neural Networks

We study the error rates of DNNs under various deployment scenarios. Specifically, we consider benign or uncorrupted scenarios, and scenarios where the input to a DNN is corrupted with adversarial perturbations. In adversarial experiments, we vary the upper bound on the norm of the perturbation vector ϵ and consider both *white-* and *black-box* attacks.

Figure 4 shows the error rates of uncompressed and compressed version of the models presented in Table 1 under benign and adversarial conditions. While generating compressed version of the models, we chose ranks that resulted in less than 2% relative drop in accuracy compared to the uncompressed models. We empirically found that a rank of 16 for the fully connected layers and a rank of 64 for the convolution layers meet the accuracy requirement. For generating adversarial perturbations we consider untargeted white-box attacks using both FGSM and PGD(20) algorithms. The results illustrate that when faced with either FGSM or PGD attacks



(a) Uncompressed (b) Compressed

Fig. 4. Prediction errors for the ETE, LM and SLM models summarized in Table 1 for the *Speech command* (SC) and *Ambient* (A) datasets under benign (i.e., $\delta = 0$) as well as white-box untargeted attacks using the FGSM and PGD(20) methods for various perturbation levels $\delta > 0$. Note that $\epsilon = 0.001, 0.005, 0.01, 0.05$ correspond to 6, 8, 10, 12 bit perturbations in the sampled pulse code modulation with values ranging between $\pm 2^{15}$ integer values. Compressed models were derived with rank 16 for the fully connected layers and rank 64 for the convolutional layers.

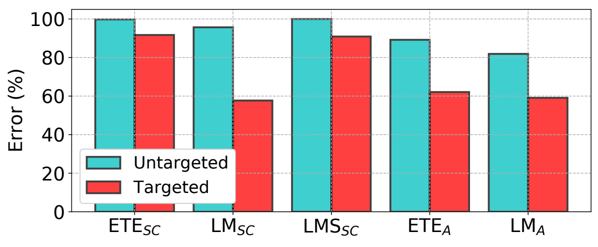


Fig. 5. Prediction errors of the models described in Table 1 on *Speech command* (SC) and *Ambient* (A) datasets for untargeted and targeted black-box NES attacks.

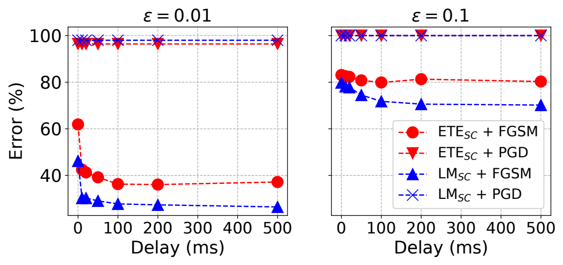
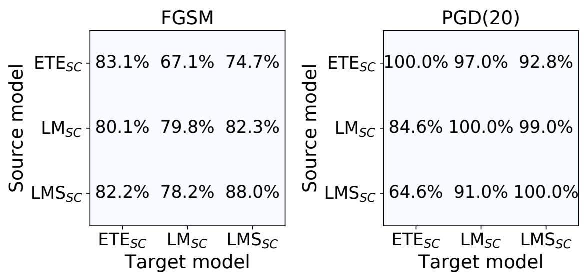


Fig. 6. Transferability of untargeted white-box FGSM and PGD(20) adversarial perturbations over synchronization delay (ms) for different values of ϵ.

for relatively high ϵ values, the error rate becomes extremely high and the recognition system employing a DNN model becomes completely unusable for both the Speech command (SC) and Ambient (A) datasets. In the experiments we consider $\epsilon = 0.001, 0.005, 0.01, 0.05$, which correspond to 6, 8, 10, 12 bit perturbations in the sampled pulse code modulation with values ranging between $\pm 2^{15}$ (integer values). To complement the results of untargeted white-box attacks, Figure 5 focuses on untargeted and targeted black-box NES attacks, for the same (uncompressed) DNNs as before. Here we carried out 20 iterations of gradient descent and for deriving gradient estimates, we use a population of 64 datapoints obtained using a search distribution of Gaussian random noise with $\sigma = 10^{-3}$ (see [48] for details). In line with existing research, our results show that DNNs, trained typically by minimizing cross-entropy loss, when deployed for audio recognition tasks remain extremely vulnerable to adversarial attacks.

5.2 Transferability of Adversarial Audio Examples

This section pursues two distinct questions related to transferability of audio adversarial attacks on real-world audio applications. The first question is *to which degree are untargeted white-box adversarial attacks affected by the lack of synchronization between an attacker's and victim's audio signals?* The second question relates to *how*



	ETE$_{SC}$	LM$_{SC}$	LMS$_{SC}$
FGSM ETE$_{SC}$	83.1%	67.1%	74.7%
LM$_{SC}$	80.1%	79.8%	82.3%
LMS$_{SC}$	82.2%	78.2%	88.0%
PGD(20) ETE$_{SC}$	100.0%	97.0%	92.8%
LM$_{SC}$	84.6%	100.0%	99.0%
LMS$_{SC}$	64.6%	91.0%	100.0%

Fig. 7. Error rates demonstrating the transferability of untargeted white-box FGSM and PGD(20) adversarial perturbations across ETE, LM and LMS models for $\epsilon = 0.1$.

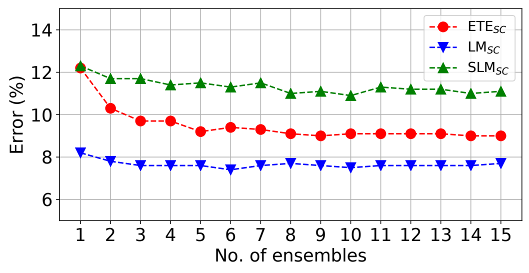


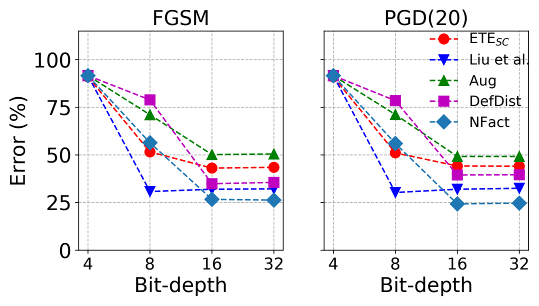
Fig. 8. Variation of the error rate as the number of ensemble elements is varied during inference for ETE, LM and LMS models.

feasible it is to generate adversarial examples for a surrogate model and then successfully attacking a system with the generated examples? Answers to these questions provide insights on the transferability of attacks across audio models.

