SirenAttack: Generating Adversarial Audio for End-to-End Acoustic Systems

Tianyu Du
Zhejiang University
zjradty@zju.edu.cn

Shouling Ji*
Zhejiang University
sji@zju.edu.cn

Jinfeng Li
Zhejiang University
lijinfeng_0713@zju.edu.cn

Qinchen Gu
Georgia Institute of Technology
qgu7@gatech.edu

Ting Wang
Pennsylvania State University
inbox.ting@gmail.com

Raheem Beyah
Georgia Institute of Technology
rbeyah@ece.gatech.edu

ABSTRACT

Despite their immense popularity, deep learning-based acoustic systems are inherently vulnerable to adversarial attacks, wherein maliciously crafted audios trigger target systems to misbehave. In this paper, we present SirenAttack, a new class of attacks to generate adversarial audios. Compared with existing attacks, SirenAttack highlights with a set of significant features: (i) versatile – it is able to deceive a range of end-to-end acoustic systems under both white-box and black-box settings; (ii) effective – it is able to generate adversarial audios that can be recognized as specific phrases by target acoustic systems; and (iii) stealthy – it is able to generate adversarial audios indistinguishable from their benign counterparts to human perception. We empirically evaluate SirenAttack on a set of state-of-the-art deep learning-based acoustic systems (including speech command recognition, speaker recognition and sound event classification), with results showing the versatility, effectiveness, and stealthiness of SirenAttack. For instance, it achieves 99.45% attack success rate on the IEMOCAP dataset against the ResNet18 model, while the generated adversarial audios are also misinterpreted by multiple popular ASR platforms, including Google Cloud Speech, Microsoft Bing Voice, and IBM Speech-to-Text. We further evaluate three potential defense methods to mitigate such attacks, which leads to promising directions for further research.

CCS CONCEPTS

• **Security and privacy** → *Usability in security and privacy.*

KEYWORDS

Adversarial Machine Learning, Neural Network, Speech Recognition

ACM Reference Format:
Tianyu Du, Shouling Ji, Jinfeng Li, Qinchen Gu, Ting Wang, and Raheem Beyah. 2020. SirenAttack: Generating Adversarial Audio for End-to-End Acoustic Systems. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (ASIA CCS '20), October 5–9, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3320269.3384733

*Shouling Ji is the corresponding author.

ASIA CCS '20, October 5–9, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-6750-9/20/10…$15.00
https://doi.org/10.1145/3320269.3384733

1 INTRODUCTION

Nowadays machine learning-powered acoustic systems are ubiquitous in our everyday lives, ranging from smart locks on mobiles to speech assistants on smart home devices and to machine translation services on clouds. In general, acoustic systems can be categorized into two types according to application scenarios: classification-oriented systems and recognition-oriented systems. A classification-oriented acoustic system typically first transforms the audios from time domain to frequency domain and then performs classification on the corresponding spectrograms. As an example, a sound event classification system, which is often integrated into acoustic surveillance systems [1, 3], recognizes physical events such as glass breaking and gunshot. Compared with classification-oriented acoustic systems, a recognition-oriented acoustic system is often more complicated since it needs to first segment audios into frames, perform prediction on each frame, and then derive the recognition results based on Connectionist-Temporal-Classification (CTC) loss [18] or attention [4]. The most typical example is the Automatic Speech Recognition (ASR) system, which is widely integrated into various popular speech assistants (e.g., Siri, Google Now).

Due to their superior performance, most of today's acoustic systems are built upon deep neural network models. However, such models are inherently vulnerable to adversarial inputs, which are maliciously crafted samples (typically by adding human-imperceptible noise to legitimate samples) to trigger target models to misbehave [17, 47]. Despite the plethora of work on the image domain [24, 28, 32, 34, 45, 56] and text domain [30, 31], the research of adversarial attacks on the audio domain is still limited, due to a number of non-trivial challenges. First, the acoustic systems need to deal with information changes in the time dimension, which is more complex than image classification systems. Second, the audio sampling rate is usually very high (e.g., 16kHz, which means sampling 16,000 point per second), but images only have hundreds/thousands of pixels in total (e.g., the size of the images in the most popular datasets, i.e., MNIST and CIFAR-10, is 28×28 and 32×32 respectively). Therefore, it is harder to craft adversarial audios than images since adding slight noise to audios are less likely to impact the local features.

Recently, several mechanisms were proposed to generate adversarial audios [8, 10, 23]. They are all based on the gradient information, thereby sharing have very slight difference from each other. Even though these works against acoustic systems are seminal, they are limited in practice due to at least one of the following reasons: (i) they are designed only for a particular acoustic model under the white-box setting; (ii) they can only conduct untargeted attacks, with the goal of simply making the target systems misbehave; and (iii) they can only generate adversarial audios targeting phonetically similar phrases. In addition, none of them was comprehensively evaluated in end-to-end settings (more detailed analysis in Section 2).

In this paper, we present SirenAttack, a new class of adversarial attacks against deep neural network-based acoustic systems. Compared with prior work, SirenAttack departs in significant ways: *versatile* – SirenAttack is applicable to a range of end-to-end acoustic systems under both white-box and black-box settings; *targeted* – SirenAttack generates adversarial audio that trigger target systems to misbehave in a highly predictable manner (e.g., misclassifying the adversarial audio into a specific class); and evasive – SirenAttack generates adversarial audios by injecting a small amount of noise to legitimate audios while having negligible impact to human perception.

Our Contribution. To the best of our knowledge, this work represents the first systematical study on generating adversarial audios for various end-to-end acoustic systems. Our main contributions can be summarized as follows.

- We present SirenAttack, a new class of adversarial attacks against deep neural network-based acoustic systems under both white-box and black-box settings. For the white-box scenario, we combine a heuristic algorithm with a gradient-based method to conduct targeted/untargeted adversarial attacks. For the black-box scenario, we propose a new approach to conduct targeted/untargeted adversarial attacks by making use of a strong, iterative, and gradient-free algorithm.
- We evaluate SirenAttack on a range of state-of-the-art deep neural network models used in popular acoustic systems, including speech command recognition, speaker recognition and sound event classification systems. Experimental results show that SirenAttack is highly effective. For instance, it achieves 99.45% success rate on the IEMOCAP dataset against the ResNet18 model. Further, the generated adversarial audios can also be misinterpreted by multiple popular ASR platforms, including Google Cloud Speech Recognition, Microsoft Bing Voice Recognition, and IBM Speech-to-Text.
- We propose three potential defense strategies to mitigate the attacks of SirenAttack and conduct preliminary evaluation. Our results shed light on building more robust deep neural network-based acoustic systems, and lead to promising directions for further research.

2 RELATED WORK

2.1 Traditional Attacks on Acoustic Systems

In [7], Carlini *et al.* generated sounds that are unintelligible to humans while can be interpreted as commands by machine learning models. This attack targets at GMM-HMM systems rather than the advanced end-to-end neural networks used in most modern speech recognition systems as we focus on in this paper. In [55], Zhang *et al.* proposed DolphinAttack, which exploits the non-linearity of the microphones to create commands inaudible to humans while audible to speech assistants. From the defence perspective, such attack can be eliminated by an enhanced microphone that can suppress acoustic signals on the ultrasound carrier. In [53], Yuan *et al.* embedded voice commands into songs, which can be recognized by ASR systems over the air while being imperceptible to a human listener. However, this kind of attacks can be defended by audio turbulence and audio squeezing in practice.

2.2 Adversarial Attacks on Acoustic Systems

Inspired by adversarial attacks on images, adversarial audios have also drawn researchers' attention. In [8], Carlini *et al.* proposed a method that can produce an adversarial audio that could be transcribed as the desired text by DeepSpeech [19] under white-box settings. Nevertheless, their method would take more than one hour to generate an adversarial audio, and thus is very inefficient. In [10], Cisse *et al.* proposed the Houdini attack that is transferable to different unknown ASR models. However, it can only construct adversarial audios targeting phonetically similar phrases. In [23], Iter *et al.* generated adversarial audios by adding perturbations to the Mel-Frequency Cepstral Coefficients (MFCC) features and then rebuilt the speech from the perturbed MFCC features. Nevertheless, the noise introduced by the inverse-MFCC process makes their adversarial audios sound strange to human. In [16], Gong *et al.* demonstrated that a 2% distortion of speech can make a Deep Neural Networks (DNNs) based model fail to recognize the identity of the speaker. However, it is an untargeted attack that is difficult to pose a real threat.

2.3 Defense for Acoustic Systems

As traditional attacks on acoustic systems have been extensively studied, there are many defense methods to eliminate the effects of them. In [29], the authors proposed a virtual button that leverages Wi-Fi to detect human motions, and voice commands are only accepted when human motion is detected. In [14], the authors proposed VAuth, which collects the body-surface vibration of the user through a wearable device and verifies that the voice command is from the user. However, these methods are limited since voice commands are not necessarily accompanied with detectable motion, and the need for wearable devices (e.g., eyeglasses) may be inconvenient. Other defence schemes [7, 12, 40] mention the possibility of using Speaker Verification (SV) systems for defense. Nevertheless, this is not very useful since the SV system itself is vulnerable to previously recorded user speech [7]. As for adversarial attacks on acoustic systems, there are few defense schemes in published literature. Therefore, in this paper, we propose three potential defenses against such attacks. More in-depth dedicated defense research is expected in the future.

