AdvPulse: Universal, Synchronization-free, and Targeted Audio Adversarial Attacks via Subsecond Perturbations

Zhuohang Li
University of Tennessee
Knoxville, TN, USA
zli96@vols.utk.edu

Yi Wu
University of Tennessee
Knoxville, TN, USA
ywu83@vols.utk.edu

Jian Liu*
University of Tennessee
Knoxville, TN, USA
jliu@utk.edu

Yingying Chen
Rutgers University
New Brunswick, NJ, USA
yingche@scarletmail.rutgers.edu

Bo Yuan
Rutgers University
New Brunswick, NJ, USA
bo.yuan@soe.rutgers.edu

ABSTRACT

Existing efforts in audio adversarial attacks only focus on the scenarios where an adversary has prior knowledge of the entire speech input so as to generate an adversarial example by aligning and mixing the audio input with corresponding adversarial perturbation. In this work we consider a more practical and challenging attack scenario where the intelligent audio system takes streaming audio inputs (e.g., live human speech) and the adversary can deceive the system by playing adversarial perturbations simultaneously. This change in attack behavior brings great challenges, preventing existing adversarial perturbation generation methods from being applied directly. In practice, (1) the adversary cannot anticipate *what* the victim will say: the adversary cannot rely on their prior knowledge of the speech signal to guide how to generate adversarial perturbations; and (2) the adversary cannot control *when* the victim will speak: the synchronization between the adversarial perturbation and the speech cannot be guaranteed. To address these challenges, in this paper we propose *AdvPulse*, a systematic approach to generate subsecond audio adversarial perturbations, that achieves the capability to alter the recognition results of streaming audio inputs in a targeted and synchronization-free manner. To circumvent the constraints on speech content and time, we exploit penalty-based universal adversarial perturbation generation algorithm and incorporate the varying time delay into the optimization process. We further tailor the adversarial perturbation according to environmental sounds to make it inconspicuous to humans. Additionally, by considering the sources of distortions occurred during the physical playback, we are able to generate more robust audio adversarial perturbations that can remain effective even under over-the-air propagation. Extensive experiments on two representative types of intelligent audio systems (i.e., speaker recognition and speech command recognition) are conducted in various realistic environments. The results show that our attack can achieve an average attack success rate of over 89.6% in indoor environments and 76.0% in inside-vehicle scenarios even with loud engine and road noises.

*Corresponding Author.

CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → *Machine learning*;

KEYWORDS

audio adversarial attack; synchronization-free; intelligent audio system

ACM Reference Format:
Zhuohang Li, Yi Wu, Jian Liu, Yingying Chen, and Bo Yuan. 2020. AdvPulse: Universal, Synchronization-free, and Targeted Audio Adversarial Attacks via Subsecond Perturbations. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20), November 9–13, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3372297.3423348

1 INTRODUCTION

Since audio interaction using one's voice has become a major convenience in modern times, voice interaction in our daily lives has been becoming either an alternative or even a full replacement of traditional graphical user interfaces. Especially, recent advances in deep learning techniques have enabled machines to achieve a near-human performance in both understanding the speech content (i.e., speech recognition) [21] and identifying the speaker from vocal traits (i.e., speaker recognition) [25]. Benefiting from such an unprecedented performance advancement, modern deep-learning-powered intelligent audio systems have been widely integrated in almost every corner of our daily lives. For instance, people can talk to their smartphones (e.g., Siri [8], Bixby [40]) or smart speakers (e.g., Google Home [20], Amazon Echo [4]) to set an alarm, inquire personal schedules, and control smart home appliances, etc. Mobile banking (e.g., Chase Voice ID [7]) exploits remote voice authentication to quickly verify users and prevent fraud, and drivers can operate their cars and access their functions simply through voice commands (e.g., Hey BMW [23], Tesla Voice Commands [46]).

With the ever-growing deployment of such intelligent audio systems, their vulnerabilities have gained considerable attention and have become an increasing public concern recently. Specifically, deep neural networks (DNNs), serving as the computational core of the state-of-the-art intelligent audio systems, are revealed to be inherently vulnerable to *adversarial attack*. This attack is where the adversary can add imperceptible adversarial perturbations to speech inputs to deceive DNN models, making the models yield false predictions. This type of attack was initially discovered in the image domain [9, 19, 33, 45] and has since spurred research interests in the audio domain.

Existing studies [3, 10, 13] demonstrated that it is possible for the adversary to inject unnoticeable adversarial perturbations to the original audio and alter the transcription result. An existing study [30] in the speaker verification task also showed that adding carefully-crafted adversarial perturbations can lead to impostors being falsely recognized as legitimate speakers. However, these studies only launched digital domain attacks, which feed the crafted adversarial examples to the intelligent audio systems directly without considering any physical effects (e.g., audio distortion, ambient noises) during the over-the-air propagation in practical scenarios. To improve the robustness of the generated adversarial examples, several recent studies [31, 52] use room impulse response (RIR) to encode the acoustic channel state information (CSI) and launch practical over-the-air attacks. There are also other efforts attempting to make the adversarial noise imperceptible to humans by leveraging psychoacoustic effects [38, 41] or extend the over-the-air attack range by employing domain adaption algorithms [12].

Limitations of Prior Research. Despite the initial success of the over-the-air audio adversarial attacks (e.g., [12, 31, 52]), they only focus on the scenarios where the adversary has prior knowledge of the entire speech input (i.e., *Static-speech attack scenario* as shown in Figure 1(a)). For each specific pre-recorded speech, the adversary can add an adversarial perturbation to form an adversarial example, which can then be played by a loudspeaker to deceive intelligent audio systems. However, these attacks are not applicable to *streaming-speech attack scenarios*, which are more practical and common in the daily use of intelligent audio systems, as shown in Figure 1(b). In this scenario, the intelligent audio system takes streaming audio inputs (e.g., live human speech) and the adversary can fool the system by playing imperceptible adversarial perturbations through a nearby loudspeaker. We summarize the following three major limitations preventing existing attacks from being launched in practice with streaming audio input below:

(1) Modifying the Entire Audio Input. Most existing attacks require the generated adversarial perturbation to be added on the entire audio input. In other words, the generated perturbation should have the exact same duration as the audio input, which is not feasible when handling streaming inputs.

(2) Synchronization. Existing attacks are based on the assumption that the input audio and the generated adversarial perturbation are strictly synchronized. To guarantee the synchronization, the adversarial perturbation is usually mixed with the audio input beforehand and then played through a loudspeaker when launching the attack.

(3) Prior Knowledge on the Audio Input. Most existing attacks require the adversary to have access to the input audio in advance. That means the adversary needs to first collect an audio clip and then compute the adversarial perturbation specifically for that piece of audio, which is not applicable to streaming input either.

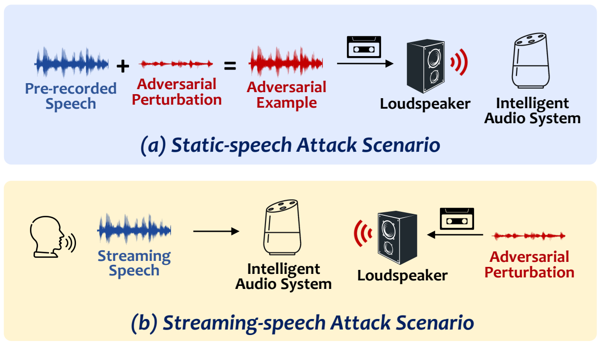


Figure 1: Comparison of the static-speech attack scenario used in existing audio adversarial attacks and the proposed streaming-speech attack scenario.

Subsecond-level Universal, Synchronization-free, and Targeted Adversarial Perturbation. Due to the time-sensitive behavior of audio signals, it is unrealistic to assume that the adversary can anticipate the upcoming audio signal and allocate a particular amount of time to modify the entire audio signal in a synchronized manner. The above limitations show that the existing attacks are only feasible in controlled static-speech scenarios (e.g., playing prefabricated audio adversarial examples). To circumvent these limitations in attacking streaming-speech-involved audio intelligent systems, in this paper, we propose *AdvPulse*, a means to generate a subsecond-level adversarial perturbation which can be added at any point of the streaming audio input to launch targeted adversarial attacks. The workflow of the proposed attack is illustrated in Figure 2. Specifically, ① instead of modifying the entire audio input, we only need to add a very short adversarial perturbation of ~ 0.5 seconds to the audio input; ② we do not require synchronization between the input audio and the adversarial perturbation (i.e., the adversarial perturbation can be injected anywhere in the streaming audio input); and ③ instead of crafting adversarial perturbation for each specific input, we generate input-agnostic universal adversarial perturbation that can make arbitrary audio input (i.e., streaming speech) to be mis-recognized as the adversary-desired label.