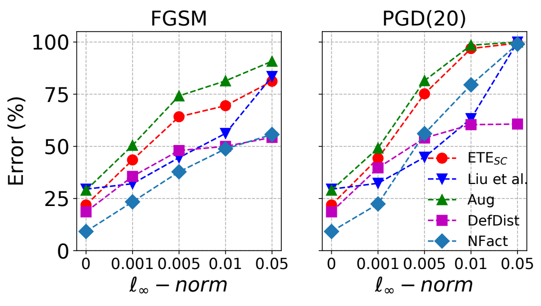
As an insight to the first question outlined above, in Figure 6 we present results from an experiment where we systematically increase the synchronization delay between victim's and attacker's signals. Specifically, we vary the delay between 0 to 500 m sec. From the figure we can observe that untargeted white-box adversarial attacks based on the PGD algorithm are more resilient to the lack of synchronisation than those based on the FGSM, across different norm bounds on the perturbation vector generated by the attacks, represented in the figure by $\epsilon = 0.01$ and $\epsilon = 0.1$. This is an important observation given the popularity of the FGSM and the fact that the lack of synchronisation between two signals is an important characteristic of real-world situations, where the two signals are generated by two independent entities acting without any kind of coordination.

To answer the second question, we conduct a number of experiments, where we select a pair of models (from Table 1) as the source and target model pair, and generate adversarial examples for the source model (assuming white-box access) and use the generated adversarial signals to attack the target model (assuming black-box access). Note that in the experiments the source and target models vary greatly in terms of their architecture, capacity and input representations. The results of the experiments are shown in Figure 7. From the figure we see that untargeted white-box adversarial attacks generated by the FGSM and the PGD algorithm do transfer well across different models, with almost perfect transfer for PGD. Overall, the results highlight the severe vulnerabilities of VC devices employing DNNs in general.

In this section, we also present experimental findings to understand the accuracy and robustness of various compression techniques commonly applied on the embedded and IoT platforms. For example, Figure 4(b) shows the effect that the magnitude of the adversarial perturbations has on the error rate when the models is factorized (see §2.3.1) for memory and computation gains. In the same vein, to assess the impact of precision quantization across both the accuracy and robustness of the developed audio models, we investigated the performance of each model across four evaluation setups under different bit-depths in Figure 9(a) for $\epsilon = 0.001$. For our experiments, we employed three reduced-precision representations using the quantization method discussed in §2.3.2: 16-bit (INT16) fixed-point which has been extensively studied to yield similar accuracy to the FP32 baseline [11, 35], 8-bit (INT8) and 4-bit (INT4) fixed-point, which are suitable for hardware platforms with native support for low-precision arithmetic, such as the INT8-enabled mobile deep-learning accelerators by Qualcomm [49], Nvidia [50, 51], Arm [52] and Samsung [53], and INT4 on Nvidia's Turing GPUs [54, 55]. As shown in Figure 9(a), a bit-depth of 16 bits follows closely the behavior of the original, non-quantized models across all types of attacks.



(a) Varying quantization levels



(b) Varying perturbations size

Fig. 9. Comparison of our proposed approach Noisy Factorization (NFact) with Defensive Distillation (DefDist), Data Augmentation (Aug), and Liu et al., on the Speech Command dataset (SC) for both FGSM and PGD(20) attacks. The baseline model corresponds to a compressed version of ETE$_{SC}$ and all models have the same number of parameters for a meaningful comparison. The plots on the left measure the effect quantization levels (4, 8, 16 and 32 bits) have on the error rate for a fixed radius $\epsilon = 0.001$ for the adversarial perturbations, whereas the plots on the right measure the effect the maximum allowed magnitude of adversarial perturbations ($\epsilon = 0, 0.001, 0.005, 0.01, 0.05$) have on the error rate for a fixed bit depth (32 bits).

For the more aggressively quantized models (INT8 and INT4), we observe the quantized models becoming more vulnerable to adversarial attacks, potentially due to the reduction in model capacity.

5.3 Improving Robustness of Models

In this section, we present results of the robust training pipeline presented in §3. First, we show that the robust training process generates smoother loss surfaces. Accordingly, we provide a low-dimensional visualization of the loss surfaces learned through normal training and through the proposed robust audio training pipeline in Figure 10. For example, the loss surfaces of ETE$_{SC}$, LM$_{SC}$ and LMS$_{SC}$ are respectively shown[2] in 10(a), 10(b) and 10(c) for a specific audio input. We further present their robust counterparts below each model, e.g., in 10(d), 10(e) and 10(f) for the same input. In detail, the x-axis (Grad. dir.) represents the direction of the gradient of the loss with respect to one particular input which is correctly classified, whereas the y-axis (Orth. to Grad.) represents a direction within the high-dimensional space, which is orthogonal to the direction of the gradient. Furthermore, the x-y plane represents the corresponding low-dimensional projection of a $\|\cdot\|_\infty$-ball with radius $\epsilon = 0.1$ around a particular input. Of particular importance is the z-axis, which denotes the magnitude of the loss function. For the normal models trained with conventional approaches (upper row), the loss surfaces are seen to be very sharply rising to high values for the smallest of perturbations, making them easy to attack with adversarial examples; on the other hand, the loss for the robust variants learned with the proposed audio training pipeline (bottom row) remain rather flat, making it difficult to find an adversarial example that would lead to an incorrect classification, showcasing the effectiveness of the proposed light-weight low-rank stochastic robustness technique presented in §3.