2.4 Remarks

In summary, the following aspects distinguish SirenAttack from existing adversarial attacks on acoustic systems. First, previous

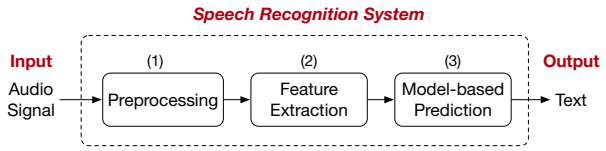


Figure 1: A typical end-to-end speech recognition system.

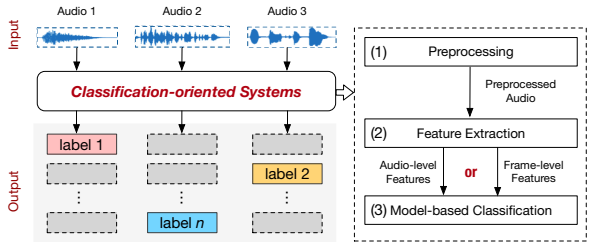


Figure 2: A typical classification-oriented system.

work usually focuses on one acoustic system and attacks only one or two models under white-box settings. In contrast, we systematically study adversarial audios against state-of-the-art acoustic models in three kinds of popular acoustic scenarios under both white-box settings and black-box settings. To our best knowledge, this is the first large-scale evaluation on the robustness of state-of-the-art acoustic models. Second, SirenAttack is computationally efficient and can generate an adversarial audio within minutes. Finally, our adversarial audios can also be misinterpreted by many popular ASR platforms, while previous studies seldom evaluate their attacks' performance on real ASR platforms. This implies that SirenAttack is more general and robust.

3 BACKGROUND

3.1 Recognition-oriented Acoustic Systems

Speech recognition is one of the most popular acoustic systems. A traditional speech recognition system is composed of an acoustic model, a dictionary, and a language model. The acoustic model and language model are trained separately. The disadvantage of this method is that it does not necessarily improve the overall recognition performance. Recently, end-to-end speech recognition becomes more and more popular, since it has great performance when given sufficient labeled training data. Therefore, we focus on attacking end-to-end speech recognition systems rather than the traditional ones in this paper.

An end-to-end speech recognition model can directly map the raw audio into the output words as shown in Fig. 1. It consists of the following three steps: (1) *Pre-processing*. This step eliminates the time periods whose signal energy falls below a particular threshold. One of the most popular technique used in this step is Voice Activity Detection (VAD), which usually consists of a noise reduction stage, a block-feature calculation stage and a classification stage. (2) *Feature Extraction*. This step splits the pre-processed audio into short frames and extracts features from each frame. The most commonly used feature extraction method in speech recognition systems is MFCC [39]. At a high-level, extracting MFCC features entails converting the audio to the frequency domain via a discrete Fourier transform, applying a Mel Filter Bank, and then performing an inverse Fourier transform to convert the signal back to the time domain. (3) *Model-based Prediction*. This step takes the extracted features as input, and matches them with an existing model to generate prediction results. Modern systems usually use Recurrent Neural Networks (RNNs) with a CTC loss function [18], which only requires one input sequence and one output sequence.

3.2 Classification-oriented Acoustic Systems

Generally, the intent of a classification-oriented acoustic system is to categorize the sample points in a clip of audio into one of the given classes. As shown in Fig. 2, a classification-oriented acoustic system consists of the following three steps: (1) *Pre-processing*. This step is the same as that in recognition-oriented systems. (2) *Feature Extraction*. This step can extract two kinds of features. One is the audio-level feature and the other is the frame-level feature. The audio-level features are extracted from the whole audio waveforms, while the frame-level features are extracted from the segmented waveform frames. A typical audio-level feature extraction method can be found in [27], and a typical frame-level feature extraction method can be found in [21]. Both of these methods employ Convolutional Neural Networks (CNNs) to extract features. (3) *Model-based Classification*. This step takes the extracted features as input, and matches them with an existing model built offline to generate classification results. The technique used in this step can vary widely. Nevertheless, modern systems usually use CNNs due to its outstanding performance in the computer vision domain.

4 ATTACK DESIGN

4.1 Problem Formulation

Given a pre-trained classification/recognition model $f : \mathcal{X} \rightarrow \mathcal{Y}$ from a feature space $\mathcal{X}$ to a set of prediction results $\mathcal{Y}$, an adversary aims to generate an adversarial audio $\boldsymbol{x_{adv}}$ from a legitimate audio $\boldsymbol{x} \in \mathcal{X}$ with its ground truth label $y \in \mathcal{Y}$, so that $\boldsymbol{x_{adv}} \approx \boldsymbol{x}$, i.e., it is difficult for human to distinguish $\boldsymbol{x_{adv}}$ from $\boldsymbol{x}$, while the classifier predicts $f(\boldsymbol{x_{adv}}) = t$ where t is the targeted phrase or class and $t \neq y$.

4.2 Threat Model

Under white-box settings, attackers are assumed to have the complete knowledge of all the details including model architecture and model parameters about the victim model and can interact with it while conducting the attack. This is a common threat model adopted in most prior work [8, 38] which assumes an adversary with the most power.

Under black-box settings, attackers are assumed to know nothing about the architecture, parameters or training data of the victim model. Therefore, the query function of the victim model can be characterized as an oracle $O(\boldsymbol{x})$ which returns the confidence value of the candidate classes. This assumption is practical, since many Machine-Leaning-as-a-Service (MLaaS) platforms usually do not release their detailed algorithms or training data but provide the confidence value of each candidate class.

4.3 Preparation

SirenAttack is based on the Particle Swarm Optimization (PSO) algorithm [13] and the fooling gradient method [47]. We begin by briefly introducing these techniques.

Particle Swarm Optimization (PSO). PSO is a heuristic and stochastic algorithm to find solutions for optimization problems by imitating the behavior of a swarm of birds [13]. It can search a very large space of candidate solutions while does not require the gradient information. At a high level, it solves a problem by iteratively making a population of candidate solutions (which we referred to as *particles*) move around in the search-space according to their fitness values. The fitness value of a particle is the evaluation result of the objective function on that particle's position in the solution space. In each iteration, each particle's movement is influenced by its local best position P_{best}, and meanwhile is guided toward the global best position G_{best} in the search-space. This iteration process is expected to move the swarm toward the best solution. Once a termination criterion is met, G_{best} should hold the solution for a global minimum.

Fooling Gradient Method. The fooling gradient method is a simple method referenced in adversarial image generation studies [47]. In this method, the gradient is computed with respect to the input data rather than the model parameters. Then the same gradient descent technique is applied to iteratively modify the input data. In a nutshell, the key differences between the standard setup for training a NN and the fooling gradient method are (1) the gradients are applied only to the input data, and (2) the loss is computed between the network's predictions and the target labels rather than the ground truth labels.

4.4 Design of the Attack

Logically, a classification-oriented task can be regarded as a one-frame instance of the recognition-oriented task. Hence, we introduce SirenAttack from the aspect of white-box and black-box settings instead of application scenarios.

4.4.1 White-box Attack. At a high level, the white-box attack contains two phases. The goal of the first phase is to find a coarse-grained noise $\boldsymbol{\delta'}$ that is close to the exact adversarial noise $\boldsymbol{\delta}$, while the goal of the second phase is to find the exact adversarial noise $\boldsymbol{\delta}$ by slightly revising $\boldsymbol{\delta'}$. Such procedure is designed under the consideration of effectiveness and efficiency. The detailed white-box attack is shown in Algorithm 1. The first phase contains the steps in line 2-13 and the second phase contains the steps in line 15-19.

First, we initialize the *epoch* to zero and generate *n_particle* randomized sequences from a uniform distribution (line 1). The randomized sequences are collectively referred to as *seeds*. Then we run the PSO subroutine (line 3) with the target output t and *seeds*. If any particle $\boldsymbol{p_i}$ produces the target output t when being added to the original audio $\boldsymbol{x}$, then the attack succeeds (line 4-5), and the particle $\boldsymbol{p_i}$ is the expected noise $\boldsymbol{\delta}$. Otherwise, we will preserve the best particle that has the minimum fitness value in the current PSO run as one of the *seeds* in the next PSO run (line 7-8). From the *n_step* epoch, we calculate the standard deviation *std* of the global best fitness value from the last *n_step* PSO runs (line 10-12). Once the *std* is below the threshold ϵ, it is not efficient for continuously running the PSO subroutine to find the exact noise $\boldsymbol{\delta}$ since the global best fitness value now changes slowly. Therefore, we only obtain a coarse-grained noise $\boldsymbol{\delta'}$ after the first phase.