To launch such an adversarial attack to deceive intelligent audio systems with streaming-speech input, we designed a series of mechanisms to address the aforementioned major limitations. Specifically, to release the requirement of synchronization, we propose to add a subsecond audio adversarial perturbation and adopt gradient-based adversarial machine learning algorithm to maximize the expected output probability of the target class over different delay conditions. This process enables that the adversarial perturbation can be added at any timestamp of the audio input while maintaining effective. Moreover, we exploit penalty-based universal training on a set of speech samples to craft audio-agnostic adversarial perturbation that can be added to arbitrary speech (e.g., streaming speech) causing the intelligent audio system to output any adversary-desired label. To make the generated adversarial perturbation unnoticeable to humans, we add more restrictions in its generation phase, making the short adversarial perturbation resemble environmental sounds (e.g., bird singing, car horns, or HVAC

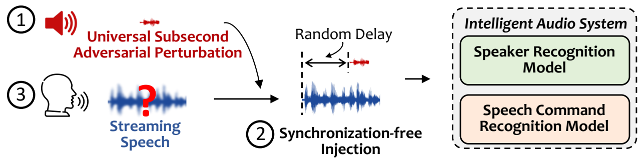

1
Universal Subsecond Adversarial Perturbation
Random Delay
3
Streaming Speech
2 Synchronization-free Injection
Intelligent Audio System
Speaker Recognition Model
Speech Command Recognition Model

Figure 2: Workflow of the proposed attack.

noises). Additionally, to improve the robustness of the generated adversarial perturbation during physical playback, we utilize several techniques to address main sources of over-the-air distortion (e.g., speaker and microphone limitations, reflection and reverberation effects, and ambient noises). To expand the understanding of the real-world vulnerability of intelligent audio systems, we evaluated our adversarial attack on two representative types of intelligent audio systems: X-vectors [44], the state-of-the-art DNN-based speaker recognition model, and Google's speech command recognition model [39][1]. We summarize our main contributions as follows:

- To the best of our knowledge, this is the first work to design a universal, synchronization-free and targeted adversarial attack against intelligent audio systems, particularly for streaming-speech attack scenario (Figure 1(b)), using subsecond adversarial perturbations.
- We utilize the penalty-based universal adversarial perturbation generation algorithm and optimize the adversarial perturbation over the entire time delay distribution, rendering the attack to be robust to arbitrary streaming audio with varying time delay conditions.
- We propose to use an environmental sound mimicking technique to make the generated subsecond adversarial perturbation resemble situational sound effects (e.g., phone's notification sound, car horns), making the generated perturbation more inconspicuous to human listeners.
- By incorporating the main sources of distortion occurred during physical playback (i.e., frequency response of speaker and microphone, reflection and reverberation, and ambient noise) into the adversarial perturbation generation process, the effectiveness of adversarial perturbations in physical over-the-air environments can be kept.
- We performed case studies on both speaker recognition and speech command recognition models. Extensive experiments are conducted in both the digital and physical domains, including inside an office, an apartment, and inside-vehicle environments. The results show that our attack can achieve a high attack success rate in deceiving these models with streaming-speech inputs (e.g., overall 89.2% and 90.3% for speaker recognition and speech command recognition in realistic settings, respectively).

2 RELATED WORK

Individual audio adversarial attacks. Research efforts on defeating intelligent audio systems started with *individual adversarial attacks*, which specifically craft unique perturbations for each individual audio sample to fool the system. For example, in speech recognition tasks (i.e., speech-to-text transcription), Cisse *et al.* [13] first showed that small distortions added to the input audio can lead to false transcriptions. Several follow-up studies leveraging genetic algorithm [3] or iterative gradient descent approaches [10, 12, 38, 52] demonstrated the success of generating more powerful adversarial examples that can alter the transcription to any adversary-desired content. In speaker recognition tasks, existing studies [11, 30, 31, 55] found that it is possible to manipulate the recognition result (e.g., making an impostor recognized as legitimate user) by injecting adversarial perturbations to the original audio. However, the aforementioned methods require prior knowledge on each individual speech input sample to craft corresponding adversarial perturbations, resulting in tremendous computational cost when generating multitude adversarial examples. This also limits the attack to merely time-insensitive settings such as generating adversarial examples from pre-recorded audio, and has prevented it from being deployed in more realistic scenarios (e.g., injecting adversarial perturbations on live human speech) where collecting audio in advance is unfeasible. A more recent study [17] proposed to use a DNN network that is trained with reinforcement learning techniques to take streaming audio input and forecast the adversarial perturbation. However, this work only focused on launching untargeted attacks and does not consider over-the-air attack either. In addition, the generated adversarial perturbation still requires synchronization with the audio to some extent, which is hard to achieve for streaming-speech attacks.

Universal audio adversarial attacks. To circumvent the time constraint brought by individual adversarial attacks, several studies proposed *universal adversarial perturbation*: an adversarial perturbation that is expected to work with any input speech signal, causing the model to make false prediction. Specifically, in speech recognition tasks, existing studies use the iterative greedy algorithm [32] to make arbitrary speech to be mis-classified [47] or falsely transcribed [35]. In speaker recognition tasks, a recent study [51] proposed to add audio-agnostic universal perturbations on arbitrary enrolled speaker's voice input in order to make the model identify the speaker as any adversary-desired speaker label. To handle audio inputs with varying audio length during optimization, Xie *et al.* [51] proposed to use repeated copies of a universal noise pattern to construct the adversarial perturbation, which is then cropped to fit the length of the given audio at test time. However, the effectiveness of this attack still relies on the synchronization between the adversarial perturbation and the audio input, which is achieved by first mixing the two signals into a single audio adversarial example and then testing the model on the generated single-channel audio. Thus, this is not feasible when launching streaming-speech attacks in practice. In summary, all the aforementioned universal attacks assume that the universal perturbation should be perfectly aligned with all the input audio signals. This essentially requires the adversary to have the knowledge of the exact timestamp to launch the attack. Moreover, to accommodate the adversarial perturbation, the input signal needs to be tailored to match the length of the perturbation. These strict temporal constraints have prevented the attack from being effectively launched in many practical attack scenarios.

Over-the-air audio adversarial attacks. Most existing audio adversarial attacks are based on digital domain assumption, i.e., feeding adversarial examples to the model directly. They are most

[1] Audio and video samples are available at https://mosis.eecs.utk.edu/advpulse.html

likely to lose their effectiveness after practical over-the-air playback due to the accompanying audio distortions (e.g., attenuation, multi-path effect, and ambient noises). Although several studies [2, 11, 54, 55] evaluated the performance of audio adversarial attacks via over-the-air propagation, the real-world distortion factors were not thoroughly investigated in the adversarial example generation process, leaving the practicality still very limited. To address this issue, room impulse response (RIR) was integrated [31, 52] to increase the robustness of the generated audio adversarial examples. Moreover, a recent study [12] showed that the transmission range can be further extended using domain adaption algorithms. Although these studies successfully achieved physical attacks, they still require prior knowledge of the input speech signal. The input signal also needs to be mixed with the well-aligned adversarial perturbation before being played by a loudspeaker to ensure synchronization. They cannot work in the scenario where the adversary wants to inject adversarial perturbations on live human speech to deceive intelligent audio systems.

Our solution. Unlike existing studies, to the best of our knowledge, we are the first to launch a *targeted* audio adversarial attack in an *audio-agnostic* and *synchronization-free* manner. The generated universal adversarial perturbations can be added to any part of *arbitrary speech signal* (e.g., streaming speech), forcing the DNN model to output the adversary-desired label. This is different from the existing attacks that require the added adversarial perturbation to be of the same duration as the input audio as we only inject a very short adversarial perturbation (e.g., ~0.5 seconds) without the need of synchronization. These properties can largely extend our possible attack scenarios, which makes attacking live-human's voice inputs feasible.

3 BACKGROUND

3.1 Problem Formalization

An intelligent audio system (e.g., speaker or speech command recognition) can be modeled as a function $f(\cdot)$ which takes an audio input waveform $x \in [-1, 1]^n$ of n samples and outputs the probability score $P = [p_0, p_1, ..., p_{m-1}]$, where $p_i \in [0, 1]$, and $\sum_{i=0}^{m-1} p_i = 1$ for a set of m classes. The audio input will be recognized by the model as the class with the highest probability score (i.e., $y_{pred} = argmax(f(x))$). The adversarial objective is to construct an audio adversarial example $x' \in [-1, 1]^n$ by adding a perturbation δ to the original audio input x with the following properties:

Subsecond Perturbation. Instead of using a perturbation with the same duration of the audio input as is used in existing studies (e.g., [10, 12, 31, 38, 52]), we added a much shorter perturbation $\delta \in [-1, 1]^l$, where $l \ll n$, to make it sound more inconspicuous.

Targeted. By introducing the adversarial perturbation, the recognition result can be changed from the true label y to the adversary-desired target label y_t, namely, $argmax(f(x)) = y$ but $argmax(f(x')) = y_t$, where $y_t \neq y$.

Synchronization-free. When launching the attack in a real-world scenario, such as injecting adversarial perturbation to the streaming speech input, it's impossible to add the perturbation at a particular point of the streaming speech. In other words, the synchronization between the adversarial perturbation and the audio input cannot be guaranteed, and there usually exists a time delay τ between the two signals. Therefore, the adversarial perturbation signal $\delta[t]$, as a function of time t, needs to be applicable at any time during the audio input playback regardless of the time delay: $argmax(f(x[t] + \delta[t - \tau])) = y_t$, where $\tau \in [0, n - l]$.

Universal. In most practical scenarios, observing the audio input in advance is not plausible. Therefore, the constructed adversarial perturbation is required to be effective on arbitrary audio input (e.g., streaming speech).

Unnoticeable. The distortion introduced by the added adversarial perturbation δ should be relatively small to make the attack unnoticeable to human. As the duration of the generated perturbation is very short, we deliberately make it sound like environmental sound (e.g., bird singing, car horns, or HVAC noises) to further hide itself while being played back over the air.

Robust for Over-the-air Attack. In order to launch a physical attack, the adversarial perturbation should be robust enough to survive real-world distortions brought by physical playbacks such as reverberations and ambient noises.

3.2 Threat Model

The aforementioned properties enable the proposed attack to be launched in a broad range of real-world scenarios. Unlike existing studies that require to play the well-crafted adversarial example (i.e., the original speech mixed with the adversarial perturbations), in this paper, we circumvent all the three major limitations described in Section 1 and consider a more common and serious live-speech-involved scenario where an adversary seeks to launch the attack by only playing the constructed adversarial perturbation. The constructed perturbation is audio-agnostic and does not need to be synchronized with the audio input, which largely extends its potential attack functionality. Additionally, the length of the perturbation is in a subsecond level and is made to sound like environmental sounds (e.g., phone's notification sound), making the whole attack process inconspicuous to people.