While Figure 10 provides a comparison between normal training and our proposed audio training pipeline, Figure 9(b) refines this analysis by introducing three additional baselines commonly used to combat adversarial attacks: defensive distillation, data augmentation and a method proposed by Liu *et al.* [40] (see §4.3). For a wider view, Figure 9(b) provides results ranging from virtually inaudible attacks corresponding to $\epsilon = 0.001$ to louder but still unrecognizable attempts to corrupt the audio system corresponding to $\epsilon = 0.05$. As can be seen from

[2] In Figure 10 we dropped the suffix *SC*, indicating that the model is trained on the Speech command dataset, for simplicity in presentation.

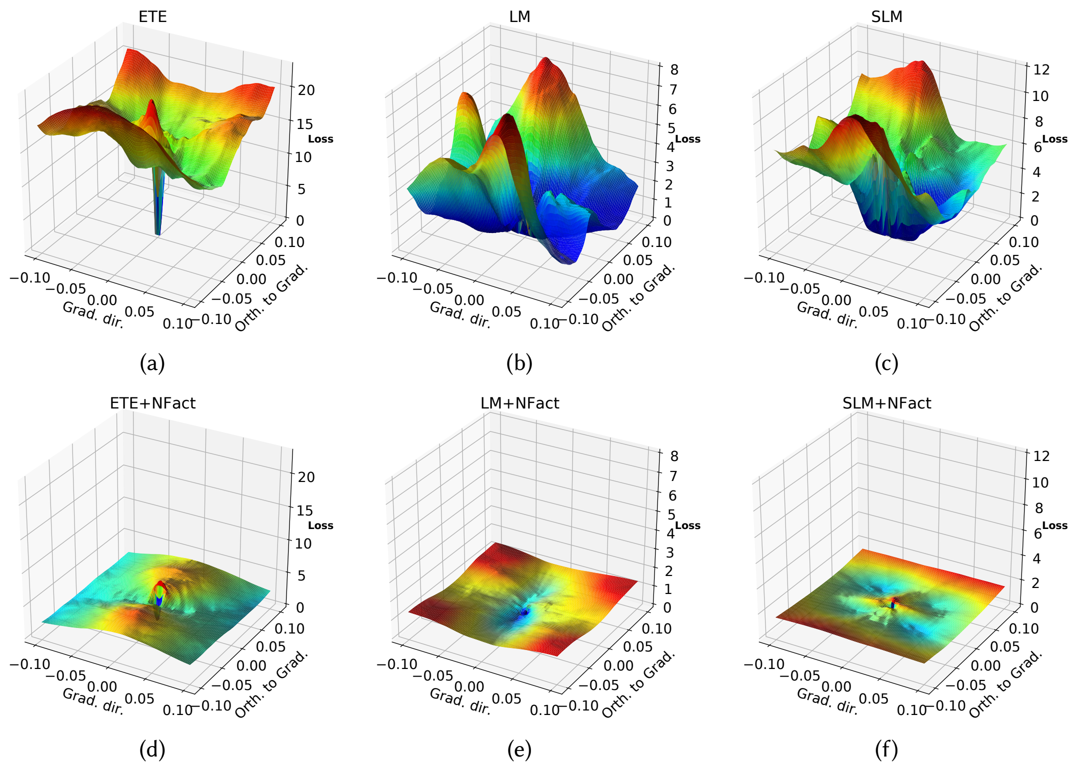


Fig. 10. Low-dimensional visualisation of the loss surfaces. The top row, i.e., sub-figure (a), (b) and (c) respectively shows the loss surfaces for ETE_{SC}, LM_{SC} and LMS_{SC}. The bottom row, i.e., sub-figure (d), (e) and (f) respectively shows the loss surfaces after robust training through the proposed audio training pipeline. *Grad. dir.* is the direction of the gradient at the data point, and *Orth. to Grad.* is an orthogonal direction to the gradient. The Z -axis represents the loss value evaluated on the points in plane spanned by the two vector directions selected.

Figure 9(b), our method yields significant defence capabilities, better than those provided by existing approaches for FGSM attacks. For the PGD algorithm, however, the performance of model stochastisty degrades for higher values of ϵ. Note that our proposed approach continues to work very well when no perturbation is added to the the input, thereby maintaining the usability of the model under a non-attacking everyday setting, which is an important property of any robust training pipeline.

The stochasticity inherent to our model will generate different output probabilities when the same input is provided. To overcome the effect of this variation, we take an ensemble approach, inline with the work of Liu *et al.* [40]. To understand the impact of the ensemble, in Figure 8 we measure the overall error with different number of ensemble size for three models trained on the Speech command (SC) dataset. We found that a small ensemble size of around 3, is good enough to achieve good overall performance, which makes IoT deployment feasible.

5.4 Real-world Attacks: A Case Study

In this section, we investigate the feasibility of real-world audio adversarial attacks and the effectiveness of the presented countermeasures in practical applications. This part of the evaluation allows us to elucidate on the significant challenges overlooked in software-simulation and purely offline dataset-based analysis of adversarial robustness of DNNs. Challenges include: signal transformations due to hardware, variance in recording quality due to relative positioning of speaker and microphone devices, and perturbation-signal mis-synchronization unavoidable for audio signals in real physical environments.

Scenario: *An adversary attempts to attack a compressed DNN deployed to a home device to recognize short spoken commands (e.g., "start", "stop").* We employed FGSM and black-box algorithm for generating adversarial examples. We performed two categories of attacks:
(i) *untargeted*, abbreviated in the following as FGSM and BB for the two attack algorithms respectively, to assess the feasibility of denial of service (DoS) over the air, and
(ii) *targeted*, abbreviated as TFGSM and TBB respectively, to simulate scenarios of voice-interface manipulation towards an adversarial goal.

Data: From the test set of *Speech Command* data, we extracted a random subset of 128 audio samples of 1 sec duration each, from commands *stop*, *off* and *no*. For targeted attacks we aimed to map *stop*, *off* and *no* to *go*, *on* and *yes* respectively. Random subsampling of *Speech Command* dataset allowed us to evaluate our methods on an unbiased set comprised of speakers with diverse demographics and different input signals, which can be encountered in real-world voice-command interfaces.