Algorithm 1 Generation of targeted adversarial audios under white-box settings

Input: Original audio $\boldsymbol{x}$, target output t, $n\_particles$, $epoch_{max}$, n_step, ϵ
Output: An targeted adversarial audio $\boldsymbol{x_{adv}}$

1: Initialize $epoch = 0$ and *seeds* and set CTC loss as the objective function;
2: **while** $std \leq \epsilon$ **do**
3: Run PSO subroutine with t and *seeds*;
4: **if** any particle produce target output t during PSO **then**
5: Solution is found. Exit.
6: **else**
7: Clear *seeds*;
8: *seeds* $\supseteq$ *best particle* that produce the minimum CTC loss from the current PSO run;
9: **end if**
10: **if** epoch $\geq$ n_step **then**
11: Calculate *std*;
12: **end if**
13: **end while**
14: Obtain coarse-grained noise $\boldsymbol{\delta'}$ from current *seeds*;
15: **while** epoch reaches $epoch_{max}$ or $O(\boldsymbol{x} + \boldsymbol{\delta'}) = t$ **do**
16: Calculate loss according to Eq. (1),
17: Update $\boldsymbol{\delta'}$ according to the gradient information;
18: $epoch = epoch + 1$;
19: **end while**
20: Get adversarial audio $\boldsymbol{x_{adv}}$ with target label t.

We would further emphasize two key aspects of our algorithm: (1) We modify the PSO algorithm to globally keep track of the current saved best particle throughout all PSO iterations instead of using the standard PSO algorithm. (2) During each iteration, PSO aims to minimize an objective function defined as $g(\boldsymbol{x} + \boldsymbol{p_i})$. Note that RNN-like models' output is a matrix containing the probability of the characters at each frame. Therefore, we choose the CTC loss function [18] as $g(\cdot)$ in this attack, i.e., $g(\boldsymbol{x}+\boldsymbol{p_i}) = CTC{-}loss(\boldsymbol{x}+\boldsymbol{p_i})$. The value of $g(\boldsymbol{x} + \boldsymbol{p_i})$ at each particle is then used to move them in a new direction.

In the second phase, we calculate the loss function and use the fooling gradient method to adjust $\boldsymbol{\delta'}$ until $O(\boldsymbol{x} + \boldsymbol{\delta'}) = t$ or *epoch* reaches $epoch_{max}$. The loss function is defined as follows:

$$minimize\ \mathcal{L}(\boldsymbol{x} + \boldsymbol{\delta'}, t) + \lambda|\delta'|^2 \tag{1}$$

where $\mathcal{L}$ is the CTC Loss and $\lambda|\delta|^2$ is the regularization term. This loss function can be revised to $-\mathcal{L}(\boldsymbol{x} + \boldsymbol{\delta'})$ to conduct untarget attacks.

4.4.2 Black-box Attack. The detailed black-box attack is shown in Algorithm 2. To fool a machine learning model, we feed it with a legitimate audio $\boldsymbol{x}$ and the target output t. We also pre-define the number of particles as $n\_particles$ and the maximum epoch number as $epoch_{max}$. The basic procedure (line 2-11) of the black-box attack is similar to the first phase of the white-box attack except for the following two things: (i) the objective function is different due to

Algorithm 2 Generation of targeted adversarial audios under black-box settings

Input: Original audio $\boldsymbol{x}$, target output t, $n\_particles$ and $epoch_{max}$
Output: An targeted adversarial audio $\boldsymbol{x_{adv}}$

1: Initialize $epoch = 0$ and $seeds$ and set Eq. (2) as the objective function;
2: **while** epoch reaches $epoch_{max}$ **do**
3: Run PSO subroutine with t and $seeds$;
4: **if** any particle produce target output t during PSO **then**
5: Solution is found. Exit.
6: **else**
7: Clear $seeds$;
8: $seeds \supseteq best\ particle$ that produce the minimum value of Eq. (2) from the current PSO run;
9: **end if**
10: $epoch = epoch + 1$;
11: **end while**
12: Get adversarial audio $\boldsymbol{x_{adv}}$ with target label t.

lacking of loss information, and (ii) the termination condition is different since we should obtain the exact noise $\boldsymbol{\delta}$ in this process. We experimented with several definitions of $g(\cdot)$ and found the following to be the most effective:

$$g(\boldsymbol{x} + \boldsymbol{p_i}) = \max(\max_{j \neq t}(O(\boldsymbol{x} + \boldsymbol{p_i})_j) - O(\boldsymbol{x} + \boldsymbol{p_i})_t, \kappa) \quad (2)$$

where $O(\boldsymbol{x} + \boldsymbol{p_i})_j$ is the confidence value of label j for input $\boldsymbol{x} + \boldsymbol{p_i}$. The function can move the particles to the position that maximizes the probability of the target label t. In addition, we can control the confidence of misprediction with the parameter κ, and a smaller κ means that the found adversarial audio will be predicted as t with higher confidence. We set $\kappa = 0$ for SirenAttack but we note here that a side benefit of this formulation is that it allows one to control the desired confidence. The algorithm iterates on this process (line 2-11) till the attack succeeds or it reaches $epoch_{max}$. If succeed, we would obtain an adversarial audio $\boldsymbol{x_{adv}}$ that can be predicted as t by the victim model. Furthermore, this function can be used to conduct untarget attacks with trivial modifications.

Compared with the white-box attack, the black-box attack is less efficient and introduces more noise in the generated adversarial audios. This is because the black-box attack lacks of loss information and gradient information. Therefore, some performance decrease of the black-box attack is reasonable.

5 WHITE-BOX ATTACK EVALUATION

5.1 Datasets

In this experiment, we take the audios from the Common Voice dataset [37] and the VCTK Corpus [49] as the original samples. The Common Voice dataset is a corpus of speech data read by users based upon the text from a number of public domain sources like user submitted blog posts, old books, movies, and other public speech corpora. It has 500 hours of samples, comprising 400,000 recordings made by 20,000 people. The VCTK Corpus includes speech data from 109 native speakers of English with various accents.

5.2 Target Model

In this set of evaluation, we examine the security of DeepSpeech [19], a state-of-the-art RNN-based ASR model proposed by Baidu, which is trained on a dataset consisting of 100,000 hours of noisy speech data and can achieve around 81% accuracy in noisy environments like restaurants. Although there are several other speech recognition models proposed in [5], we choose DeepSpeech as our target model due to the following reasons: (i) it is hard to reproduce the results in those papers due to lacking of sufficient implementation details, and (ii) the input data format of some available Speech-To-Text engines (e.g., WaveNet) are MFCC features instead of raw audio waveforms, and therefore we need to rebuild the adversarial audio from the inverse MFCC process which will greatly reduce the quality of the audios. On the other hand, the DeepSpeech model implemented by the Mozilla group, which has more than 6,000 stars in the Github repository, is a proper choice for evaluating SirenAttack. Its input data format is raw audio waveform. Though it is a research project now, the developers of DeepSpeech claimed that Baidu would integrate DeepSpeech into automatic car, CoolBox and wearable devices in the future. Thus, it is more practical compared with other models.

5.3 Evaluation Metric

There are two objective audio quality assessment techniques [11], i.e., Signal-to-Noise Ratio (SNR) and Objective Difference Grade (ODG). As previous works usually use SNR to evaluate the quality of generated adversarial audios [25, 53], we also use SNR to evaluate the audio quality for consistency and comparison with previous works.

SNR is a metric extensively used to quantify the level of signal power to noise power, which is calculated as follows:

$$SNR(dB) = 10\ log_{10}(\frac{P_{\boldsymbol{x}}}{P_{\boldsymbol{\delta}}}) \quad (3)$$

where $\boldsymbol{x}$ is the original audio waveform, $\boldsymbol{\delta}$ is the added noise, and $P_{\boldsymbol{x}}$ and $P_{\boldsymbol{\delta}}$ are the power of the original signal and the noise signal, respectively. A large SNR value indicates a small noise scale. For our purpose, we use it to measure the distortion of the adversarial audio relative to the original audio.

According to the International Federation of the Phonographic Industry (IFPI), the imperceptible noise requires at least 20 dB SNR value between the noise signal and the original signal. However, this is unnecessary for SirenAttack. SirenAttack tolerates the noise to some extent as long as it does not impact human perception. Therefore, the SNR of the generated adversarial audio is acceptable even though they do not reach the 20 dB threshould. To further demonstrate this, a user study was conducted in Section 7.3.

5.4 Implementation

We conducted the experiments on a server with two Intel Xeon E5-2640 v4 CPUs running at 2.40GHz, 64 GB memory, 4TB HDD and a GeForce GTX 1080 Ti GPU card. We set $epoch_{max} = 300$, $n\_step = 5$, $\epsilon = 2$ and the iteration limit of PSO to 30 in all experiments. For the PSO subroutine, we set $n\_particles = 25$, $c_1 = c_2 = 1.4961$. Specially, r_1 and r_2 are random values uniformly sampled from $[0, 1]$ to avoid consistency. In addition, we adopted the adaptive method

Table 1: Results of the white-box attack on DeepSpeech.