Possible Attack Scenarios. Our attack is applicable to the *static-speech attack scenario* (Figure 1(a)), where the adversary needs to create the adversarial example (i.e., the audio input mixed with the adversarial perturbation) and then play it back through a loudspeaker. In this scenario, the adversarial example could be played inconspicuously by the adversary to make the intelligent audio systems inadvertently recognize and obey the adversary-desired intention. For instance, the speaker recognition module could be fooled to mistakenly recognize that the speech was uttered by an adversary-desired speaker. The speech command recognition module can also be controlled to execute the mistakenly recognized malicious command. More importantly, the proposed universal, synchronization-free, and targeted adversarial attack makes the *streaming-speech attack scenario* (Figure 1(b)) feasible. In this scenario, by playing the well-crafted universal adversarial perturbation through a loudspeaker, the adversary can compromise the intelligent audio system while interacting with an actual person. Similarly, the perturbation stealthily played by the loudspeaker can make either speaker recognition or speech command recognition modules mistakenly recognize the streaming audio input as the adversary-desired speaker or malicious command respectively. This constructed perturbation is very short (subsecond level) and sounds like environmental sounds, which can be periodically played by a nearby loudspeaker (smart TV loudspeaker, in-vehicle/in-ceiling/on-wall loudspeakers) without raising the victim's suspicion. It can also be embedded in the audio tracks of regular media (e.g., Youtube videos, radios or TVs), potentially deceiving all the intelligent audio systems exposed to the media.

Challenges of Adversary. Due to the inherent sequence order and time-varying behavior of live speech, launching such an attack in the *streaming-speech attack scenario* poses several challenges to the adversary: *(1) Independence of Audio Input.* In the streaming-speech scenario, the adversary cannot anticipate the upcoming audio from the actual speaker (i.e., each speech is unique and will be only uttered once), which prohibits the adversary from optimizing the adversarial perturbation for specific audio signal in advance. This requires the adversary to generate a *universal* adversarial perturbation that can remain effective on arbitrary audio inputs from the user. *(2) Independence of Emission Time.* The adversary has no prior knowledge on the exact time when the speech will be uttered. As a result, the generated adversarial perturbation needs to have *synchronization-free* properties, meaning that the adversarial perturbation should remain effective regardless of the emission time during the interaction between the user and the system.

Adversary's Capability. The attack workflow of most existing studies (e.g., [12, 31, 52, 55]) are two-fold: first, calculate the adversarial perturbation δ for a given audio input x and synchronously apply the adversarial perturbation to get the adversarial example $x' = x + \delta$; second, play the prefabricated audio adversarial example through a loudspeaker. However, in our proposed attack, we assume that the adversary has no control over the streaming audio input x in terms of speech content and emission time. Moreover, the adversary can have some prior knowledge of the possible environmental sounds (e.g., bird singing, car horns, or HVAC noises) of the target intelligent system so as to make the adversarial perturbation more difficult to be noticed by people. In addition, to craft the adversarial perturbation, we assume that the adversary has knowledge over the architecture and the parameters of the model (i.e., a white-box setting), as is used in most previous studies (e.g., [10, 12, 31, 38, 52]).

3.3 Target Models

Speaker Recognition Model. In this work, we used *X-vectors* system [44] as our target speaker recognition model, which is the state-of-the-art text-independent DNN-based model and has been used as baseline in several follow-up studies (e.g., [43, 50, 56]). Specifically, X-vectors system is composed of three building blocks: Mel-frequency cepstral coefficients (MFCC) feature extraction, DNN embedding model, and the probabilistic linear discriminant analysis (PLDA) module. The DNN embedding model is structured by stacking five time-delayed [36] layers, a statistic pooling layer and two affine layers. The DNN embedding model is pre-trained in an end-to-end manner with a categorical cross entropy loss, and a separately trained PLDA classifier is used to calculate embedding score. In our X-vectors implementation, we used the pre-trained embedding model provided in the Kaldi toolkit [37] and used the first 10 speakers (3 males and 7 females) in the VCTK corpus dataset [48] as the enrolled speakers. The detailed information about each speaker is shown in Table 5 in Appendix. Each speaker has about 400 utterances recorded at 48 kHz that were split into training and testing sets with a ratio of 4 to 1, with each audio being cropped to 1.75 s. After enrollment, the baseline speaker recognition accuracy on the testing set achieves 97.2%.

Speech Command Recognition Model. The target speech command recognition model we used is an efficient and light-weight keyword spotting model based on convolutional neural networks [39]. This model has been used as the target model for many existing attacks (e.g., [3, 17, 47]). Specifically, we used Tensorflow's [1] official implementation[2], which is trained on the voice command dataset [49] to classify 10 speech commands: "yes", "no", "up", "down", "left", "right", "on", "off", "stop", and "go". The dataset contains a total number of 46,278 recordings of 1 s sampled at 16 kHz, 80% of which is used for training. After training, the baseline command recognition accuracy on the remaining testing samples is able to achieve 89.0%.

4 DESIGN OF ADVPULSE

4.1 Synchronization-free Subsecond Targeted Adversarial Perturbation

4.1.1 Feasibility of Subsecond Adversarial Perturbation. Conventionally, the problem with crafting a targeted audio adversarial perturbation is based on how to modify each data sample of the audio input signal to make the intelligent audio model recognize it as the target class [10]. Explicitly, given an audio input $x \in [-1, 1]^n$ and the target label y_t, the adversary can solve the following formulation to obtain the targeted adversarial perturbation $\delta \in [-1, 1]^n$:

$$\begin{aligned} \text{minimize} \quad & dB_x(\delta), \\ \text{subject to} \quad & argmax\big(f(x+\delta)\big) = y_t; \\ & x + \delta \in [-1, 1]^n, \end{aligned} \tag{1}$$

where $dB_x(\delta) = dB(\delta) - dB(x)$ and it is used to measure the relative loudness of the perturbation comparing to the audio input. This can be achieved by minimizing the following objective function (a relaxation of Equation 1):

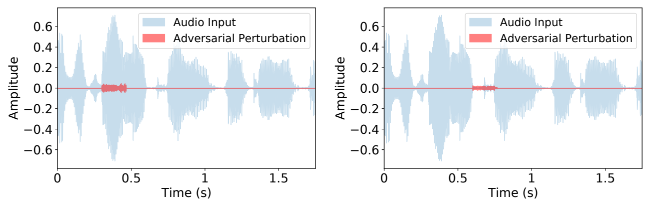
$$\begin{aligned} \text{minimize} \quad & \mathcal{L}\big(f(x+\delta), y_t\big) + \alpha \cdot ||\delta||_2, \\ \text{subject to} \quad & ||\delta||_2 \le \epsilon, \end{aligned} \tag{2}$$

where $\mathcal{L}\big(f(x+\delta), y_t\big)$ is the loss function representing the distance between the model output of the adversarial example $x + \delta$ and the target label y_t, α is the scaling coefficient, $|| \cdot ||_2$ denotes the L_2 norm, and ϵ controls the upper bound of the perturbation.

However, the requirement of modifying the entire audio naturally prohibits synchronization-free. Since the audio input x and the adversarial perturbation δ are required to have the same length and if δ is delayed by time τ, a segment of δ with length τ will be no longer acting on x. To launch a synchronization-free adversarial attack on intelligent audio systems, we thus need to first solve this question: *Is it possible to only modify part of the input signal, desirably a short segment, to fool the model?*

Commonly, intelligent audio systems process speech signals into frames, with each frame being handled by the DNN model separately (e.g., the time-delayed neural network structure used

[2]https://github.com/tensorflow/tensorflow/tree/master/tensorflow/examples/speech_commands



(a) Speech-part perturbation (b) Non-speech-part perturbation

Figure 3: Illustration of adding a subsecond adversarial perturbation on the audio input, causing the model to mistakenly recognize the speaker (*Spk-1*) as the target (*Spk-8*): (a) adding the perturbation on the speech part; and (b) adding the perturbation on the non-speech part.

in speaker recognition or the convolution neural network structure used in speech command recognition). However, the extracted feature-map of each frame is usually aggregated via a statistic pooling layer or a fully-connected layer before feeding it to the classifier or softmax layer where the final prediction is made. Therefore, by adding adversarial perturbation to only part of the speech signal, the adversary could influence the feature-map extracted from the corresponding speech frames and in turn potentially affect the final recognition result.

To further verify the feasibility of adding a short segment of adversarial perturbation to fool the model, we conducted a preliminary experiment on the speaker recognition system (i.e., X-vectors system [44]) using 10 enrolled speakers of the VCTK corpus dataset [48]. Then we solved the optimal adversarial perturbation through the same optimization process formulated in Equation 2, where we use the categorical cross entropy as the loss function. In this experiment, as shown in Figure 3, we intentionally generate two adversarial perturbations acting on the speech-part and non-speech-part of the audio input respectively, to force the model recognize *Spk-1* (i.e., the actual label of the audio input) as the target label *Spk-8*. The added adversarial perturbation is only about 0.16 seconds and can be applied to either speech-part signals or non-speech-part signals (i.e., natural pauses between words). The initial success of this attack shows the potential of synchronization-free adversarial perturbation: adding a very short perturbation regardless of its position on the audio input to fool the model.

4.1.2 Subsecond Synchronization-free Adversarial Perturbation Generation. With the inherent temporal constraint bypassed, we now discuss how to generate a subsecond audio adversarial perturbation $\delta \in [-1, 1]^l$ ($l \ll n$) that is robust to various unsynchronized conditions. Inspired by the Expectation Over Transformation (EOT) technique [6] proposed for synthesizing robust visual adversarial examples, we incorporate the time shifting caused under unsynchronized conditions into the adversarial perturbation generation process. Instead of directly solving the adversarial perturbation acting at a specific timestamp, we seek to minimize the expected effective loss when the perturbation is shifted by a delay of τ, where τ obeys the uniform distribution between the time interval 0 and $n - l$. Therefore, Equation 2 would become:

$$\text{minimize} \underset{\tau \sim U(0,n-l)}{\mathbb{E}} \left[\mathcal{L}\big(f\big(x + Shift(\delta, \tau)\big), y_t\big) + \alpha \cdot ||\delta||_2 \right], \quad (3)$$

where $Shift(\delta, \tau)$ shifts the adversarial perturbation δ by time τ. To maximize the degree of synchronization-free, we sample random time delays with a step size of 1 digital sample (i.e., the smallest unit for time measurement in audio). Through this process, the adversarial perturbation will learn to produce a generic acoustic feature characterizing the target class to fool the model regardless of timing conditions. In other words, the adversarial perturbation can be injected anywhere in the streaming audio input for deceiving intelligent audio systems, which is the essential requirement for streaming-speech attack scenario.