Models: We evaluated our attacks on DNN architectures trained both on raw features and log-mel frequencies. Namely, we experimented with ETE_{SC} and LM_{SC} models as defined and trained in §4 – dataset-specific subscripts are omitted in the exposition below for brevity, as we always refer to the model architecture trained on the Speech command dataset. Moreover, we assessed their robust counterparts, NFacETE and NFacLM, respectively, after setting regularization hyperparameter $\alpha = 0.01$.

Noise levels: Consistently with experiments in previous sections, for recordings involving adversarial noise, we set SNR to 15dB. Empirically, this loudness level allowed adversarial noise to reach our recording devices in all tested placements, while being an unrecognisable signal perceptually resembling low background noise.

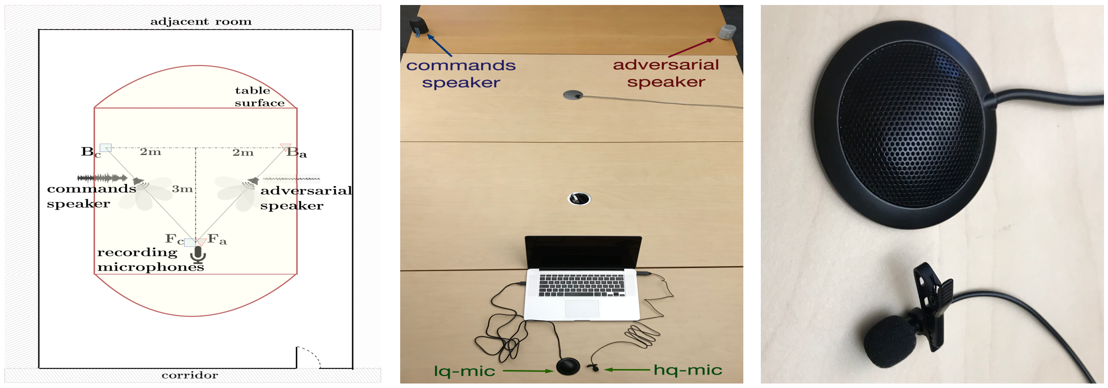
Adversarial perturbations: According to the respective recognition models, adversarial perturbations δ : $||\delta||_\infty \leq 0.1$ were selected from the corpus of perturbations generated for experiments in §4.

Method: As illustrated in Figure 11a, we carried out our experiment inside a medium-size room. We used the following devices:

- Throughout the experiment *two microphones* of varying quality were placed on a table at the positions seen in Figure 11b. Used microphones are displayed in Figure 11c.
- *Two speakers* were placed at various positions on the table, as detailed in Figure 11a: *Commands speaker* is used for playing audio commands, e.g., as in the *Speech Command* dataset. A separate speaker, termed in the following as *adversarial speaker*, is used to transmit adversarial perturbations.

At first, we measured the baseline real-world performance of the trained models by recording on the microphones clean audio played from the commands speaker, while adversarial speaker remained muted. This was repeated for the speaker collocated with the microphones (position F_c) and speaker placed at distance (position B_c). Secondly, we turned on the adversarial speaker, setting SNR to 15dB, and repeated the recordings of the superimposed signals. Successively, we moved commands and adversarial speakers in the foreground and background (positions F_c, B_c and F_a, B_a respectively) obtaining a total of 4 configurations for their relative positions. The configuration with both speakers in the background is displayed in Figure 11b. All audio experiments were carried out at a 44.1KHz sampling frequency.

Challenges: In particular, this real-world experiment measures the impact of the following two challenges present in physical-world attacks:



(a) (b) (c)

Fig. 11. (a) Experimental setup: A set of speech commands is played over the air from a speaker successively placed in the foreground (position F_c) and background (position B_c). Audio from *commands speaker* is superimposed by an perturbation played from an *adversarial speaker* successively placed at positions F_a and B_a. The resulting combination sound is recorded by two microphones of different recording quality placed in the foreground. (b) Snapshot of the experiment with commands and adversarial speakers placed in the background (at positions B_c and B_a respectively). (c) Closeup to the two recording devices used: high-quality microphone (*hq-mic*) at the bottom and low-quality microphone (*lq-mic*) on top.

(1) *Varying positions of commands source, attack device and recognition system.* In real-world situations, the variations in angle and distance of the voice commands source from the recording device can potentially impact the quality of recording. Relative position also affects the adversarial property of the perturbation, as it can cause mis-synchronisation of the two signals.
(2) *Physical limitations of sensing technology.* Before reaching the audio recognition component, input captured by sensory systems undergoes transformations and is affected by hardware imperfections. We study this impact on adversarial perturbations.

Results: Table 2 demonstrates real-world recognition performance under no adversarial attack for commands speaker playing in the foreground and background. This performance provides a baseline for the evaluation of our system's robustness when under attack, as seen in results presented in the remainder of the section. When commands are played in the foreground (position F_c), recognition performance on recorded samples is lower compared to our software only dataset simulations. This effect can be attributed to the physical limitations of our audio system. Model performance gets significantly degraded for both microphones when the commands speaker is moved to the background (position B_c). This is an entangled effect of both challenges 1 and 2: relative distance implies some further attenuation of the recorded signal, while in extreme cases transmission delay might also induce truncation to the recorded audio. Training on the log-mel spectrum offered slight enhancement of performance in both positions for the two microphones. Robust training performance in the absence of attack was akin to normal training. Interestingly, in the case of our lower-quality microphone, when commands speaker is placed at distance, robust training offered reduction of error rate to half. In this part of the experiment, enhancement of performance in the robust model is mostly due to the random perturbations term included in the regularizer of loss function. Overall deploying a robust model in the absence of adversarial noise offered accuracy varying from 67% to 93% at all positions of commands speaker for the two microphones.

In Table 3, we summarize the real-world attack results, after turning on the adversarial speaker. The order of rows in this table (from top to bottom) provides adversarial cases of increasing difficulty, beginning with a commands

Table 2. Performance of command recognition for different models and placements of commands speaker in the absence of adversarial perturbations, using the (a) higher (*hq-mic*) and (b) lower (*hq-mic*) quality microphones.