Dataset	Original Length	Target Length	Performance (without VAD)			Performance with VAD		
			Success Rate	SNR(dB)	Time(s)	Success Rate	SNR(dB)	Time(s)
Common Voice	11.87 words	4.74 words	100%	18.72	1560.14	100%	20.01	1201.66
VCTK Corpus	12.93 words	4.74 words	100%	16.54	1897.49	100%	18.34	1613.87

Table 2: Adversarial Audios against DeepSpeech.

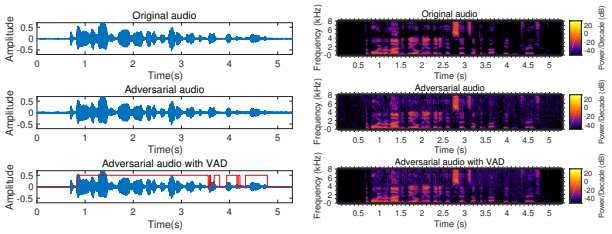
Number	Original Audio (Recognized result of DeepSpeech)	Adversarial Audio (Recognized result of DeepSpeech)	SNR(dB)	Time(s)
1	Follow the instructions here	Read last sms from boss	18.07	685.14
2	One can imagine these two covered with sand running up the little street in the bright sunlight	Ask capital one to make a credit card payment	17.48	2205.63
3	They're calling to us not to give up and to keep on fighting	Call the police for help quickly	15.47	1425.86
4	What do you think of that	Turn on flashlight	16.93	791.43
5	Six spoons of fresh snow peas, five thick slabs of blue cheese, and maybe a snack for her brother Bob	Send a message to Derek: Hang on, I'm going to get more coffee	23.98	2274.51
6	I told him we could teach her to ignore people who waste her time	What is my schedule for tomorrow	14.99	1729.38
7	The industry is not well organized	Show me my last messages	17.11	987.17
8	Nature knows me as the wisest being in creation the sun said	Please restart the phone	19.07	2079.11
9	The boy reminded the old man that he had said something about hidden treasure	Clear SMS history from my phone	21.45	1879.29
10	It was dropping off in flakes and raining down on the sand	Remove all photos in my phone	20.71	1177.88

on inertia weight w, i.e., we initially set $w = 0.9$, which makes the PSO has strong global optimization ability; with the increasing of the iteration, w is decremented, so that the PSO has strong local optimization ability; when the iteration ends, $w = 0.1$. For the gradient-based phase, we did some search over hyper-parameters such as learning rate to find a trade-off between effectiveness and efficiency. In particular, we set the learning rate as 1.

5.5 Results and Analysis

Effectiveness and Efficiency. The main experimental results are shown in Table 1, which summarizes the performance of SirenAttack on the two datasets. We randomly choosed 200 instances from the Common Voice dataset and the VCTK Corpus as the original audios. The target commands were randomly chosen from a list of all the Google Now voice commands[1]. We evaluate the average time of generating an adversarial audio, since it is important for an adversary to mount the attacks in realistic settings. From Table 1, we can see that SirenAttack is very effective and efficient. It takes less than 1,600 seconds and 1,900 seconds on average to generate a successful adversarial audio (100% success rate) on the Common Voice dataset and the VCTK Corpus, respectively. Therefore, attackers may create plenty of adversarial audios in a short time. Furthermore, the adversarial audios have small distortion as shown in Table 1. For instance, the average SNR of the generated adversarial audios on the Common Voice dataset is 18.72 dB, which means less than 2% distortion compared with the original audios.

To visualize the distortion, we plot the waveform and spectrogram of an example original audio and the corresponding adversarial audio in Fig. 3. The spectrogram of the original audio and the corresponding adversarial audio in Fig. 3(b) are obtained from



(a) Waveform (b) Spectrogram

Figure 3: Comparison of the waveform and spectrogram among the original audio, adversarial audio without VAD and adversarial audio with VAD. The original transcription is "*the boy reminded the old man that he had said something about hidden treasure*" while the adversarial transcription is "*clear SMS history from my phone*".

Short-Time Fourier Transform (STFT) of the waveform, where the horizontal axis represents time, the vertical axis represents frequency, and the color indicates the strength of energy. In fact, after the sound enters the human ear, the cochlea will also process the sound similar to STFT. Therefore, the sounds that people can distinguish often show specific patterns on the spectrogram. From Fig. 3(b), we can see that although the noise covers a broad spectrum, its energy is much lower than the vocal part. Hence, the noise in the adversarial audios is ignorable to humans and such attack is very stealthy.

Examples. Table 2 shows five examples in which the prediction results of adversarial audios are completely changed. For instance, the case of converting "*follow the instructions here*" to "*read last sms from boss*" can be used to steal users' privacy information

[1]https://www.greenbot.com/article/2359684/android/a-list-of-all-the-ok-google-voice-commands.html

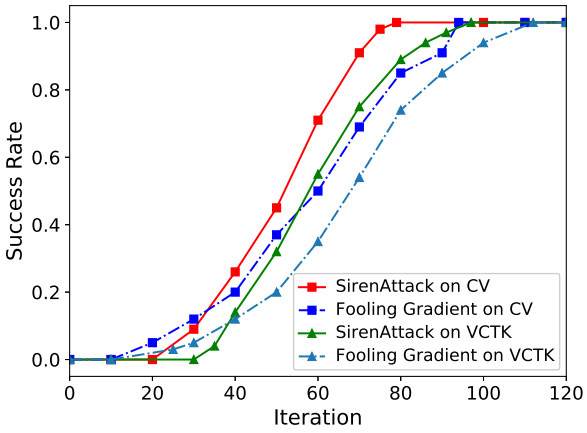
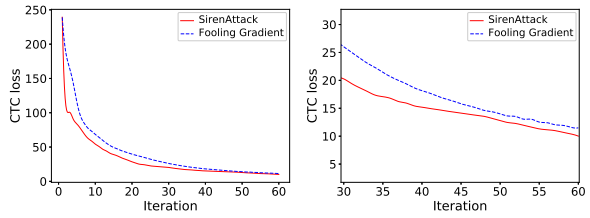


Figure 4: The correlation between the iteration and the success rate.



(a) Iteration 0-60 (b) Iteration 30-60

Figure 5: CTC loss of the fooling gradient method and SirenAttack when converting the original audio to an adversarial audio.

through their speech assistants. Therefore, this kind of attack can be leveraged by attackers to conduct malicious attacks on speech recognition systems. In addition, we observe a positive correlation between the length of the utterance and the time required to generate adversarial audios, which means generating longer adversarial audios may suffer from scaling issues to some extent.

Performance Comparison. We compare SirenAttack with the fooling gradient method [23] by showing the correlation between the iteration and the success rate in Fig. 4. Observe that SirenAttack on the Common Voice dataset reaches 100% success rate at iteration 79 while the fooling gradient method reaches 100% success rate at iteration 94. This suggests that SirenAttack is more efficient. Though the fooling gradient method finds the first adversarial audio faster than SirenAttack, its success rate increases slower. Specifically, taking converting "*the boy reminded the old man that he had said something about hidden treasure*" to "*clear SMS history from my phone*" as an example, we show the CTC loss of the fooling gradient method (blue dotted line) and SirenAttack (red solid line) in Fig. 5. We can see that the CTC loss is decreasing faster in SirenAttack than that in the fooling gradient method. This implies that SirenAttack chooses a direction that can find an adversarial audio faster than the fooling gradient method.

Table 3: Synthesized Commands.

Number	Commands
1	Okay Google
2	Restart the phone
3	Flashlight on
4	Read email
5	Clear notification
6	Airplane mode on
7	Turn on wireless hot spot
8	Read last sms from boss
9	Open the front door
10	Turn off the light
11	Ask capital one to make a credit card payment

Improved Attack. To further improve the performance of SirenAttack, we use the Voice Activity Detection (VAD) Toolkit described in [26] to find the active part of audios and only add noise to the active region. The results are also shown in Table 1, from which we can see that VAD does increase the SNR of the generated adversarial audios and improve the efficiency of the generation process. For instance, the adversarial audios on the Common Voice dataset have an average SNR of 20.01 dB (18.72 dB without VAD) and an average generation time of 1201.66 seconds (1560.14 seconds without VAD) when applying VAD. Further, we compare the waveform and spectrogram of an example adversarial audio with and without VAD in Fig. 3, from which we can see that the inactive voice parts occupy nearly one third of the original audio. Therefore, adding noise to the active parts of audios does increase the SNR of adversarial audios, i.e., generate better adversarial audios.

6 BLACK-BOX ATTACK EVALUATION

6.1 Target Applications

We conducted black-box attacks under three different scenes, including speech command recognition, speaker recognition and sound event classification.