4.2 Universal Audio Adversarial Perturbation

In the practical steaming-speech attack scenario, prior knowledge on the audio input is usually inaccessible. We therefore need to further circumvent the constraint on the prior knowledge of audio input to enable the attack to be launched in a real-time manner.

Our goal is to find a universal subsecond adversarial perturbation $\delta \in [-1, 1]^l$ computed from a relatively small set of training data samples to force the model to recognize arbitrary new audio input (e.g., streaming audio input) as the target label with high probability. Let μ denote the distribution of the training data samples, which are a set of utterances spoken by the actual speaker for attacking *speaker recognition systems* or a set of utterances of the actual command for attacking *speech command classification systems*. We thus aim to find a universal perturbation to fool the model on almost all the audio inputs sampled from μ and alter the classification result to the desired target class with a high probability (i.e., $\forall x \sim \mu, \mathbb{P}\big(f(x+\delta) = y_t\big) \longrightarrow 1$). It worth noting that, unlike existing untargeted universal attacks [35, 47] where the goal is to fool the classifier to make false prediction (i.e., $\forall x \sim \mu, \mathbb{P}\big(f(x+\delta) \neq f(x)\big) \longrightarrow 1$), we seek to launch a *targeted universal* attack, which allows the adversary to pick the target class that they desire. We believe this is a more harmful type of attack as it offers the adversary the ability to control the attack outcome (e.g., making their voice to be recognized as a specific speaker with access privilege).

To coin such an attack, we used a penalty-based method to find the universal adversarial perturbation by optimizing the following expectation function over a set of training data samples sampled from the distribution μ:

$$\text{minimize} \underset{x \sim \mu, \tau \sim U(0,n-l)}{\mathbb{E}} \left[\mathcal{L}\big(f\big(x + Shift(\delta, \tau)\big), y_t\big) + \alpha \cdot ||\delta||_2 \right]. \quad (4)$$

Let $\mathcal{D} = \{(x_1, y_1), ..., (x_k, y_k)\}$ be a set of training data sampled from μ. To approximate the true data distribution with its sample distribution, we iterate through each data sample to update the adversarial perturbation δ by applying gradients calculated from Equation 3. The process is repeated for several epochs until the desired attack success rate is reached.

To ensure the validity of the audio produced from the above process, we need to enforce a box constraint on the output adversarial perturbation as well as the adversarial example: $\delta \in [-1, 1]^l, x' = x + \delta \in [-1, 1]^n$. This can be achieved through projected gradient descent, which clips the value of the audio after each iteration to be within the set range. However, this could yield non-optimal results in complicated update steps [9], especially in our case where multi-level clipping is needed. To mitigate this issue, we introduce a new variable z, where $\delta = tanh(z)$. This changes the box-constrained optimization on δ to unconstrained optimization on z. The pseudocode of the proposed algorithm is described in Algorithm 1,

Algorithm 1 Universal and Synchronization-free Adversarial Perturbation Generation (SGD is used for simplicity)

```
Input: Data samples $\mathcal{D} = \{(x_1, y_1), \ldots, (x_k, y_k)\}$, target model $f(\cdot)$, desired target class $y_t$, hyperparameters $\alpha, \beta$
Output: Universal and Synchronization-free Adversarial Perturbation $\delta$
1: Randomly initialize $z \leftarrow [0, 1]^l$
2: for number of epochs do
3:     $\tau \leftarrow U(0, n - l)$                          ▷ Sample random time delay
4:     for each data sample $(x_i, y_i) \in \mathcal{D}$ do
5:         $\delta \leftarrow tanh(z)$                     ▷ Get constrained audio perturbation
6:         $\delta \leftarrow Shift(\delta, \tau)$                  ▷ Time shifting
7:         $x' \leftarrow clip(x + \delta, [-1, 1])$            ▷ Craft adversarial example
8:         $\mathcal{L}_{total} \leftarrow \mathcal{L}(f(x'), y_t) + \alpha \cdot ||\delta||_2$   ▷ Compute total loss via Equation 3
9:         $z \leftarrow z - \beta \cdot \frac{\partial \mathcal{L}_{total}}{\partial z}$            ▷ Update perturbation via $z$
10:    end for
11: end for
```

where stochastic gradient descent (SGD) [27] is used for simplicity. In practice, this method could work with other gradient-based optimization algorithms, such as RMSProp [18], Adam [28], Nadam [15], or AdaGuard [16]. We empirically choose to use Adam for faster convergence.

4.3 Environmental Sound Mimicking

To further make the attack less suspicious, we tailored the generated adversarial perturbation according to environmental sounds. Specifically, the adversary can craft adversarial perturbation to sound like any situational sound that would normally appear in the environment (e.g., bird singing, car horns, or HVAC nosies). For a chosen environmental sound template $\hat{\delta}$, we introduce another term to Equation 4 to penalize the shape difference between the adversarial perturbation and the sound template:

$$\text{minimize} \underset{x\sim\mu, \tau\sim U(0,n-l)}{\mathbb{E}} \Big[\mathcal{L}\big(f\big(x + Shift(\delta, \tau)\big), y_t\big) + \alpha \cdot ||\delta||_2 + \gamma \cdot dist(\delta, \hat{\delta})\Big], \tag{5}$$

where $dist(\delta, \hat{\delta})$ denotes the measured distance between the two audio signal according to a chosen distance metric. As shown in Appendix A.3, after comparing 4 different metrics, we chose to use the L2 distance between the two time-series signals for environmental sound mimicking.

4.4 Robust Adversarial Perturbation for Over-the-air Attack

In practice, the crafted adversarial perturbation played by a loudspeaker will experience heavy distortions incurred by signal attenuation, multi-path effect, and ambient noises when propagating over the air. Such an inevitable audio distortion would make the perturbation lose its effectiveness, with a high possibility. In this subsection, we enhance the robustness of the generated adversarial perturbation to enable physical over-the-air attacks by incorporating the effects of *speaker & microphone limitation*, *absorption and reverberation* and *ambient noise* into the adversarial perturbation generation process.

4.4.1 Using Band-pass Filter to Cope with Speaker & Microphone Limitations. Audio devices such as loudspeakers and microphones are normally designed to work in the human audible frequency range (e.g., 20Hz to 20kHz). However, due to hardware limitations, most loudspeakers and microphones do not respond to these frequencies uniformly, which would cause the relative amplification in some frequency ranges and attenuation in others, in turn affecting the performance of the attack. For instance, we measure the frequency response by playing a chirp signal ranging from 2Hz to 20kHz through the Bose Companion 2 speakers and recording from an omni-directional microphone as shown in Appendix Figure 13. We can observe a clear attenuation in lower frequencies (i.e., below 50Hz). To mitigate this effect, we imposed a band-pass filter during the adversarial perturbation generation process as a constrain to limit the adversarial perturbation to be in the valid frequency range:

$$\text{minimize} \underset{x\sim\mu, \tau\sim U(0,n-l)}{\mathbb{E}} \Big[\mathcal{L}\big(f(\hat{x}), y_t\big) + \alpha \cdot ||\delta||_2 + \gamma \cdot dist(\delta, \hat{\delta})\Big], \tag{6}$$

where $\hat{x} = x + \underset{50\sim8000Hz}{BPF}\big(Shift(\delta, \tau)\big)$ and $\underset{50\sim8000Hz}{BPF}$ denotes the band-pass filter operation. We chose 8kHz as the upper cutoff frequency because it is the Nyquist frequency of 16kHz sampling rate, which is commonly used in intelligent audio systems (e.g., Google Assistant SDK [42]).

4.4.2 Using Room Impulse Response to Cope with Absorption and Reverberation. During the over-the-air propagation, the surrounding environment will lead to absorption and reverberation, causing the received audio signal to be very different from the original transmitted signal. The room impulse response (RIR) models the transfer function between the sound source and the received sound at the microphone end, thus it can be used to emulate the over-the-air distortions in the adversarial perturbation optimization process to enhance the attack's robustness. The RIR can vary largely according to different room layouts, the position of the sound source and the microphone, and the absorbent nature of each reflective surfaces. Thus, we use a group of real RIRs collected in various environments to improve the robustness of the generated adversarial perturbation regardless of the launching condition and environment. Specifically, we utilized the REVERB challenge database [29], the RWCP sound scene database [34], and the Aachen impulse response database [26], resulting in a total number of 218 physically measured RIRs under different room layout (e.g., small, medium, large room). Through integrating these RIRs, Equation 6 becomes:

$$\text{minimize} \underset{x\sim\mu, \tau\sim U(0,n-l), h\sim H}{\mathbb{E}} \Big[\mathcal{L}\big(f(\hat{x}), y_t\big) + \alpha \cdot ||\delta||_2 + \gamma \cdot dist(\delta, \hat{\delta})\Big], \tag{7}$$

where $\hat{x} = x + \underset{50\sim8000Hz}{BPF}\big(Shift(\delta, \tau)\big) \otimes h$, and h is the RIR sampled from the collected RIR distribution H. It's worth noting that if the adversary can anticipate the layout of the attacking environment (e.g., rough room size) where the attack would be launched, the adversary can further improve the attack performance by narrowing down the available RIRs to a specific subset that is specifically collected to reflect the attack environment.

4.4.3 Mitigating the Effect of Ambient Noise. In practice, ambient noise is inevitable during recording and is highly variable, ranging from continuous environmental white/pink noise (e.g., traffic

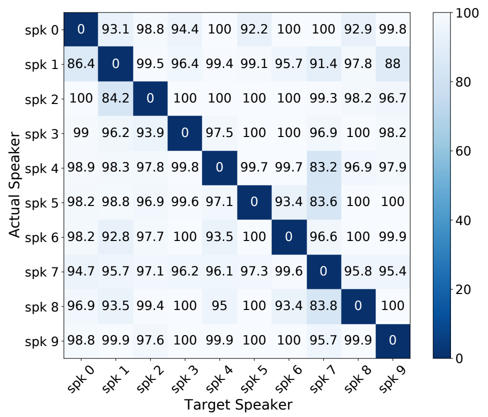


Figure 4: Performance of the digital attack on the speaker recognition model.