Speaker Position	hq-mic (error in %)				lq-mic (error in %)			
	ETE	NFacETE	LM	NFacLM	ETE	NFacETE	LM	NFacLM
F_c	9.18	9.67	6.45	6.93	15.64	13.09	10.64	12.79
B_c	35.16	33.01	16.70	22.85	61.62	30.18	59.77	29.30

Table 3. Performance of command recognition on recordings in the presence of adversarial perturbations for all models and placements of commands (*comm*) and adversarial (*adver*) speakers, using the (a) higher (*hq-mic*) and (b) lower (*lq-mic*) quality microphones tested in our real-world experiment. Presented results for models trained on raw features (*ETE*) and log-mel frequencies (*LM*), and their robustly trained counterparts (*NFacETE*, *NFacLM* resp.), and untargeted and targeted attacks via Fast Gradient Sign Method and natural evolution strategies based Black-box attack (*FGSM*, *TFGSM*, *BB* and *TBB* resp.)

Speakers Positions		hq-mic (error in %)															
comm	*adver*	ETE				NFacETE				LM				NFacLM			
		FGSM	TFGSM	BB	TBB	FGSM	TFGSM	BB	TBB	FGSM	TFGSM	BB	TBB	FGSM	TFGSM	BB	TBB
F_c	B_a	25.00	0.78	14.06	2.34	26.56	2.34	12.50	0	26.56	1.56	14.06	0.78	31.25	3.91	10.94	2.34
B_c	B_a	79.69	3.91	60.16	4.69	72.66	6.25	39.06	3.91	74.22	5.47	49.22	9.38	77.34	14.84	32.81	10.94
F_c	F_a	59.38	4.69	62.50	9.38	60.16	13.28	46.88	1.56	80.47	0	68.75	25.00	87.50	12.50	60.16	18.75
B_c	F_a	92.19	24.22	83.59	32.03	76.56	2.34	94.53	30.47	82.03	10.94	82.81	25.00	92.19	22.66	92.19	33.59
Speakers Positions		lq-mic (error in %)															
comm	*adver*	ETE				NFacETE				LM				NFacLM			
		FGSM	TFGSM	BB	TBB	FGSM	TFGSM	BB	TBB	FGSM	TFGSM	BB	TBB	FGSM	TFGSM	BB	TBB
F_c	B_a	30.47	0.78	25.78	2.34	24.22	0.78	19.53	0.78	28.12	1.56	27.34	2.34	28.91	2.34	21.09	2.34
B_c	B_a	85.94	3.91	67.97	3.12	68.75	3.12	67.19	2.34	81.25	3.12	53.12	7.03	60.16	0.78	45.31	3.12
F_c	F_a	85.94	14.06	75.78	13.28	71.88	1.56	66.41	10.16	85.94	7.03	81.25	22.66	92.97	20.31	74.22	18.75
B_c	F_a	97.66	24.22	92.19	22.66	97.66	14.06	98.44	17.19	89.06	7.81	91.41	18.75	98.44	22.66	96.88	32.03

speaker collocated with the recording device, while adversarial speaker is placed in the background, and ending with the opposite configuration (F_c, B_a). Error caused by untargeted attacks in the first position ranges between 10% and 31%, while, when adversarial speaker is placed next to microphone and commands speaker is in the background, error takes values between 76% and 98%.

Targeted attacks are significantly more difficult in the real-world: superimposing targeted perturbation to the signal should guide the original audio to a smaller region of the input space (corresponding to the target class) compared to an untargeted perturbation (which simply moves the input out of its true class region). Thus, mis-synchronisation and signal transformation effects due to transmission and hardware cause more significant degradation of targeted attacks performance. Indeed, for the easiest adversarial scenario, namely commands speaker next to the recording device and adversarial speaker at distance, we are able to achieve a maximum of 33% prediction error in targeted attacks. We have to note here that this range of attacks is expected to present an additional degree of variability, related to which target class we choose for each of our input classes, which we didn't consider exhaustively in the present experiment.

Placing the commands and adversarial speakers both at distance is slightly less effective for real-world adversarial attacks compared to keeping both collocated with the microphones, due to the reduced recording quality effects we mentioned above. FGSM is generally more effective compared to black-box attack. Considering that natural evolution strategies based perturbations are computed following stochastic gradients, our expectation is that under the black-box assumption, input is led closer to the decision boundary compared to FGSM, hence the effects of quality degradation due to real-world recording seem to be more important for the black-box attack. Moreover, in the presence of adversarial noise, we observed that there is no clear improvement pattern when replacing a model trained on raw audio features with one trained on log-mel spectrum coefficients.

Effectiveness of robust training is more prominent for placements that minimize mis-synchronization of commands and adversarial perturbations, e.g. at positions (B_c, B_a) and (F_c, F_a), offering error reduction that can exceed 15%. Our findings indicate that real-world temporal effects should be also taken into account in the regularizations used for robust training.

Our results show that placing the commands speaker closer to the microphone causes lower untargeted attack efficacy. However, regardless of the spatial arrangement of commands and adversarial speakers, under low SNR, untargeted black-box and FGSM attacks can achieve high success rates, constituting a realistic threat and highlighting severe vulnerabilities of audio models deployed on embedded and IoT platforms. Hence, enhancing neural network training with adversarial robustness is a necessity for security critical audio applications.

In a significant part of the experiments presented in Table 3 we found that the low-rank stochastic compressed models perform better than the uncompressed models, although there are some instances where we do not see this improvement. The main reasons for this might include: (i) the lack of perfect synchronization between the commands and adversarial perturbations, especially in the case of targeted attacks, (ii) incurred distortions due to multi path reflections and other distortions during transmission, which adversely affect the quality of the transmitted perturbations, (iii) the non-linearity present on the microphone and the speaker used in the experiments, and (iv) poor sensitivity of the low-quality microphone and noise during audio recording. Thus, more research is needed to understand these real-world factors and their effects on carefully constructed small adversarial perturbations for successful practical audio-adversarial attacks.