6.1.1 Speech Command Recognition. In this scenario, we generated adversarial commands that can be recognized as target phrases for speech command recognition systems. For instance, we may start with an audio saying "*yes*", which can be correctly recognized by the system. After applying the attack, the system will recognize the input as "*no*" while a human still clearly hears "*yes*".

First, we briefly introduce the datasets and the victim models used in this set of evaluation. The used datasets are briefly described as follows: (i) *Speech Commands Dataset* [50]. This dataset consists of 65,000 audio files of 30 short words. Each file is a one-second audio of a single word like: "*yes*", "*no*", digits, and directions. (ii) *Synthesized Commands.* As shown in Table 3, we synthesized 33,000 audio files of 11 long speech commands with 3,000 clips per label at different speeds and tones through several famous Text-to-Speech engine including Baidu, Google, Bing and IBM. The 11 commands are commonly used in daily life, such as "*okay Google*" and "*turn on the airplane mode*". They were chosen to represent a variety of potential attacks against personal speech assistants.

The target victim models are as following: (i) *The CNN described in* [43]. This model, which is pre-trained by the TensorFlow team,

is an efficient and light-weight keyword spotting model based on a CNN and achieves 96.10% classification accuracy on the Speech Commands Dataset. (ii) *Six State-of-the-art Speech Command Recognition Models.* We use *VGG19* [46], *DenseNet* [22], *ResNet18* [20], *ResNeXt* [52], *WideResNet18* [54] and *DPN-92* [9] as the target victim models. These models are well known for their good classification performance on image data. In addition, they have good performance in the TensorFlow Speech Recognition Challenge[2]. Therefore, we modify them to adapt to the spectrogram input.

6.1.2 Speaker Recognition. Speaker recognition is the identification of a person from the characteristics of voices [42], which can be used to authenticate the identity of a speaker as part of a security process. We simplify the speaker recognition task in our experiment by limiting it to a ten-class classification problem, which is reasonable and common [16]. It makes the model perform better. In addition, from the evaluation perspective, it is more meaningful to attack a model that has good performance. Then, we target the same kinds of models used in the speech command recognition task. Further, we conduct the adversarial attack using the *IEMOCAP* dataset [6], which consists of ten speakers (five female, five male) and is a commonly used dataset in speech paralinguistic research [16].

6.1.3 Sound Event Classification. The goal of sound event classification is to give a predefined label to the sound event (e.g., "*dogbark*", "*siren*") within an audio signal. It has numerous applications, including audio surveillance systems [1, 35], hearing aids [2], smart room monitoring [48], and pornographic content detection [33]. In this scenario, our goal is to fool the sound event classification systems into producing an incorrect target prediction. For instance, we may start with an audio correctly recognized as "*gunshot*", a dangerous event that may cause attention from monitors. However, the system will classify the corresponding adversarial audio as a normal event (e.g., "*dogbark*"), while human beings can still hear "*gunshot*".

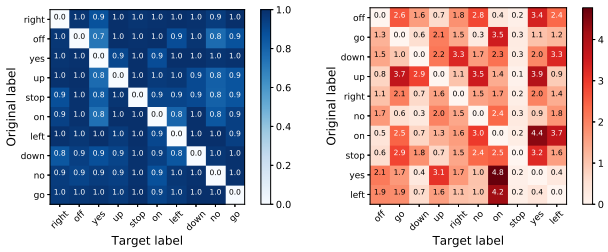
We employ three large-scale sound event datasets to evaluate SirenAttack: (i) *AudioSet* [15]. It has 632 sound event classes covering a wide range of everyday environmental sounds. (ii) *ESC-50* [41]. It consists of 2,000 5-second-long environmental audio recordings organized into 50 classes with 40 audios per class. (iii) *UrbanSound8K* [44]. It contains 8,732 labeled sound excerpts (no more than four seconds for each) of urban sounds from ten classes.

As for the victim models, we use the YouTube-8M starter code[3] to train three victim models including the *Logistic Model (LM)*, the *Mixture of Experts (MoE) model* and the *Frame-Level Logistic Model (FLLM)* , according to the instruction of AudioSet[4].

6.2 Implementation

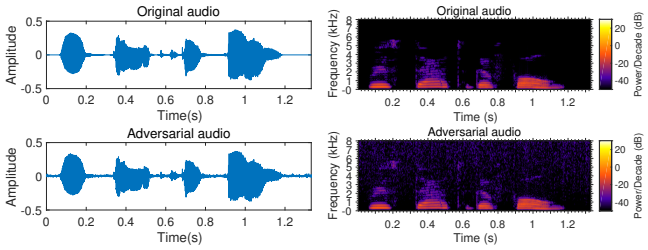
The implementation details of our black-box attack are almost the same as that in the white-box attack. One difference is that we need to train some targeted models due to the lack of pre-trained models. Therefore, except for the CNN model, all models were trained in a hold-out test strategy, i.e., 80%, 10%, 10% of the data was used for training, validation and test, respectively. Hyper-parameters were tuned only on the validation set.

[2]https://www.kaggle.com/c/tensorflow-speech-recognition-challenge
[3]https://github.com/google/youtube-8m
[4]https://research.google.com/audioset



(a) Success Rate (b) Time (min)

Figure 6: Performance (success rate and time) of SirenAttack for every $\{source, target\}$ pair on the Speech Commands Dataset against the CNN model.



(a) Waveform (b) Spectrogram

Figure 7: Comparison of the waveform and spectrogram between an original audio (upper graphs) and the corresponding adversarial audio (lower graphs) of the Synthesized Commands dataset with $\delta = 100$. The original transcription is "*restart the phone*" while the adversarial transcription is "*open the front door*".

6.3 Evaluation Results

Attacks on Speech Command Recognition Systems. We selected 2,000 audio clips from the Speech Commands Dataset with 200 clips per label and generated nine targeted adversarial audios for each audio file. For instance, for audio "*no*", we generated adversarial audios that are intended to be identified as "*yes*", "*left*" and so on. If SirenAttack fails to find an adversarial audio within the $epoch_{max}$ iterations, we declare this as failure and vice versa. Again, we evaluated the required time and SNR. Notice that the CNN model was pre-trained on the Speech Commands Dataset. Hence, we did not evaluate it on the Synthesized Commands dataset.

The attack results are shown in Table 4 with $\delta = 800$ and $epoch_{max} = 300$, including the models' accuracy on the original dataset, the success rate of SirenAttack, the SNR of the generated adversarial audios and the average time to generate an adversarial audio. Figs. 6(a) and 6(b) show the pair-to-pair success rate and the average time to generate an adversarial audio of SirenAttack on the Speech Commands Dataset (resp., the Synthesized Commands

Table 4: Performance of the Black-box Attack on Speech Command Recognition and Speaker Recognition.

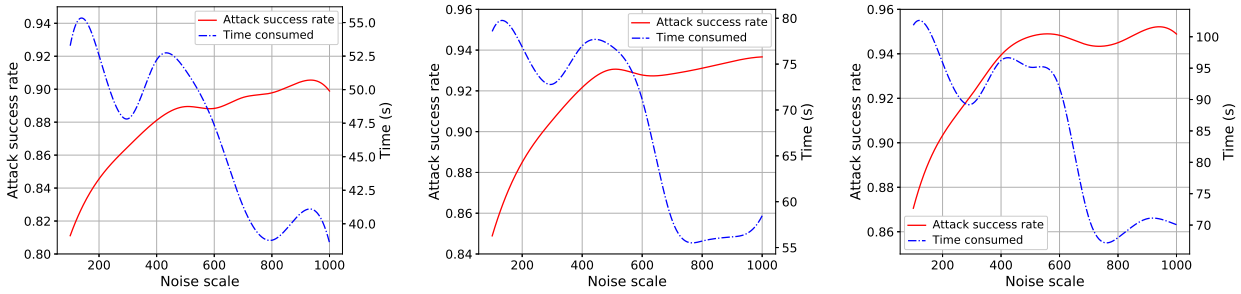
Model/Dataset	Speech Commands				Synthesized Commands				IEMOCAP			
	Accuracy	Success Rate	SNR(dB)	Time(s)	Accuracy	Success Rate	SNR(dB)	Time(s)	Accuracy	Success Rate	SNR(dB)	Time(s)
CNN	96.10%	95.25%	22.36	100.69	-	-	-	-	-	-	-	-
VGG19	91.39%	88.10%	18.22	332.26	93.12%	93.75%	17.04	420.89	85.01%	91.65%	16.33	376.40
DenseNet	94.93%	86.90%	15.34	458.13	93.34%	89.25%	15.13	602.81	84.28%	94.65%	15.28	572.23
ResNet18	92.06%	87.35%	15.87	340.31	93.90%	90.15%	17.28	381.27	86.37%	99.45%	23.12	386.15
ResNeXt	94.28%	90.05%	17.03	317.92	94.80%	92.60%	18.49	458.44	87.66%	95.60%	20.87	420.37
WideResNet18	90.80%	89.25%	17.57	368.29	92.68%	91.05%	17.23	403.76	92.41%	93.95%	17.06	393.56
DPN92	95.20%	83.60%	14.04	462.58	96.81%	90.55%	14.77	587.80	86.93%	92.80%	15.51	564.98

Table 5: Performance of the Black-box Attack on Event Sound Classification.