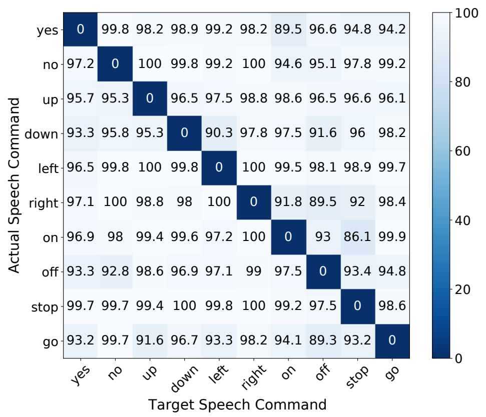


Figure 5: Performance of the digital attack on the speech command recognition model.

noise, rain sound, engine noise and air conditioning noise) to sudden sounds (e.g., phone ring, extraneous speech). Taking this into account, we utilized a set of collected ambient noise and randomly sampled an individual noise at each optimization step to solve:

$$\underset{}{\text{minimize}} \quad \mathop{\mathbb{E}}_{x\sim\mu,\tau\sim U(0,n-l),h\sim H,w\sim W}\Big[\mathcal{L}(f(\hat{x}),y_t)+\alpha\cdot||\delta||_2 + \gamma\cdot dist(\delta,\hat{\delta})\Big], \tag{8}$$

where $\hat{x} = x + \underset{50\sim8000Hz}{BPF}\big(Shift(\delta,\tau)\big)\otimes h + w$ and w is the ambient noise sampled from the noise dataset W. Specifically, we use 92 isotropic noise samples collected in different room layouts (e.g., small, medium, large) from the RWCP sound scene database [34] as our noise dataset. Similarly to RIR, instead of using a generic dataset, a sophisticated adversary can gather his own customized noise dataset for a specific attack scenario (e.g., living room, office, airport, and mall) to further improve the attack performance.

5 EVALUATION OF DIGITAL ATTACK

5.1 Experimental Methodology

Adversarial Perturbation Generation. We implemented AdvPulse on the Tensorflow [1] platform and generated digital adversarial perturbations according to Equation 5 using an NVIDIA 2080Ti GPU. As for the attack configuration, we set $\alpha = 0.01$, $\beta = 0.001$, $\gamma = 0.01$, and chose the adversarial perturbation duration to be 0.5 seconds. The impact of the perturbation durations on the attack performance is studied in Section 5.3. The total number of available audio samples per class for speaker recognition model and speech command recognition model are 47 and 100, respectively. For each class, we split the data into training and testing sets with a ratio of 4 to 1. We used a segment of bird singing, as shown in Figure 14, as the environmental sound template because this resembles phone's notification sound (e.g., Twitter's notification sound), and therefore can be used to help to launch the attack in various environments without alerting the victim.

Attack Test Datasets. Our two target models and their data settings are presented in Section 3.3, and each model has ten classes. For each model, we ran our attack to craft a specific adversarial perturbation for each actual-target label pair, resulting in a total number of 90 subsecond adversarial perturbations. To verify the synchronization-free property of the generated adversarial perturbation, we varied the delay τ from 0 seconds to the maximum possible value for each audio input (i.e., the difference between the duration of the original speech and the adversarial perturbation) with a step of 0.001 seconds and generated adversarial examples by mixing each test utterance and the delayed perturbation. In total, we generated a very large attack test dataset containing 1,125,000 and 900,000 adversarial examples (i.e., speech inputs mixed with delayed perturbations) for speaker recognition and speech command recognition modules, respectively.

Evaluation Metrics. *(1) Attack Success Rate*: This represents the number of succeeded attacks over the total number of attack attempts. Since this is a targeted attack, we only reported a success if the predicted class matches the desired target class. *(2) Confusion Matrix*: Each row represents the actual class uttered by the speaker and each column shows the target class it is classified as by the system. Each cell in the matrix corresponds to the fraction of class in the row that is classified as the class in the column. *(3) Signal-to-noise Ratio (SNR)*: The generated adversarial perturbation and the original environmental sound template are compared using SNR: $SNR = 10log_{10}(\frac{P_x}{P_p})$, where P_x and P_p are the average power of the signal and the perturbation, respectively. Larger SNR values indicate that the adversarial perturbation is closer to the environmental sound template and therefore harder to be noticed.

5.2 Attack Performance

Speaker Recognition Model. Figure 4 illustrates the detailed class-wise evaluation result on the speaker recognition model. Specifically, each cell shows the average attack success rate of applying the adversarial perturbation on all testing audio samples with varying delays. As we can see, AdvPulse can achieve a high attack success rate on the speaker recognition model: only 6 out of 90 combinations have average attack success rates under 90%, with the lowest attack success rate being 83.2%, and the overall average attack success rate is 96.9%. The average SNR of all adversarial perturbations is 8.3 dB.

Speech Command Recognition Model. Figure 5 shows the class-wise result of the attack on the speech command recognition model. Only 4 out of 90 combinations have attack success rates

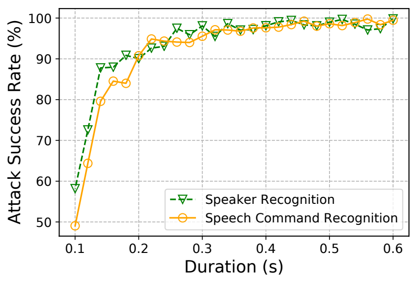


Figure 6: Attack success rate with different perturbation duration.

below 90%, with the lowest being 86.1%, which occurs when attempting to make all "stop" commands be mis-recognized as "on". The overall average attack success rate is 96.8%, and the average SNR of all adversarial perturbations from all actual-target class pairs is 13.7 dB.

These results show that our generated adversarial perturbation has good generality over unseen data and do not require synchronization, making it possible to attack streaming audio inputs (e.g., live human speech). Additionally, the calculated SNR shows that the generated adversarial perturbation is very much close to our situational environmental sound template, potentially concealing the perturbation in the environment from being noticed.

5.3 Impact of Perturbation Duration

The choice of the adversarial perturbation duration is crucial. Short durations make the attack more flexible and also less noticeable but decrease performance, while long durations guarantee high attack success rate but also place a stricter timing requirement. To study this trade-off impact, we evaluated the attack with different durations of the adversarial perturbation (0.1 to 0.6 seconds with a step of 0.02 seconds) for both speaker and speech command recognition models. For each specific duration, we selected 20 test samples from the same original class and applied iterative gradient descent to generate adversarial perturbation by solving Equation 5. We then added the perturbations with varying delays to the test samples. Figure 6 shows the average success rate with various perturbation durations. We can see that despite some random errors caused by SGD, a clear trend can be observed where the attack success rate increases drastically with the increase of the perturbation duration and achieves high performance (over 90%) if the duration is greater than 0.2 seconds. However, we conservatively chose to use 0.5 seconds as the default perturbation duration to guarantee a better performance in practical over-the-air scenarios.

6 EVALUATION OF PHYSICAL OVER-THE-AIR ATTACK

6.1 Experimental Methodology

Adversarial Perturbation Generation. We used the same attack configuration as our digital attack presented in Section 5.1. Unlike the digital attack that relies on Equation 5, to improve the robustness of adversarial perturbations for over-the-air attack, we generated subsecond (0.5 seconds) adversarial perturbations according to Equation 8. The average SNRs of adversarial perturbations generated for speaker and speech command recognition models are

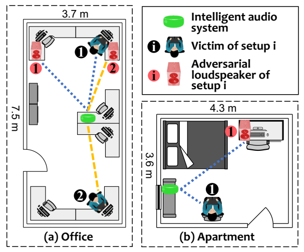


Figure 7: Illustration of office and apartment setups.

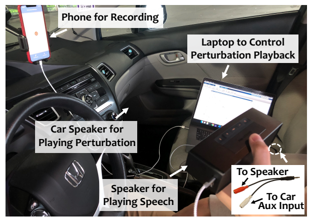

Phone for Recording
Laptop to Control Perturbation Playback
Car Speaker for Playing Perturbation
Speaker for Playing Speech
To Speaker
To Car Aux Input

Figure 8: Illustration of the inside-vehicle setup.

4.7 dB and 6.0 dB, respectively. It is worth noting that the adversarial perturbations were trained on several generic RIR datasets (i.e., the REVERB challenge database [29], the RWCP sound scene database [34], the Aachen impulse response database [26]) and a generic noise dataset (i.e., isotropic noise samples from the RWCP sound scene database [34]) instead of a well-suited one for each particular scenario. However, it's possible to utilize a customized RIR/noise dataset to further improve the attack performance under certain scenarios (e.g., using *engine noise* dataset to improve the performance in inside-vehicle scenarios).

Setup: Office and Apartment Scenarios. Figure 7 illustrates the environmental setup in two in-door scenarios: an office and a bedroom of an apartment. The office size is approximately 7.5 m × 3.7 m and the main noise sources are the fans of multiple desktop computers and AC. The apartment is about 4.3 m × 3.6 m and the main ambient noises are distant speeches from neighbors and bird singing outside of the window. The ambient noises of the office and the bed room were 41.0 dB_{SPL} and 38.8 dB_{SPL}, respectively, both of which were measured using a noise meter (i.e., RISEPRO decibel meter). The office scenario (Figure 7(a)) has two setups: In setup 1, the victim and the adversarial loudspeaker are located at two adjacent desks on the same side of the office, where the target intelligent audio system is placed at the center of the office. The distances between the victim and the adversarial loudspeaker to the intelligent audio system are 2.6 m and 2.7 m, respectively. Setup 2 depicts a scenario where the victim and the adversarial loudspeaker are located at two different sides of the room, and their distances to the intelligent audio system are 3.0 m and 2.6 m, respectively. In the apartment setup (Figure 7(b)), the intelligent audio system is placed on the shelf by the window, and the victim sits in a chair near the intelligent audio system while the adversarial loudspeaker is placed at the desk. The distances from the victim and the adversarial loudspeaker to the system are 1.6 m and 2.7 m, respectively. We used a TKGOU omnidirectional conference microphone as the receiving device of the intelligent audio system. For better control and repeatability, the victim (participant) was asked to hold a loudspeaker (i.e., Edifier R980T) for playing streaming speech (i.e., victim speaker's utterances or speech commands in our test dataset) and the adversarial loudspeaker (i.e., Edifier R980T) was used to play the generated adversarial perturbations with varying delays, attempting to attack the system. The utterances played by the loudspeaker held by the victim are not involved in the training and played in a streaming manner, which is sufficient to evaluate our attack in the streaming-speech scenario. We further demonstrated the feasibility of attacking live human speech on intelligent audio systems in Section 6.3.