6 RELATED WORK

Szegedy *et al.* [20] first noticed the vulnerability of deep image classifiers to imperceptible adversarial perturbations in model's input space. Since then, a growing body of techniques in attacking and improving the robustness of DNNs in various adversarial settings have been proposed.

The existence of adversarial examples poses significant threats to safety-critical applications of deep learning models in real-world systems. The practical deployment of adversarial attacks in the physical world was first explored in the image domain in [56], where it was shown that adversarial images constructed in software, when printed on paper and sensed by a phone camera, could fool phone apps for image classification. Subsequently, [16, 57] noted that the effectiveness of printed adversarial images can be sensitive to camera distances and angles. Robustifying adversarial examples in 2 and 3 dimensions for real-world visual recognition systems has been the focus of [58], where authors incorporate realistic transformations (e.g. rotation, translation, varying lighting conditions) in their white-box perturbations and verify their viability in the physical world.

In the audio domain, Yakura [59] *et al.* recently proposed a method for generating robust adversarial examples for over-the-air audio attacks, via introducing band-pass filters, environment's impulse response and Gaussian noise that can simulate transformation caused by playback. Contrary to ours, their approach is tailored to recurrent speech recognition models [60], where reverberation effects are more pronounced, and uses multiple playback cycles, which make perturbation crafting more costly. Moreover, in our work we aim to optimize countermeasures against threats in real-world systems, hence we emphasize on developing strategies that make systems robust against adversarial inputs in a variety of scenarios, rather than enhancing real-world robustness of adversarial examples. Exploiting the gap in perception mechanisms between human and machine speech recognition [61], Carlini *et al.* [17] successfully crafted unintelligible audio that is interpreted as commands by the voice-interfaces of mobile devices. Further, it has been shown that voice processing systems can be maliciously manipulated by inaudible sounds using ultrasonic carriers [62] and psychoacoustic hiding [63]. In addition, recent work [18] has demonstrated high success rates in white-box attacks towards automatic speech-to-text transcription systems. As opposed to above mentioned works, in our study we do not emphasize on optimizing the spectrum of acoustic perturbation to human listening, avoiding the extra training cost due to frequency constraints. However, we are still able to keep low distortion levels by bounding the $|| \cdot ||_\infty$-norm of the perturbation vector and evaluate our methods with low SNR. Moreover, we consider a wider range of constrained attack scenarios, including black-box access to recognition system. We note that our findings on architecture type effects and robust training seem to be transferable in principle to inaudible automatic speech recognition attacks literature.

Constructing adversarial examples under threat models that assume only loss-oracle access to image and audio recognition system has been tackled in previous work, using *genetic algorithms* [64], *gradient estimation* methods [48, 65], or a combination of both [66]. Aiming to optimize adversarial examples generation for budgeted attacks, in this study we adapted the natural evolution strategies based method [48]. We found empirically that this algorithm allows high rates of success in adversarial generation by allowing a (reasonable for existing methods) budget of $2 \times 64 \times 20 = 2560$ queries per sample. In work contemporary to ours [67], state-of-the-art results has been achieved by *bandit optimization*, which we leave as a direction for future research.

The deployability of DNNs to resource-constrained embedded devices has initiated a line of research in portability of adversarial examples under model compression. The work in [68] looks at the effect of the HashNet compression technique on adversarial image classifiers, but does not deploy their compressed networks onto embedded devices and hence does not investigate the resilience of the trained models on practical real-world adversarial scenarios. Zhao *et al.* [69] demonstrated that adversarial examples of images remain transferable under pruning, while transferability under quantisation is highly sensitive to precision.

In [41] it was observed that, if the input dimension is huge, even tiny perturbations, distinguishing an original from an adversarial image, can accumulate into an inner product with a weight vector to produce a huge distortion which sets the DNN astray onto a wrong classification. Subsequently, work in [70, 71] proposes to combat such dimensionality phenomenon with a sparsifying front end which projects the input towards a lower dimensional space. Alternatively, [72] incorporates a subnetwork trained to distinguish legitimate from adversarial images.

While a typical adversarial attack focuses on the construction of a small perturbation tailored to misclassify a specific input, [73] seeks one perturbation that fools a fixed classifier on most natural images. To mitigate such attacks, [74] puts forth a mechanism which not only infers the presence of such universal perturbations but that also corrects the distorted images in order to correctly predict their labels. Away from the multi-class domain, [75] focusses on speech to text attacks, where, given a chosen text as intended output, the attacker attempts to generate an audio input signal with small $\| \cdot \|_\infty$ norm which is difficult to characterise by humans but is easily recognised by the victim's device (running Google Now or Siri) as an alternative predefined text chosen by the attacker. Black- and white-box attacks, with and without background noise are reported but no compression techniques are investigated, as a means to understand their robustness properties.

7 CONCLUSION

In this paper we showed that it is possible to attack the types of compressed audio DNNs that have gained popularity for enabling audio intelligence on embedded devices and therefore that the multitude of voice-enabled embedded devices around us are vulnerable to such attacks. To counteract these vulnerabilities, we proposed a new training methodology that yields compressed models with high resistance to adversarial attacks. This is achieved by introducing a new light-weight low-rank stochastic model ensemble together with a novel consistency loss. Both are leveraged during training together with standard cross entropy and adversarial losses to increase the overall model robustness. We followed through with an extensive set of software and real-world experimental results, which showed that our training pipeline significantly improves the robustness of audio models over existing solutions such as defensive distillation and data augmentation.