Feature Type	Target Model	ESC-50				Urbansound8K				AudioSet			
		Accuracy	Success Rate	SNR(dB)	Time(s)	Accuracy	Success Rate	SNR(dB)	Time(s)	Accuracy	Success Rate	SNR(dB)	Time(s)
Audio-level	LM	99.95%	85.33%	18.28	269.34	88.04%	84.00%	14.51	598.62	85.49%	92.67%	18.04	283.57
	MoE	99.82%	84.00%	17.91	283.41	92.36%	82.00%	12.80	609.63	88.32%	91.33%	13.58	287.19
Frame-level	FLLM	87.71%	80.67%	18.11	328.75	84.28%	80.00%	19.73	487.04	81.02%	90.67%	21.62	403.41

Table 6: Examples of the Black-box Attack on Event Sound Classification.

Feature Type	Target Model	ESC-50				Urbansound8K				AudioSet			
		Original	Target	SNR(dB)	Time(s)	Original	Target	SNR(dB)	Time(s)	Original	Target	SNR(dB)	Time(s)
Audio-level	LM	Breaking	Crickets	29.51	166.02	Gunshot	Dog bark	17.55	223.76	Breaking	Clock alarm	22.19	214.93
	MoE	Breaking	Crickets	25.23	167.11	Gunshot	Dog bark	17.57	224.75	Breaking	Clock alarm	11.51	219.16
	LM	Siren	Wind	12.09	222.51	Siren	Street music	16.26	735.41	Gunshot	Children playing	12.88	224.84
	MoE	Siren	Wind	12.21	219.08	Siren	Street music	14.03	745.96	Gunshot	Children playing	11.34	229.21
Frame-level	FLLM	Breaking	Crickets	15.85	289.61	Gunshot	Dog bark	20.05	306.03	Breaking	Frog	28.60	340.73
		Siren	Crickets	15.81	308.62	Siren	Street music	20.99	385.69	Gunshot	Dog bark	17.10	309.11



(a) $epoch_{max}$ = 100 (b) $epoch_{max}$ = 200 (c) $epoch_{max}$ = 300

Figure 8: The success rate and required time with different noise scale for Speech Commands Dataset.

dataset). Fig. 7 shows the waveform and spectrogram of an example original audio and the corresponding adversarial audio. We can observe the following from Table 4 and Fig. 6.

- From Table 4 and Fig. 6(a), we can see that SirenAttack is effective when against all the target models, even when the models have high performance on the legitimate datasets. For instance, SirenAttack has 95.25% success rate on the Speech Commands Dataset when against the CNN model and has 93.75% success rate on the Synthesized Commands dataset when against the VGG19 model. In addition, we notice that certain transformations seem to be easier than others. For instance, the conversion from "*yes*" to "*stop*" can be done in 10 iterations, while the conversion from "*stop*" to "*yes*" takes 160 iterations. We conjecture that this might result from the victim model's different prediction robustness among different categories. Another interesting observation is that some intermediate adversarial audios appear in the attacking process. For example, when we convert "*restart the phone*" to "*flashlight on*", the transcription result first changes to "*clear notification*" and then changes to "*flashlight on*".
- From Table 4, the average generation time of an adversarial audio is very short. For instance, the average generation time of an adversarial audio of the Speech Commands Dataset when against

the CNN model is 100.69 seconds. In addition, from Fig. 6(b) we can see that all of the adversarial audios can be generated in less than 5 minutes, and some $\{source, target\}$ pairs like $\{go, stop\}$ can be done within one minute. Therefore, SirenAttack is very efficient in practice.

- From Table 4, we can also see that the noise is slight, e.g., the SNR of the adversarial audios ranges from 14 dB to 22 dB on both datasets when against the target models. This implies that the noise in the adversarial audios is less than 3%. In addition, we show the waveform and spectrogram of an example audio and its corresponding adversarial audio in Fig. 7, from which we can also see that the noise is almost ignorable. This demonstrates the stealthy of SirenAttack.

Attacks on Speaker Recognition Systems. In this evaluation, we used the 1,000 audio clips from the IEMOCAP dataset with 100 clips per speaker and generated nine targeted adversarial audios for each audio file. The attack results are shown in Table 4 with $\delta = 800$ and $epoch_{max} = 300$. **??** further shows the pair-to-pair success rate of SirenAttack and the average time to generate an adversarial audio. Similar to the attack on the speech command recognition systems, SirenAttack is also very effective against all the target models as shown in Table 4 and **??**. For instance, SirenAttack has 99.45% success rate when against the ResNet18 model. In addition, SirenAttack is also efficient in this task, e.g., the average generation time of an adversarial audio of the IEMOCAP dataset against the VGG19 model is 376.40 seconds.

Attacks on Sound Event Classification Systems. In this evaluation, we trained the LM model and the MoE model on the audio-level features, which were extracted by the method in [27]. In addition, we trained the FLLM model on the frame-level features, which were extracted by the VGGish model in [21]. For ESC-50 and UrbanSound8K datasets, the victim models were trained on their own datasets respectively; for AudioSet, we took the pre-trained models on UrbanSound8K as the victim models. To demonstrate that SirenAttack can convert the threatening events to normal events, we randomly picked 150 $\{source, target\}$ pairs matching the $\{threatening\ event, normal\ event\}$ pattern from each of the three datasets to evaluate SirenAttack.

The results are shown in Table 5, from which we can see that SirenAttack is also effective and efficient against the target models. For instance, SirenAttack has 92.67% success rate on the AudioSet dataset when against the LM model with the average generation time of 283.57 seconds. Table 6 demonstrates some examples of SirenAttack to convert threatening events to normal events, e.g., SirenAttack can convert the threatening event "*gunshot*" to the normal event "*dogbark*", which can be used as an attack on acoustic surveillance systems.

7 FURTHER ANALYSIS

7.1 Perturbation Analysis

Now, we evaluate the impact of the noise scale δ and $epoch_{max}$ on the effectiveness and efficiency of generating adversarial audios. Specifically, we generated adversarial audios on the Speech Commands Dataset with different bound values of noises as well as $epoch_{max} = 100, 200, 300$. The targeted model is CNN. The success rate and required time are shown in Fig. 8, from which we can see that the trend of success rate is generally consistant with the noise scale. For instance, when $epoch_{max} = 100$, SirenAttack has 82% success rate with $\delta = 100$ while having 90% success rate with $\delta = 1000$. This implies that attackers can use larger noise scale to improve the success rate of their attacks. On the other hand, a larger δ also implies lower utility, i.e., human may notice the changes of the audio. From Fig. 8, we can also see that when $\delta = 800$, all the three time curves reach the minimum value. In addition, when $epoch_{max} = 300$, the overall success rate is higher than that of $epoch_{max} = 100, 200$. These findings help us derive better parameter settings. Hence, we use $\delta = 800, epoch_{max} = 300$ in our evaluation.

Table 7: Transferability evaluation results.

	Sphinx	Google	Bing	Houndify	Wit.ai	IBM
Success Rate	39.60%	10.00%	14.00%	12.80%	21.20%	20.40%

Table 8: Example results of transferability evaluation.

Number	Original Text	Advesarial Text	ASR Platforms	Results
1	stop	no	Sphinx	no
2	off	on	IBM	on
3	down	no	Wit.ai, Bing	no
4	go	no	Wit.ai	no
5	go	yes	Sphinx	yes
6	left	yes	Wit.ai, IBM	yeah
7	on	right	Wit.ai	alright
8	right	on	Google, Bing	play
9	right	down	Google, Bing	play
10	off	no	Bing	call
11	on	stop	Bing	call
12	on	up	Wit.ai	okay
13	stop	off	Wit.ai	the
14	down	up	Bing	phone
15	stop	go	Wit.ai	tell

7.2 Transferability Evaluation

Previous studies have shown that adversarial images generated for one model can be misclassified by other models, even when they have different architectures [17], i.e., adversarial images exhibit transferability, which can be used to conduct black-box attacks. Therefore, we are interested in (i) whether the transferability also exists in adversarial audios, and (ii) whether this property can be used to conduct black-box attacks. Specifically, we used 500 adversarial audios that are generated from the Speech Commands Dataset with the target model VGG19 to conduct proof-of-concept attack on several famous ASR platforms, including Sphinx, Google Cloud Speech Recognition, Microsoft Bing Voice Recognition, Houndify, Wit.ai and IBM Speech-to-Text. Note that we do not directly conduct black-box attacks on these ASR platforms since they are all recognition-oriented models which do not give any information except for the final transcription. In this scene, it is very difficult, if possible, to directly conduct black-box attacks on these models while guaranteeing the added noise is human-imperceptible.