Table 1: Attack success rates in indoor scenarios under different setups.

Setup	Speaker Recognition	Speech Command Recognition	Ambient Noise Level
Office (Setup 1)	82.7% (186/225)	91.1% (205/225)	41.0 dB_{SPL}
Office (Setup 2)	92.4% (208/225)	92.0% (207/225)	
Apartment	92.9% (209/225)	86.7% (195/225)	38.8 dB_{SPL}

Setup: Inside-vehicle Scenario. Figure 8 shows the setup for inside-vehicle scenario in a 2013 Honda Civic Sedan. The perturbation was played through car speakers and the speech was played through a loudspeaker (i.e., Bose SoundLink Mini II) held at the driver's speaking position using a Y-adapter. We used a smartphone (i.e., iPhone XS Max) as the receiving device of the intelligent audio system, and measured the ambient noise leveling using a noise meter (i.e., RISEPRO decibel meter). We evaluated our attack under three states: (1) *Engine Off*: the car was parked with the electric power on and the engine turned off, with an average ambient noise level of 35.9 dB_{SPL} (ranging from 34.0 to 37.5 dB_{SPL}); (2) *Engine On (Idling)*: the car was parked with the engine turned on, with an average ambient noise level of 42.8 dB_{SPL} (ranging from 41.9 to 44.2 dB_{SPL}); and (3) *Engine On (Cruising)*: the car was cursing with an average speed around 30 – 40 mph, and the main sources of noise are engine noise, wind noise, and road noise. The average ambient noise level was 64.7 dB_{SPL}, with the maximum noise level reaching 67.0 dB_{SPL}.

Evaluation Metrics. *(1) Attack Success Rate*: The same metric as described in Section 5.1. *(2) Baseline Accuracy*: The system accuracy of the Android implementation of the speech command recognition model used in Section 6.3 to evaluate live human attack. It is the ratio of correctly recognized speech commands over total number of speech commands the user spoke.

6.2 Attack Performance

Indoor Scenarios. Table 1 shows the results of the indoor scenario. For each model, we evaluated our attack in each setup by randomly choosing 5 pairs of actual-target classes. For each pair, we randomly selected 3 test audio samples from the actual class. During the playback of each test audio sample, we played the generated perturbation 15 times with uncontrolled varying delays. This gave us a total number of 225 recordings for each model in each setup. The perturbations were played at a similar volume of a regular phone notification sound (50 – 55 dB_{SPL}), which is much quieter comparing to speech (~ 60 dB_{SPL}). As we can see from Table 1, our attack can retain a high attack success rate in all the representative indoor setups, with an average attack success rate of 89.3% against speaker recognition systems and 89.9% against speech command recognition systems.

Inside-vehicle Scenario. Table 2 presents the experiment results of the inside-vehicle scenario. In each vehicle state, we randomly selected 5 pairs of the actual-target classes for each model. For each pair, we randomly selected 1 test audio sample from the actual class and played the generated perturbation 10 times with uncontrolled varying delays, resulting in a total number of 50 recordings for each model in each status. The perturbations were played through built-in speakers at the same level of loudness of a regular car radio (55 – 60 dB_{SPL}), which is quieter than road noises (~ 65 dB_{SPL}). As shown in Table 2, our attack can achieve a high attack success rate when the engine is turned off or idling (only failed 4 times against speaker recognition model and 1 time against speech command recognition model out of 100 trials). This shows that our attack is resilient to severe reverberation during playing the adversarial example in the narrow cabin. Even when the vehicle is cruising with a loud wind/road noise, our attack can still achieve a relatively high success rate (over 74% against both systems).

Table 2: Attack success rates in inside-vehicle scenario under different statuses.

State	Speaker Recognition	Speech Command Recognition	Ambient Noise Level
Engine Off	96% (48/50)	100% (50/50)	35.9 (34.0 – 37.5) dB_{SPL}
Engine On (Idling)	96% (48/50)	98% (49/50)	42.8 (41.9 – 44.2) dB_{SPL}
Engine On (Cruising)	74% (37/50)	78% (39/50)	64.7 (62.8 – 67.0) dB_{SPL}

Table 3: Attack success rates with different level of adversarial perturbation loudness.

Adversarial Perturbation Loudness (dB_{SPL})	45	50	55	60
Attack Success Rate	64% (32/50)	94% (47/50)	100% (50/50)	100% (50/50)

6.3 Attacking Live Human Speech

Due to the strict dependency on speaker identity and the lack of training data on the speaker recognition model, we only validated our attack against live human speech on the speech command recognition system. Specifically, we implemented an Android app of our target speech recognition model based on Tensorflow Lite and installed it on a smartphone (i.e., Google Pixel). As shown in Figure 9, we evaluated the live human attack in two scenarios: *(1) Laptop*: The victim speaks into the smartphone while the adversarial perturbation is played from a nearby laptop. This represents the scenario where the adversary remotely launches the attack by broadcasting adversarial perturbations through online media (e.g., Youtube Channel). *(2) Loudspeaker*: The victim speaks into the smartphone while the adversarial perturbation is played from a loudspeaker at a distance of 1 m. This is possible when the adversary launches the attack through an adversary-controlled loudspeaker (e.g., built-in speakers of smart appliances) placed in the vicinity.

We recruited 4 participants to play the role of victim, 2 being males (*p-1* & *p-4*) and 2 being females (*p-2* & *p-3*). Each participant was first asked to read out 3 arbitrary speech commands (10 times for each) without playing the adversarial perturbation, which serves as the baseline for the system. Then, each participant was asked to say the same set of speech commands again (10 times for each) in both scenarios with the adversarial perturbation playing in the background. Figure 10 depicts the calculated baseline accuracy and the attack success rate. As we can see, the system has a relatively high accuracy when facing benign inputs, with an average baseline accuracy of over 95%. Benefiting from the synchronization-free and audio-agnostic adversarial perturbation optimization process, our attack manged to reach an average accuracy of 98.7% (only failed 3 times out of a total number of 240 trails in these two scenarios).

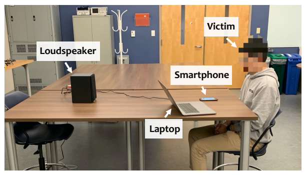


Figure 9: Illustration of the live human speech setup.

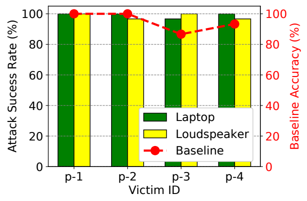


Figure 10: Results of attacking live human speech.

6.4 Impact of Perturbation Loudness

We used the loudspeaker setup as shown in Figure 9 to study the impact of adversarial perturbation loudness. The loudness of both the adversarial perturbation and human speech were measured using a noise meter (i.e., RISEPRO decibel meter) placed at the smartphone's position. We asked one participant (*p-4*) to read out 5 arbitrary speech commands, each for 10 times, while we played the adversarial perturbation in the background with different loudness by varying the volume of the loudspeaker. For reference, we measured the ambient noise level in the quiet room to be around $41 - 43$ dB_{SPL}, and the sound pressure level of the victim's voice to be around $60 - 62$ dB_{SPL}. As shown in Table 3, the attack success rate can reach 100% when the adversarial perturbation is played at 55 or 60 dB_{SPL}. The attack success rate decreases slightly when the volume is tuned down to 50 dB_{SPL}, and is further reduced when the adversarial perturbation is played at a very small volume (64% at 45 dB_{SPL} that is similar to the ambient noise). Considering the ultra-short duration of the adversarial perturbation and its high similarity with the situational environmental sound, our attack can be launched using a low volume (comparable to the ambient noise) to comprise systems without raising any suspicion. In addition to the perturbation loudness, we also studied the impact of attack distance with details presented in Appendix A.4.

7 DEFENSES

7.1 Performance Under Defense Settings

Some prior studies [14, 53, 54] have pointed out that applying audio pre-processing techniques before the recognition process to reduce the fidelity of the signal can potentially make the fragile adversarial perturbation lose its effectiveness. We evaluated our attack against four commonly-used defenses: *(1) Low-pass Filtering*: Apply a Butterworth low-pass filter with various cutoff frequencies; *(2) Quantization* (used in prior work [53]): Round the 16-bit signed integer amplitude values to its nearest integer multiple of q, where q is set to be $256, 512$ and 1024; *(3) MP3 Compression* (used in prior work [14]): Perform MP3 compression on the adversarial example prior to recognition; and *(4) Down-sampling* (used in prior work [53, 54]): Down-sample the adversarial example to a lower sampling rate (e.g., $2, 4$, and $8kHz$) and then perform signal recovery[3].

Specifically, we used a set of clean audio samples from each class to generate adversarial examples targeting the remaining 9 classes, by adding adversarial perturbations with random time delay. This results in a total number of 154 clean audio samples and $1,386$ adversarial examples for the speaker recognition system, and 200 clean audio samples and $1,800$ adversarial examples for the speech recognition system respectively. From Table 4, we can observe that these defense techniques barely impact the adversarial examples generated for the speech recognition system: the lowest attack success rate occurred when quantization with $q = 1024$ is applied, which is 96.8%, while the system accuracy on benign inputs is decreased to 74.0%. While for the speaker recognition system, when applying low-pass filtering and quantization, we observe that the system accuracy on benign inputs decreases faster than the attack success rate of the adversarial examples, rendering these defenses impractical. Although MP3 compression and down-sampling are shown to be relatively effective on the adversarial examples, they also significantly affect the recognition accuracy of benign inputs. This result demonstrated that simple audio pre-processing techniques either are ineffective or will largely degrade the system performance with normal audio inputs.