ACKNOWLEDGMENTS

We would like to thank Fahim Kawsar, Timothy Hospedales, and Samin Ishtiaq for giving comments on the earlier version of the draft. This work is partially supported by Nokia Bell Labs through their donation for the Centre of Mobile, Wearable Systems and Augmented Intelligence to the University of Cambridge. Early research on this work was done by Sourav when he was employed at Nokia Bell Labs.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press Cambridge, 2016, vol. 1.
[3] https://developer.amazon.com/en-US/alexa/alexa-skills-kit [Retrieved: May 13, 2020].
[4] https://biztechmagazine.com/article/2018/11/voiceprint-security-game-changer-banks-and-credit-unions-all-sizes [Retrived: May 13, 2020].
[5] https://www.apple.com/uk/ios/siri/ [Retrieved: May 13, 2020].
[6] https://store.google.com/gb/product/google_home_mini [Retrieved: May 13, 2020].
[7] https://developer.amazon.com/alexa [Retrieved: May 13, 2020].
[8] V. Sze, Y. Chen, T. Yang, and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec 2017.
[9] A. Zhou *et al.*, "Incremental Network Quantization: Towards Lossless CNNs with Low-Precision Weights," in *International Conference on Learning Representations (ICLR)*, 2017.
[10] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks," in *European Conference on Computer Vision (ECCV)*, Cham, 2016, pp. 525–542.
[11] K. Guo, L. Sui, J. Qiu, J. Yu, J. Wang, S. Yao, S. Han, Y. Wang, and H. Yang, "Angel-Eye: A Complete Design Flow for Mapping CNN Onto Embedded FPGA," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 37, no. 1, pp. 35–47, 2018.
[12] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
[13] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, and F. Kawsar, "An Early Resource Characterization of Deep Learning on Wearables, Smartphones and Internet-of-Things Devices," in *Proceedings of the 2015 International Workshop on Internet of Things towards Applications*. ACM, 2015, pp. 7–12.
[14] S. Bhattacharya and N. D. Lane, "Sparsification and separation of deep learning layers for constrained resource inference on wearables," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2016.
[15] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.
[16] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust Physical-World Attacks on Deep Learning Visual Classification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
[17] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. A. Wagner, and W. Zhou, "Hidden Voice Commands," in *USENIX Security Symposium, USENIX Security 16*, 2016, pp. 513–530.
[18] N. Carlini and D. Wagner, "Audio Adversarial Examples: Targeted Attacks on Speech-to-Text," in *2018 IEEE Security and Privacy Workshops (SPW)*, 2018, pp. 1–7.
[19] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 39–57.
[20] 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.
[21] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal, "Enhancing robustness of machine learning systems via data transformations, arxiv preprint," in *52nd Annual Conference on Information Sciences and Systems (CISS)*, 2018.
[22] S. Wachter, B. D. Mittelstadt, and C. Russell, "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR," *Harv. JL & Tech.*, vol. 31, p. 841, 2017.
[23] 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 (ICLR)*, 2018.
[24] C. Tai, T. Xiao, Y. Zhang, X. Wang, and W. E, "Convolutional neural networks with low-rank regularization," in *International Conference on Learning Representations (ICLR)*, 2016.

[25] F. Chollet, "Xception: Deep Learning With Depthwise Separable Convolutions," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," in *NIPS Deep Learning Workshop*, 2015.

[27] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in *International Conference on Learning Representations (ICLR)*, 2016.

[28] N. D. Lane, S. Bhattacharya, A. Mathur, P. Georgiev, C. Forlivesi, and F. Kawsar, "Squeezing Deep Learning into Mobile and Embedded Devices," *IEEE Pervasive Computing*, vol. 16, no. 3, pp. 82–88, 2017.

[29] S. I. Venieris, A. Kouris, and C.-S. Bouganis, "Deploying Deep Neural Networks in the Embedded Space," in *2nd International Workshop on Embedded and Mobile Deep Learning (EMDL)*, 2018.

[30] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, L. Jiao, L. Qendro, and F. Kawsar, "DeepX: A Software Accelerator for Low-Power Deep Learning Inference on Mobile Devices," in *International Conference on Information Processing in Sensor Networks (IPSN)*, 2016.

[31] M. Rizakis, S. I. Venieris, A. Kouris, and C.-S. Bouganis, "Approximate FPGA-based LSTMs under Computation Time Constraints," in *14th International Symposium on Applied Reconfigurable Computing (ARC)*. Springer, 2018, pp. 3–15.

[32] Łukasz Dudziak, M. S. Abdelfattah, R. Vipperla, S. Laskaridis, and N. D. Lane, "ShrinkML: End-to-End ASR Model Compression Using Reinforcement Learning," in *Proc. Interspeech 2019*, 2019, pp. 2235–2239.

[33] C. Tai, T. Xiao, Y. Zhang, X. Wang, and W. E, "Convolutional Neural Networks with Low-Rank Regularization," in *International Conference on Learning Representations (ICLR)*, 2016.

[34] A. Kouris, S. I. Venieris, and C. Bouganis, "CascadeCNN: Pushing the Performance Limits of Quantisation in Convolutional Neural Networks," in *28th International Conference on Field Programmable Logic and Applications (FPL)*, 2018, pp. 155–1557.

[35] P. Gysel, J. Pimentel, M. Motamedi, and S. Ghiasi, "Ristretto: A Framework for Empirical Study of Resource-Efficient Inference in Convolutional Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, vol. 29, no. 11, pp. 5784–5789, Nov 2018.

[36] A. Mishra, E. Nurvitadhi, J. J. Cook, and D. Marr, "WRPN: Wide Reduced-Precision Networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[37] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry, "Adversarially robust generalization requires more data," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 5019–5031.

[38] H. Kannan, A. Kurakin, and I. Goodfellow, "Adversarial logit pairing," *arXiv preprint arXiv:1803.06373*, 2018.

[39] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," 2019.

[40] X. Liu, M. Cheng, H. Zhang, and C.-J. Hsieh, "Towards Robust Neural Networks via Random Self-ensemble," in *European Conference on Computer Vision (ECCV)*, 2018.

[41] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in *International Conference on Learning Representations (ICLR)*, 2015.

[42] A. Al-Dujaili, S. Srikant, E. Hemberg, and U.-M. O'Reilly, "On the Application of Danskin's Theorem to Derivative-Free Minimax Optimization," in *AIP Conference Proceedings*, vol. 2070, no. 1, 2019, p. 020026.

[43] J. Gibson, M. V. Segbroeck, and S. S. Narayanan, "Comparing time-frequency representations for directional derivative features," in *INTERSPEECH*, 2014.

[44] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," *arXiv e-prints*, p. arXiv:1804.03209, Apr 2018.