The evaluation results are shown in Table 7, from which we can see that the adversarial audios generated by SirenAttack can also be misinterpreted as the target text by the target ASR platforms

to some extent. For instance, SirenAttack achieves 39.60% success rate on the Sphinx platform. This implies that the adversarial audios generated by SirenAttack can be used to mount targeted black-box attacks to other ASR platforms. Line 1-7 in Table 8 show some examples that are successfully transferred against other ASR platforms. In addition, line 8-15 in Table 8 show some additional misclassification results, which imply that the adversarial audios generated by SirenAttack may pose threats to people's privacy when being concatenated with other words, such as "*call 911*", "*okay Google*", "*restart the phone*", and "*tell me the phone number of Jack*".

7.3 Human Perceptual Study

To quantify the perceptual realism of the adversarial audios generated by SirenAttack, we also perform a user study with human participants on Amazon Mechanical Turk (MTurk). Before the study, we consulted with the IRB office and this study was approved and we did not collect any other information of participants except for necessary result data.

In the study, we recruited 200 native English speakers whose age ranges from 18 to 40 to participate in our survey. Each participant was asked to listen to 20 legitimate audios and 20 adversarial audios generated from the Speech Command dataset with CNN as the target model in a quiet environment. During each trial, participants are given unlimited time to replay audios and make their decisions. For each audio, a series of questions need to be answered, i.e., (1) what they heard from this audio (choose one option from the given ten options, i.e., *stop, go, yes, no, left, right, off, on, up, and down*); (2) whether they heard anything abnormal than a regular command (the four options are *no, not sure, a little noisy, and noisy*); (3) if choosing *a little noisy* or *noisy* option in (2), where they believe the noise comes from (the three options are *the device (speaker, radio, etc.), the sample itself, other*).

After examining the results, we find that 93.50% legitimate audios can be recognized correctly while 92.00% adversarial audios can be recognized as their original labels. None of the adversarial audios is classified as its adversarial label. This indicates that the generated adversarial audios have little impact on human perception. What's more, 38.5% of participants think the adversarial audios are a little noisy and only 4.5% participants think the noise are from the samples themselves. Furthermore, 10.5% of participants think the adversarial audios are noisy and only 2.5% participants think the noise are from the samples themselves. This implies that SirenAttack is stealty.

8 POTENTIAL DEFENSES

As there are few defense methods for adversarial audio attacks to the best of our knowledge, we conduct a preliminary exploration of potential defense schemes. By default, all the adversarial audios are generated using our black-box attack and we use the same implementation and evaluation settings as that in Section 6.

Adversarial Training. Adversarial training means training a new model with both legitimate and adversarial examples. We show the performance of this scheme along with detailed settings in Table 9, where the accuracy means the prediction accuracy of the new models on the legitimate audios. From Table 9, we can see

Table 9: Adversarial training as a defense strategy.

Dataset	# of Legitimate Audio	# of Adversarial Audio	Target Model	Accuracy	Success rate
Speech Commands	20,000	1,000	CNN	94.22%	17.90%
Synthesized Commands	33,000	1,100	VGG19	90.84%	23.30%
IEMOCAP	10,000	1,000	ResNet18	85.79%	20.50%

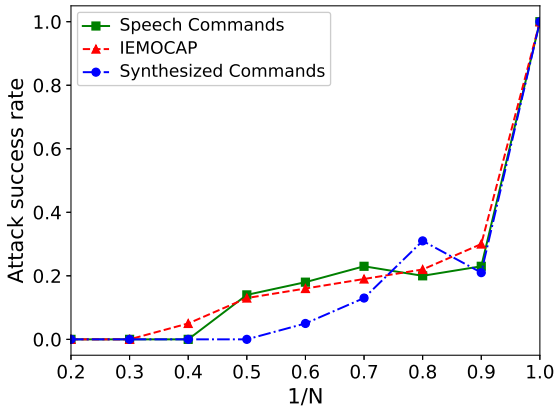


Figure 9: Results of the audio downsampling defense on three datasets.

that the success rate of adversarial audios decreases while the models' performance on legitimate samples does not change too much. However, a limitation of adversarial training is that it needs to know the details of the attack strategy and to have sufficient adversarial audios for training. In practice, however, attackers usually do not make their approaches or adversarial audios public. Further, they can change the parameters of the attack frequently (e.g., the perturbation factor [16]) to evade the defense. Therefore, adversarial training is limited in defending unknown adversarial attacks.

Audio Downsampling. The second potential defense method is to reduce the sampling rate of the input audio $\boldsymbol{x}$. We denote the downsampled audio as $D(\boldsymbol{x})$, and its recognition result is referred to as y_D. When we feed an acoustic system with an adversarial audio $\boldsymbol{x_{adv}}$ with label y_{adv}, if $y_D \neq y_{adv}$, $\boldsymbol{x_{adv}}$ will be determined as successfully defended. The results of this defense are shown in Fig. 9, where x-axis means that the original sampling rate is N times of the downsampled rate. From Fig. 9, we can see that this defense can reduce the success rate of SirenAttack. For instance, when $\frac{1}{N} = 0.8$, the success rate of SirenAttack is 20%. However, according to the Nyquist sampling theorem [51], this method would cause distortion when the sampling rate is below twice of the highest frequency of the original audios.

Moving Average Filtering (MAF). Now, we use a sliding window with a fixed length for MAF to reduce the impact of adversarial noise. Specifically, for a sampling point x_i, we consider the $k-1$ points before and after it as local reference points and replace x_i by the average value of its reference points. The results are shown in

Table 10: Moving Average Filtering as a defense strategy.

Dataset	# of Adversarial Audios	Model	k	Success Rate
Speech Commands	1,000	CNN	5	20.60%
Synthesized Commands	1,100	VGG19	5	4.90%
IEMOCAP	1,000	ResNet18	5	28.50%

Table 10, from which we can see that MAF can reduce the success rate of SirenAttack. For instance, when $k = 5$, the success rate of SirenAttack decreases to 20.60% on the Speech Commands Dataset. However, MAF might reduce the quality of audios, thus having a negative impact on the models' performance.

9 DISCUSSION

Universal Adversarial Perturbations. In SirenAttack, we need to find special adversarial noises for each audio. In the image domain, it is possible to construct a single perturbation δ that can lead to misclassifications for various images when applied to them [36]. This kind of attack would also be incredibly powerful to audio if it is possible. We take this as a future research direction.

More Threatening Attacks. In our evaluation, we assume that an attacker can directly feed the audio files to the victim model. This is realistic since many speech content monitors can directly censor the raw audios. Therefore, SirenAttack would indeed pose a threat on the web environment. However, a more powerful attack scene is "over-the-air", where an attacker calculates and plays an adversarial noise signal $\boldsymbol{\delta(t)}$ according to a legitimate audio $\boldsymbol{x(t)}$ in real time, so that the superimposed audio $\boldsymbol{x(t) + \delta(t)}$ would be interpreted as a malicious command. In addition, SirenAttack can be combined with other attacks to form more dangerous ones, e.g., combine SirenAttack with GVS-Attacks [12] so that the malware can replay an adversarial audio when it finds an opportunity. We also plan to study this attack in the future.

Other Limitations and Future Work. Although the generated adversarial audios can be misclassified by popular ASR platforms, the success rate is not very high. Therefore, how to generated adversarial audios with better transferability deserves further research. Furthermore, developing effective and robust defense schemes is also a promising future work.

10 CONCLUSION

In this paper, we study targeted adversarial attacks against acoustic systems in both white-box and black-box settings. To the best of our knowledge, this is the first systematical study on generating adversarial audios for various acoustic systems, including speech recognition, speaker recognition and sound event classification. Extensive experimental results show that SirenAttack is effective and efficient, and has potential threats on many real applications. We also discuss three potential approaches to defend against such attacks.

ACKNOWLEDGMENTS

This work was partly supported by the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. LR19F020003, the Provincial Key Research and Development Program of Zhejiang, China under No. 2019C01055, NSFC under No. 61772466, U1936215, and U1836202, the Ant Financial Research Funding, the National Key Research and Development Program of China under No. 2018YFB0804102, and the Alibaba-ZJU Joint Research Institute of Frontier Technologies.