[3]The sampling rates of audio inputs used by the speaker recognition system and the speech command recognition system are $48kHz$ and $16kHz$, respectively.

7.2 More Advanced Adaptive Attacks

A sophisticated adversary can further design adaptive attacks against these input-transformation-based defenses by incorporating the corresponding transformation into the optimization process. For instance, to bypass low-pass-filtering-based defenses, the adversary can simply lower the frequency upper bound of the band-pass filter in Equation 6. For the transformations that are not smooth or not directly differentiable, the adversary can use backward pass differentiable approximation (BPDA) [5] to get approximated gradients. For example, to bypass MP3 compression, at each optimization step in Algorithm 1, the adversary can update the perturbation δ according to gradients calculated on $\mathcal{L}\big(f(MP3(x')), y_t\big)$.

8 DISCUSSION

8.1 Attacking ASR Systems with Long Inputs

Previously we evaluated our attack on speaker recognition system with input speech containing short sentences ($\sim$ 5 words) and speech command recognition system for recognizing single-word commands. In this subsection, we discuss the possibility of extending our attack to more sophisticated automatic speech recognition (ASR) systems with the ability to transcribe long sentences.

Temporal Dependency of Sequence-to-sequence Speech Recognition System. Speaker and speech command recognition models (our target systems) are essentially sequence-to-vector models, which usually utilize a temporal pooling layer to aggregate the extracted frame-level features across the time dimension so that a decision can be made with information collected from the entire speech. Differently, ASR systems leverage sequence-to-sequence models (e.g., recurrent neural network (RNN)) to transcribe a speech signal into corresponding text, where the output relies on not only the input at the current time step, but also the hidden state which encodes the representations learned from previous inputs. This structural difference should be taken into account when designing the attack against these two types of models. For instance, as shown in Figure 11, by injecting a short adversarial perturbation at arbitrary frames an adversary can impact the final recognition result of speaker recognition systems directly. While, for ASR systems, the influence of short perturbation injected at one frame is brought to its subsequent frames through the altered hidden states. Moreover, to take into account articular and linguistic dependencies, modern

Table 4: Attack performance in the presence of commonly-used defenses.

		No Defense	Low-pass Filtering			Quantization			Mp3 Compression	Down-sampling	
			2 kHz	4 kHz	8 kHz	256	512	1024		8 kHz	16 kHz
Speaker Recognition Systen	Accuracy on Benign Inputs	98.0%	51.9%	72.0%	93.5%	77.9%	43.5%	18.2%	52.6%	11.7%	29.9%
	Attack Success Rate	98.9%	80.8%	90.2%	94.6%	98.4%	96.0%	77.5%	26.3%	10.5%	21.0%
Speech Command Recognition System	Accuracy on Benign Inputs	87.0%	87.5%	87.5%	–	84.0%	83.0%	74.0%	88.0%	88.0%	–
	Attack Success Rate	98.0%	98.0%	98.0%	–	98.8%	98.7%	96.8%	98.0%	97.9%	–

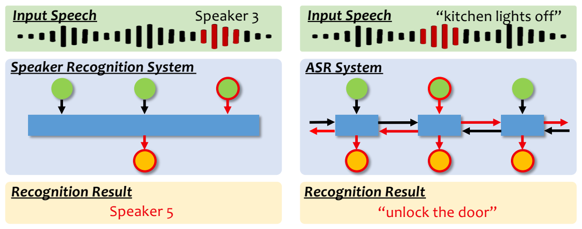
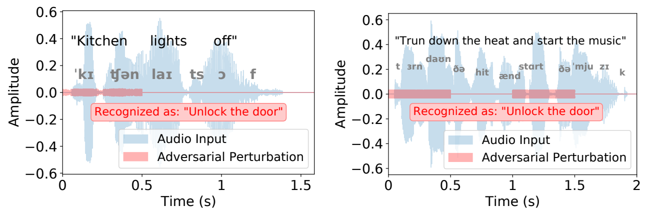


Figure 11: Structural difference between speaker recognition system and automatic speech recognition system. Areas marked red represent adversary-perturbed parts.



(a) Injecting single perturbation (b) Injecting multiple perturbations

Figure 12: Illustration of attacking ASR systems with short perturbations.

ASR systems utilize bidirectional models to correct the interpretation of the current phoneme by looking at both past and future phonemes, making it possible to add perturbation at one time step and change the transcription of the entire sequence.

Attacking Long Inputs. To verify the feasibility of attacking ASR systems with long input sentences, we used DeepSpeech [24] as the target system with connectionist temporal classification (CTC) loss [22] for our proof-of-concept experiments. Figure 12 (a) showcases a scenario where the added perturbation only covers the word "kitchen", but is able to influence the transcription of the subsequent words through changing hidden representations, resulting in the whole command to be falsely recognized as "unlock the door". During our experiments, we observed that it is hard for the perturbation added at one point to penetrate even longer audio inputs, with the effect decaying usually after $2-3$ words. However, according to Amazon[4], the mostly frequently used voice commands are usually only $2-4$ words long, making it sufficient to attack common voice commands in practice. For longer commands, attacks are still feasible by injecting multiple perturbation segments. For instance, in Figure 12 (b) we successfully made "turn down the heat and start the music" to be recognized as "unlock the door" by injecting two short perturbations. In our future work, we will explore the effect of hidden states and input conditions at varying time steps on the transcription results to further improve the robustness and synchronization freedom of the attack.

8.2 Practical Deployment

Different from existing studies, our proposed *AdvPusle* enables *man-in-the-middle attacks* on smart audio systems, where an adversary alters the user's streaming speech on the fly. This opens up many human-involved attack scenarios that are previously infeasible if having following restrictions: (1) The system requires the user to be present to enable its voice-interface (e.g., some systems may come equipped with sensor-based liveness detection schemes); (2) The process involves interactive audio inputs, making it hard for an adversary to foresee the conversational content and craft adversarial examples beforehand accordingly. For instance, interactive voice response (IVR) systems used for telephone banking require the user to interact with the system or a human representative while passing the speaker verification process; (3) The scenario requires to actively alter the recognition result of the victim's speech. For example, an adversary wants to make voice-controllable smart vehicles do the opposite as the driver's commands to potentially cause severe damage. The enabled properties by *AdvPusle* would be desirable to many more human-involved attack scenarios.

In our experiments, we chose to use bird singing and phone notification sound as the environmental sound template for illustration. In practice, the perturbation can also be disguised as continuous noises such as HVAC noise, car engine noise, or music, to be played periodically. Additionally, the user might notice something suspicious if the command is continued to be carried out incorrectly. However, in most cases, the malicious command may have already been executed and caused corresponding consequences (e.g., door has been opened; vehicle has been steered to a wrong direction).

9 CONCLUSION

In this work, we proposed *AdvPulse*, a practical adversarial audio attack against intelligent audio systems in the scenario where the system takes streaming audio inputs (e.g., live human speech). Unlike existing attacks that require the adversary to have prior knowledge of the entire audio input, we generated input-agnostic universal subsecond audio adversarial perturbations that can be injected anywhere in the streaming audio input. We also deliberately made it akin to environmental sounds to minimize suspicion while launching the attack. Additionally, various sources of audio distortions caused by physical playback are considered to improve the robustness of the perturbations during over-the-air propagation. Extensive experiments against both speaker and speech command recognition models under various realistic scenarios demonstrated the attack's effectiveness.

ACKNOWLEDGMENTS

We would like to thank our anonymous reviewers for their constructive feedback. This work was supported in part by National Science Foundation Grants CNS1820624, CNS1814590, and Air Force Research Laboratory Grant FA87501820058.

[4]https://www.amazon.com/alexa-skills/b?ie=UTF8&node=13727921011

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 265–283.

[2] Hadi Abdullah, Washington Garcia, Christian Peeters, Patrick Traynor, Kevin RB Butler, and Joseph Wilson. 2019. Practical hidden voice attacks against speech and speaker recognition systems. *arXiv preprint arXiv:1904.05734* (2019).

[3] Moustafa Alzantot, Bharathan Balaji, and Mani Srivastava. 2017. Did you hear that? adversarial examples against automatic speech recognition. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.

[4] Amazon. 2020. Amazon Echo. https://www.amazon.com/all-new-Echo/dp/B07R1CXKN7

[5] Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420* (2018).

[6] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2017. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397* (2017).

[7] Chase Bank. 2019. Security as unique as your voice. https://www.chase.com/personal/voice-biometrics.

[8] Karissa Bell. 2015. A smarter Siri learns to recognize the sound of your voice in iOS 9. https://mashable.com/2015/09/11/hey-siri-voice-recognition/

[9] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. 39–57.

[10] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. In *Proceedings of the IEEE Security and Privacy Workshops (SPW)*. 1–7.

[11] Guangke Chen, Sen Chen, Lingling Fan, Xiaoning Du, Zhe Zhao, Fu Song, and Yang Liu. 2019. Who is Real Bob? Adversarial Attacks on Speaker Recognition Systems. *arXiv preprint arXiv:1911.01840* (2019).

[12] Tao Chen, Longfei Shangguan, Zhenjiang Li, and Kyle Jamieson. 2020. Metamorph: Injecting Inaudible Commands into Over-the-air Voice Controlled Systems. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

[13] Moustapha M Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. 2017. Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. In *Proceedings of Advances in neural information processing systems (NeurIPS)*. 6977–6987.

[14] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Li Chen, Michael E Kounavis, and Duen Horng Chau. 2018. Adagio: Interactive experimentation with adversarial attack and defense for audio. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 677–681.

[15] Timothy Dozat. 2016. Incorporating nesterov momentum into adam. In *International Conference on Learning Representations (ICLR)*.