[45] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time-frequency representations for audio scene detection," Technical report, HAL, https://sites.google.com/site/alainrakotomamonjy/home/audio-scene, 2014.

[46] Y. Aytar, C. Vondrick, and A. Torralba, "SoundNet: Learning Sound Representations from Unlabeled Video," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[47] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural Evolution Strategies," *Journal of Machine Learning Research (JMLR)*, vol. 15, no. 1, pp. 949–980, 2014.

[48] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box Adversarial Attacks with Limited Queries and Information," in *Proceedings of the 35th International Conference on Machine Learning, (ICML)*, 2018, pp. 2142–2151.

[49] L. Codrescu, W. Anderson, S. Venkumanhanti, M. Zeng, E. Plondke, C. Koob, A. Ingle, C. Tabony, and R. Maule, "Hexagon DSP: An Architecture Optimized for Mobile Multimedia and Communications," *IEEE Micro*, vol. 34, no. 2, pp. 34–43, 2014.

[50] J. Choquette, O. Giroux, and D. Foley, "Volta: Performance and Programmability," *IEEE Micro*, vol. 38, no. 2, pp. 42–52, 2018.

[51] Nvidia, "Nvidia Deep Learning Accelerator (NVDLA)," http://nvdla.org/, [Retrieved: May 13, 2020].

[52] Arm, "Arm Machine Learning Processor," https://developer.arm.com/ip-products/processors/machine-learning/arm-ml-processor, [Retrieved: May 13, 2020].

[53] J. Song, Y. Cho, J. Park, J. Jang, S. Lee, J. Song, J. Lee, and I. Kang, "7.1 An 11.5TOPS/W 1024-MAC Butterfly Structure Dual-Core Sparsity-Aware Neural Processing Unit in 8nm Flagship Mobile SoC," in *International Solid-State Circuits Conference (ISSCC)*, 2019, pp. 130–132.

[54] J. Burgess, "RTX ON – The NVIDIA TURING GPU," in *2019 IEEE Hot Chips 31 Symposium (HCS)*, 2019, pp. 1–27.

[55] M. Almeida, S. Laskaridis, I. Leontiadis, S. I. Venieris, and N. D. Lane, “EmBench: Quantifying Performance Variations of Deep Neural Networks Across Modern Commodity Devices,” in *The 3rd International Workshop on Deep Learning for Mobile Systems and Applications (EMDL)*, 2019, pp. 1–6.

[56] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” *arXiv preprint arXiv:1607.02533*, 2016.

[57] J. Lu, H. Sibai, E. Fabry, and D. A. Forsyth, “NO Need to Worry about Adversarial Examples in Object Detection in Autonomous Vehicles,” *arXiv preprint arXiv:1707.03501*, 2017.

[58] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, “Synthesizing Robust Adversarial Examples,” in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018, pp. 284–293.

[59] H. Yakura and J. Sakuma, “Robust Audio Adversarial Example for a Physical Attack,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, 2019, pp. 5334–5341.

[60] A. Y. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, “Deep Speech: Scaling up end-to-end speech recognition,” *CoRR*.

[61] T. Vaidya, Y. Zhang, M. Sherr, and C. Shields, “Cocaine Noodles: Exploiting the Gap between Human and Machine Speech Recognition,” in *9th USENIX Workshop on Offensive Technologies (WOOT)*, 2015.

[62] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, “DolphinAttack: Inaudible Voice Commands,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 103–117.

[63] L. Schönherr, K. Kohls, S. Zeiler, T. Holz, and D. Kolossa, “Adversarial Attacks Against Automatic Speech Recognition Systems via Psychoacoustic Hiding,” in *Network and Distributed Systems Security (NDSS) Symposium*.

[64] M. Alzantot, B. Balaji, and M. B. Srivastava, “Did you hear that? Adversarial Examples Against Automatic Speech Recognition,” in *NIPS 2017 Machine Deception Workshop*, 2017.

[65] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “ZOO: Zeroth Order Optimization Based Black-Box Attacks to Deep Neural Networks without Training Substitute Models,” in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security (AISec)*. ACM, 2017, pp. 15–26.

[66] R. Taori, A. Kamsetty, B. Chu, and N. Vemuri, “Targeted Adversarial Examples for Black Box Audio Systems,” in *2019 IEEE Security and Privacy Workshops (SPW)*, 2019, pp. 15–20.

[67] A. Ilyas, L. Engstrom, and A. Madry, “Prior Convictions: Black-box Adversarial Attacks with Bandits and Priors,” in *International Conference on Learning Representations (ICLR)*, 2019.

[68] Q. Liu, T. Liu, Z. Liu, Y. Wang, Y. Jin, and W. Wen, “Security Analysis and Enhancement of Model Compressed Deep Learning Systems under Adversarial Attacks,” in *Proceedings of the 23rd Asia and South Pacific Design Automation Conference (ASPDAC)*, 2018, p. 721–726.

[69] Y. Zhao, I. Shumailov, R. Mullins, and R. Anderson, “To compress or not to compress: Understanding the Interactions between Adversarial Attacks and Neural Network Compression,” in *MLSys*, 2018.

[70] S. Gopalakrishnan, Z. Marzi, U. Madhow, and R. Pedarsani, “Combating Adversarial Attacks Using Sparse Representations,” in *ICLR Workshop*, 2018.

[71] Z. Marzi, S. Gopalakrishnan, U. Madhow, and R. Pedarsani, “Sparsity-based Defense against Adversarial Attacks on Linear Classifiers,” in *IEEE International Symposium on Information Theory (ISIT)*, 2018.

[72] J. Hendrik Metzen, T. Genewein, V. Fischer, and B. Bischoff, “On Detecting Adversarial Perturbations,” in *International Conference on Learning Representations (ICLR)*, 2017.

[73] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal Adversarial Perturbations,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[74] N. Akhtar, J. Liu, and A. Mian, “Defense Against Universal Adversarial Perturbations,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[75] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou, “Hidden Voice Commands,” in *25th USENIX Security Symposium (USENIX Security 16)*, Austin, TX, 2016, pp. 513–530.