REFERENCES

[1] 2018. *ShotSpotter.* Retrieved Octobor 21, 2018 from http://www.shotspotter.com/
[2] Enrique Alexandre, Lucas Cuadra, Manuel Rosa, and Francisco Lopez-Ferreras. 2007. Feature Selection for Sound Classification in Hearing Aids Through Restricted Search Driven by Genetic Algorithms. *IEEE Transactions on Audio, Speech, and Language Processing* 15, 8 (2007), 2249–2256.
[3] Pradeep K Atrey, Namunu C Maddage, and Mohan S Kankanhalli. 2006. Audio based event detection for multimedia surveillance. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Toulouse, France), Vol. 5. IEEE, 813–816. https://doi.org/10.1109/ICASSP.2006.1661400
[4] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. 2016. End-to-end attention-based large vocabulary speech recognition. In *Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 4945–4949.
[5] Eric Battenberg, Jitong Chen, Rewon Child, Adam Coates, Yashesh Gaur, Yi Li, Hairong Liu, Sanjeev Satheesh, David Seetapun, Anuroop Sriram, et al. 2017. Exploring neural transducers for end-to-end speech recognition. In *Automatic Speech Recognition and Understanding Workshop (ASRU).* IEEE, 206–213.
[6] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. IEMOCAP: Interactive emotional dyadic motion capture database. *Language resources and evaluation* 42, 4 (2008), 335.
[7] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden Voice Commands.. In *USENIX Security* (Austin, TX, USA). USENIX Association, Berkeley, CA, USA, 513–530.
[8] Nicholas Carlini and David Wagner. 2018. Audio Adversarial Examples: Targeted Attacks on Speech-to-Text. (2018). arXiv:1801.01944
[9] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. 2017. Dual path networks. In *Advances in Neural Information Processing Systems.* 4470–4478.
[10] Moustapha M Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. 2017. Houdini: Fooling Deep Structured Visual and Speech Recognition Models with Adversarial Examples. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 6977–6987.
[11] Pranab Kumar Dhar and Tetsuya Shimamura. 2015. *Advances in audio watermarking based on singular value decomposition.* Springer.
[12] Wenrui Diao, Xiangyu Liu, Zhe Zhou, and Kehuan Zhang. 2014. Your voice assistant is mine: How to abuse speakers to steal information and control your phone. In *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM).* ACM, 63–74.
[13] Russell Eberhart and James Kennedy. 1995. A new optimizer using particle swarm theory. In *International Symposium on Micro Machine and Human Science (MHS'95).* IEEE, 39–43.
[14] Huan Feng, Kassem Fawaz, and Kang G Shin. 2017. Continuous authentication for voice assistants. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking (MobiCom).* 343–355.
[15] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio set: An ontology and human-labeled dataset for audio events. In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing.* 776–780.
[16] Yuan Gong and Christian Poellabauer. 2017. Crafting Adversarial Examples For Speech Paralinguistics Applications. (2017). arXiv:1711.03280
[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR).* 1–11.
[18] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *International Conference on Machine learning (ICML).* ACM, 369–376.
[19] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. 2014. Deep speech: Scaling up end-to-end speech recognition. (2014). arXiv:1412.5567
[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 770–778.

[21] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. 2017. CNN architectures for large-scale audio classification. In *International Conference on Acoustics, Speech and Signal Processing*. IEEE, 131–135.

[22] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[23] Dan Iter, Jade Huang, and Mike Jermann. 2017. Generating adversarial examples for speech recognition. (2017).

[24] Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, and Ting Wang. 2018. Model-reuse attacks on deep learning systems. In *CCS*. 349–363.

[25] C. Kereliuk, B. L. Sturm, and J. Larsen. 2015. Deep Learning and Music Adversaries. *IEEE Transactions on Multimedia* 17, 11 (Nov 2015), 2059–2071. https://doi.org/10.1109/TMM.2015.2478068

[26] Juntae Kim and Minsoo Hahn. 2018. Voice Activity Detection Using an Adaptive Context Attention Model. *IEEE Signal Processing Letters* 25, 8 (Aug 2018), 1181–1185. https://doi.org/10.1109/LSP.2018.2811740

[27] Anurag Kumar, Maksim Khadkevich, and Christian Fugen. 2017. Knowledge Transfer from Weakly Labeled Audio using Convolutional Neural Network for Sound Events and Scenes. (2017). arXiv:1711.01369

[28] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. In *Workshop of the 5th International Conference on Learning Representations (ICLR)*.

[29] Xinyu Lei, Guan-Hua Tu, Alex X Liu, Chi-Yu Li, and Tian Xie. 2017. The Insecurity of Home Digital Voice Assistants-Amazon Alexa as a Case Study. *arXiv preprint arXiv:1712.03327* (2017).

[30] Jinfeng Li, Tianyu Du, Shouling Ji, Rong Zhang, Quan Lu, Min Yang, and Ting Wang. 2020. TextShield: Robust Text Classification Based on Multimodal Embedding and Neural Machine Translation. In *USENIX Security*.

[31] Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. TextBugger: Generating Adversarial Text Against Real-world Applications. In *NDSS*.

[32] Xurong Li, Shouling Ji, Meng Han, Juntao Ji, Zhenyu Ren, Yushan Liu, and Chunming Wu. 2019. Adversarial examples versus cloud-based detectors: A black-box empirical study. *IEEE Transactions on Dependable and Secure Computing* (2019).

[33] Jae-Deok Lim, Jeong-Nyeo Kim, Young-Giu Jung, Young-Doo Yoon, and Cheol-Hoon Lee. 2014. Improving performance of X-rated video classification with the optimized repeated curve-like spectrum feature and the skip-and-analysis processing. *Multimedia Tools Appl.* 71, 2 (May 2014), 717–740.

[34] Xiang Ling, Shouling Ji, Jiaxu Zou, Jiannan Wang, Chunming Wu, Bo Li, and Ting Wang. 2019. Deepsec: A uniform platform for security analysis of deep learning model. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 673–690.

[35] Kuba Łopatka, Paweł Zwan, and Andrzej Czyżewski. 2010. *Dangerous Sound Event Recognition Using Support Vector Machine Classifiers*. Springer Berlin Heidelberg, Berlin, Heidelberg, 49–57. https://doi.org/10.1007/978-3-642-14989-4_5

[36] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal Adversarial Perturbations. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 86–94.

[37] Mozilla. 2017. Common Voice Dataset. https://voice.mozilla.org/zh-CN/data

[38] Anh Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 427–436.

[39] Douglas O'Shaughnessy. 2008. Automatic speech recognition: History, methods and challenges. *Pattern Recognition* 41, 10 (2008), 2965–2979.

[40] Giuseppe Petracca, Yuqiong Sun, Trent Jaeger, and Ahmad Atamli. 2015. Audroid: Preventing attacks on audio channels in mobile devices. In *Annual Computer Security Applications Conference (ACSAC)* (Los Angeles, CA, USA) *(ACSAC '15)*. ACM, New York, NY, USA, 181–190. https://doi.org/10.1145/2818000.2818005

[41] Karol J Piczak. 2015. ESC: Dataset for environmental sound classification. In *MM '15 Proceedings of the 23rd ACM international conference on Multimedia* (Brisbane, Australia) *(MM '15)*. ACM, New York, NY, USA, 1015–1018. https://doi.org/10.1145/2733373.2806390

[42] Arnab Poddar, Md Sahidullah, and Goutam Saha. 2018. Speaker verification with short utterances: a review of challenges, trends and opportunities. *IET Biometrics* 7, 2 (2018), 91–101. https://doi.org/10.1049/iet-bmt.2017.0065

[43] Tara N Sainath and Carolina Parada. 2015. Convolutional neural networks for small-footprint keyword spotting. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 1478–1482.

[44] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. 2014. A Dataset and Taxonomy for Urban Sound Research. In *International Conference on Multimedia* (Orlando, Florida, USA). ACM, New York, NY, USA, 1041–1044. https://doi.org/10.1145/2647868.2655045

[45] Chenghui Shi, Xiaogang Xu, Shouling Ji, Kai Bu, Jianhai Chen, Raheem Beyah, and Ting Wang. 2019. Adversarial captchas. *arXiv preprint arXiv:1901.01107* (2019).

[46] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)* (2015).

[47] Christian Szegedy. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*.

[48] Michel Vacher, Jean-François Serignat, and Stephane Chaillol. 2007. Sound classification in a smart room environment: an approach using GMM and HMM methods. In *Proceedings of the 4th IEEE Conference on Speech Technology and Human-Computer Dialogue (SpeD 2007)* (Bucharest) *(Advances in Spoken Language Technology)*, Vol. 1. Publishing House of the Romanian Academy, Iasi, Romania, 135–146. https://hal.archives-ouvertes.fr/hal-00957418

[49] Christophe Veaux, Junichi Yamagishi, Kirsten MacDonald, et al. 2017. CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit. (2017).

[50] Pete Warden. 2017. Speech Commands: A public dataset for single-word speech recognition. *Dataset available from http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz* (2017).

[51] Wikipedia. 2018. Wikipedia, Nyquist-Shannon sampling theorem. https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem

[52] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5987–5995.

[53] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, XiaoFeng Wang, and Carl A. Gunter. 2018. CommanderSong: A Systematic Approach for Practical Adversarial Voice Recognition. In *27th USENIX Security Symposium*. USENIX Association, 49–64.

[54] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. (2016). arXiv:1605.07146

[55] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. 2017. DolphinAttack: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS)* (Dallas, Texas, USA). ACM, New York, NY, USA, 103–117. https://doi.org/10.1145/3133956.3134052

[56] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. 2020. Interpretable deep learning under fire. In *USENIX Security*.