[16] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, Jul (2011), 2121–2159.

[17] Yuan Gong, Boyang Li, Christian Poellabauer, and Yiyu Shi. 2019. Real-time adversarial attacks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

[19] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[20] Google. 2020. Google Home. https://store.google.com/us/product/google_home

[21] Google. 2020. Speech-to-text Conversion Powered by Machine Learning. https://cloud.google.com/speech-to-text

[22] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*. 369–376.

[23] Chris Hall. 2019. Hey BMW: Your intelligent voice assistant is actually pretty good. https://www.pocket-lint.com/cars/news/bmw/148690-hey-bmw-your-intelligent-voice-assistant-is-actually-pretty-good.

[24] 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. *arXiv preprint arXiv:1412.5567* (2014).

[25] Xuedong Huang. 2017. Microsoft researchers achieve new conversational speech recognition milestone. https://www.microsoft.com/en-us/research/blog/microsoft-researchers-achieve-new-conversational-speech-recognition-milestone/

[26] Marco Jeub, Magnus Schafer, and Peter Vary. 2009. A binaural room impulse response database for the evaluation of dereverberation algorithms. In *International Conference on Digital Signal Processing*. IEEE, 1–5.

[27] Jack Kiefer, Jacob Wolfowitz, et al. 1952. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics* 23, 3 (1952), 462–466.

[28] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[29] Keisuke Kinoshita, Marc Delcroix, Takuya Yoshioka, Tomohiro Nakatani, Emanuel Habets, Reinhold Haeb-Umbach, Volker Leutnant, Armin Sehr, Walter Kellermann, Roland Maas, et al. 2013. The REVERB challenge: A common evaluation framework for dereverberation and recognition of reverberant speech. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. 1–4.

[30] Felix Kreuk, Yossi Adi, Moustapha Cisse, and Joseph Keshet. 2018. Fooling end-to-end speaker verification with adversarial examples. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 1962–1966.

[31] Zhuohang Li, Cong Shi, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. 2020. Practical Adversarial Attacks Against Speaker Recognition Systems. In *Proceedings of the 21st International Workshop on Mobile Computing Systems and Applications (HotMobile)*. 9–14.

[32] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. 2017. Universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1765–1773.

[33] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2574–2582.

[34] Satoshi Nakamura, Kazuo Hiyane, Futoshi Asano, Takanobu Nishiura, and Takeshi Yamada. 2000. Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition. In *Language Resources and Evaluation Conference*. 965–968.

[35] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. 2019. Universal adversarial perturbations for speech recognition systems. *arXiv preprint arXiv:1905.03828* (2019).

[36] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

[37] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. 2011. The Kaldi Speech Recognition Toolkit. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.

[38] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. 2019. Imperceptible, Robust, and Targeted Adversarial Examples for Automatic Speech Recognition. In *Proceedings of the International Conference on Machine Learning (ICLR)*. 5231–5240.

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

[40] Samsung. 2020. Unlocks your phone with Bixby Voice. https://www.samsung.com/us/support/answer/ANS00082783/

[41] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. 2018. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. *arXiv preprint arXiv:1808.05665* (2018).

[42] Google Assistant SDK. 2020. Best Practices for Audio. https://developers.google.com/assistant/sdk/guides/service/python/best-practices/audio.

[43] Suwon Shon, Hao Tang, and James Glass. 2018. Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model. In *IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 1007–1013.

[44] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. 2018. X-vectors: Robust dnn embeddings for speaker recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 5329–5333.

[45] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[46] Tesla. 2020. Voice Commands. https://www.tesla.com/support/voice-commands.

[47] Jon Vadillo and Roberto Santana. 2019. Universal adversarial examples in speech command classification. *arXiv preprint arXiv:1911.10182* (2019).

[48] Christophe Veaux, Junichi Yamagishi, Kirsten MacDonald, et al. 2017. CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit. *University of Edinburgh. The Centre for Speech Technology Research (CSTR)* (2017).

[49] Pete Warden. 2018. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209* (2018).

[50] Weidi Xie, Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. 2019. Utterance-level aggregation for speaker recognition in the wild. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5791–5795.

[51] Yi Xie, Cong Shi, Zhuohang Li, Jian Liu, Yingying Chen, and Bo Yuan. 2020. Real-time, Universal, and Robust Adversarial Attacks Against Speaker Recognition Systems. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[52] Hiromu Yakura and Jun Sakuma. 2018. Robust audio adversarial example for a physical attack. *arXiv preprint arXiv:1810.11793* (2018).

[53] Zhuolin Yang, Bo Li, Pin-Yu Chen, and Dawn Song. 2018. Characterizing audio adversarial examples using temporal dependency. *arXiv preprint arXiv:1809.10875* (2018).

[54] 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 Security 18)*. 49–64.

[55] Lei Zhang, Yan Meng, Jiahao Yu, Chong Xiang, Brandon Falk, and Haojin Zhu. 2020. Voiceprint Mimicry Attack Towards Speaker Verification System in Smart Home. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*.

[56] Yingke Zhu, Tom Ko, David Snyder, Brian Mak, and Daniel Povey. 2018. Self-Attentive Speaker Embeddings for Text-Independent Speaker Verification.. In *Interspeech*. 3573–3577.

A APPENDIX

A.1 Speaker Information

Table 5: Information of each speaker.

Speaker	ID	Age	Gender	Accents	Region
spk 0	225	23	F	English	Southern England
spk 1	226	22	M	English	Surrey
spk 2	227	38	M	English	Cumbria
spk 3	228	22	F	English	Southern England
spk 4	229	23	F	English	Southern England
spk 5	230	22	F	English	Stockton-on-tees
spk 6	231	23	F	English	Southern England
spk 7	232	23	M	English	Southern England
spk 8	233	23	F	English	Staffordshire
spk 9	234	22	F	Scottish	West Dumfries

A.2 Frequency Response

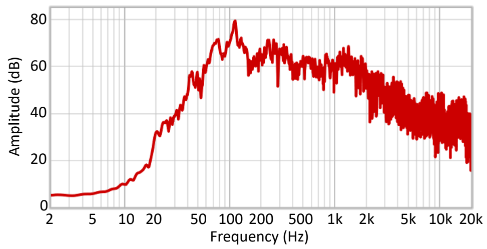


Figure 13: Measured frequency response.

A.3 Comparison of Distance Metrics

We performed a preliminary study to evaluate the quality of the generated adversarial perturbation in terms of the similarity with the environmental sound template using four distance metrics: *(1) Time Series*: the L2 distance between the two time-series signals; *(2) DCT*: the L2 distance between the discrete cosine transform of the two signals; *(3) STFT*: the L2 distance between the short-time Fourier transform of the two signals; and *(4) MFCC*: the L2 distance between the extracted Mel-frequency cepstral coefficients. We randomly initialized the audio perturbation (the random seed is set to be the same between trials) and performed a 2000 epoch gradient descent with a learning rate of 0.001 to minimize each distance. Figure 14 shows the spectrogram of the original environmental sound template (bird singing) and the generated audio perturbation using different distance metrics. As we can see, using Time Series, DCT and STFT can all effectively produce the adversarial perturbation that sounds similar to the environmental sound template, rendering the attack inconspicuous. To shorten the training time[5], we chose to use Time Series as the distance metric.

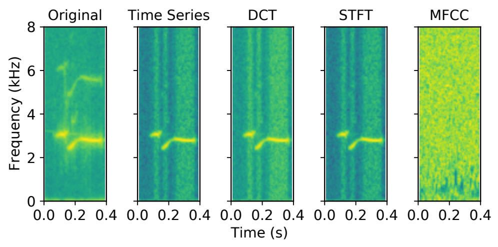


Figure 14: Spectrogram of the adversarial perturbations generated using different distance metrics to mimic environmental sound.

A.4 Impact of Attack Distance

We used the setup shown in Figure 15 to study the impact of the attack distance. As shown in the figure, one participant (*p-4*) is asked to sit at one end of the table with a smartphone (i.e., Google Pixel) and a noise meter (i.e., RISEPRO decibel meter) placed in front of him, while the loudspeaker is placed at different distances (i.e., $1 - 5$m) with a fixed volume. We asked the participant to first read out 5 arbitrary speech commands, 10 times each, without playing adversarial perturbation, to serve as the baseline system accuracy. Then the participant was asked to read out the same set of speech commands (10 times for each) while the perturbation was played from the loudspeaker placed at increasing distances with its loudness decreasing correspondingly. For reference, we measured the ambient noise to be around 46 dB_{SPL}, the speech of the participant to be around $74 - 78$ dB_{SPL}, and the baseline system accuracy to be 90%. Figure 16 shows the resulting attack success rate and perturbation loudness measured by the noise meter. We can see that our attack not only achieves a high attack success rate in the close-distance scenario (e.g., 100% at 1m and 1.5m), but also maintains a moderate accuracy under far-field settings with relatively strong reverberations (e.g., 68% at 4.5m and 62% at 5m) and low perturbation loudness that is comparable to the ambient noise (e.g., 48.4 dB_{SPL} at 5 m). This result demonstrated that by considering the effects of speaker and microphone limitation, absorption and reverberation, and ambient noise, the over-the-air optimization process can indeed provide robust adversarial perturbations.

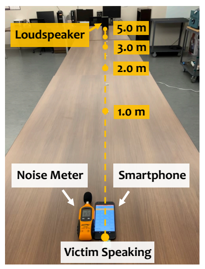


Figure 15: The distance study setup.

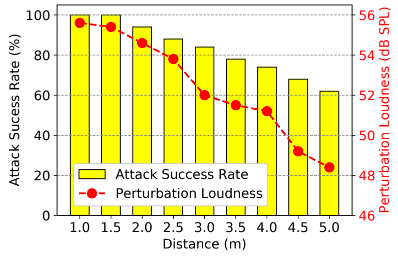


Figure 16: Attack performance at varying distances.

[5]The training time measured on a single NVIDIA GTX 2080Ti GPU: Time Series took 1.035s, DCT took 10.187s, STFT took 10.210s, and MFCC took 16.537s.