TOWARD PRIVATE, SECURE, AND ROBUST AI-ENABLED VOICE SERVICES

By

Hanqing Guo

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science—Doctor of Philosophy

2024

ABSTRACT

Voice, as a primary way for people to communicate with each other and interact with computers and smart devices, is expected to be trustworthy and reliable. For example, modern authentications use voice as a biometric to verify a user's identity; users give voice commands to control the smart devices via speech-to-text services. Compared to other biometrics such as iris, fingerprint, and face ID, voice biometrics show high usability because it does not require complicated hardware other than a microphone to support the authentication. Besides, the voice biometric can also be adapted for remote call authentication. Furthermore, voice serves as a crucial interface for humans to interact with smart devices, representing the most intuitive method for giving commands to artificial intelligence (AI) agents. The potential for smart devices and robots to comprehend human speech in the future holds great promise.

However, recent studies demonstrated the vulnerabilities of using voice interfaces to communicate, conduct speaker authentication, and deliver messages to smart devices. This dissertation aims to introduce the background of AI-enabled voice services; discover the vulnerabilities of modern voice models and systems; understand the root cause of the vulnerabilities; and provide security solutions to safeguard voice services.

First, we focus on speaker authentication security. Particularly, we propose a secure and robust speaker verification system called SuperVoice. By discovering the high-frequency energy in human speech, we find the special characteristics between different persons, and between humans and machines. Exploiting the high-frequency energy, the SuperVoice can enhance the performance of verified speakers and defend against machine-played attacks such as replay attacks, adversarial attacks, and inaudible attacks. Moreover, we propose a backdoor attack called MasterKey to attack speaker authentication systems. Compared to previous attacks, we focus on a real-world practical setting where the attacker possesses no knowledge of the intended victim.

Second, we explore the speech recognition security. Specifically, we design a new adversarial attack named SpecPatch to attack vulnerable speech recognition models. This attack alters the speech recognition model output by injecting a short, imperceptible noise-like sound. Compared to previous adversarial audio attacks, the SpecPatch shows strong resistance under different types of distortions and is able to succeed even when the user is present. Furthermore, we propose PhantomSound, a query-efficient black-box attack toward commercial speech recognition services/APIs/voice assistants. Different from existing black-box adversarial attacks on voice assistants, PhantomSound leverages the decision-based attack to produce effective adversarial audios and reduces the number of queries by optimizing the gradient estimation. We demonstrate the danger of PhantomSound on commercial speech recognition services and off-the-shelf smart voice assistants.

Third, we investigate the voice privacy protection. To address the privacy leakage issue of voice communication, we create a system, called NEC, that uses an AI model to selectively jam the user's voice from an unauthorized recorder. The NEC transmits speaker-specified noise via an inaudible channel to jam the only user's sound. We successfully implemented the NEC, and demonstrated that NEC can protect the user's sound from being recorded.

This dissertation comprehensively addresses the prevalent challenges and vulnerabilities in voice-enabled services. In an age where voice-enabled devices are becoming ubiquitous in homes and public spaces, ensuring the security of these devices is paramount. Our research helps in safeguarding the privacy and safety of the general public, who are often the targets of security breaches. In conclusion, our comprehensive analysis and proactive solutions to the challenges in AI-enabled voice interaction systems represent a leap forward. We offer a security perspective in a field that is critical to the technological advancement of our society. Our contributions may lay the groundwork for safer, more secure voice AI interactions, benefiting both the security community and society as a whole.

To Nan, and Leo for their love and support.

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude toward my advisor Dr. Li Xiao. I am grateful for the opportunity she gave me to join her lab from the beginning, for her patience with me, and for the courage she gave me when I encountered difficulties in my research. I am even more thankful for her selfless help to my family during the COVID period. She is undoubtedly a great mentor, not only in academics and life but also in guiding me on the path of life.

I would also like to thank my advisor, Dr. Qiben Yan. I am grateful for his guidance, correction, and instruction on my research direction and methodology. I have learned from him how to build up my research taste and research vision. He taught me how to be professional, responsible, and creative during my PhD studies. I appreciate his patience in mentoring me, his tolerance for my mistakes, and his support and backup behind the screen for every paper submission. I am deeply grateful for his guidance and will carry his attitude towards research into my future work. Thank you, Dr. Xiao and Dr. Yan, for believing in me and empowering me to believe in myself!

I am thankful to Dr. Zhichao Cao for serving in my Qualifying and Guidance committees, offering multiple opportunities for academic growth, and helping me with faculty job applications. I am thankful to Dr. Eric Hunter for his service in my Guidance committee and for thoughtful feedback from his speech expertise on my research. I am thankful to my past and present colleagues from eLANS Lab and SEIT Lab — Dr. Nikolay Ivanov, Bocheng Chen, Guangjing Wang, Yuanda Wang, Jianzhi Lou, Dr. Lingkun Li, Dr. Li Liu, Dr. Xiao Zhang, James Mariani, Kanishka Wijewardena, Griffin Klevering, Yiwen Hu, Yujin Ki.

I am thankful to the external research collaborators and mentors that I had the honor to work with: Dr. Yunhao Liu, Dr. Jiliang Wang, Shuai Tong (Tsinghua University), Chenning Li (Massachusetts Institute of Technology), Dr. Junfeng Guo (The University of Maryland), Dr. Xun Chen (Samsung Research America), Dr. Ning Zhang (Washington University in St. Louis), Dr. Mi Zhang (Ohio State University), Dr. Qiaona Hu, Dr. Gabriel Passamani Andrade, Dr. Xiao Zeng (Amazon). I am grateful to many faculty members from Michigan State University for helping me navigate academia: Dr. Kristen Johnson, Dr. Sandeep Kulkarni, Dr. Manni Liu, Dr. Philip

McKinley, Dr. Joshua Nahum, Dr. Yiying Tong, Dr. Eric Torng, Dr. Guan-Hua Tu, Dr. Huacheng Zeng, and Dr. Jiayu Zhou.

Most importantly, I am thankful to my wife Nan Zhang, my son Leo, my mother Hongxia, and my father Hongjian — for their love, support, patience, and believing in me.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Voice is a primary way for people to interact with friends, computers, and smart devices. People talk to each other for information exchange; users talk to AI agents or smart devices to control them via speech; and the cloud services use the voice trait to authenticate their users. With the prevalent usage of voice in modern communication, the security research community has started to worry about the safety of using voice in multiple scenarios. For example, the adversary can bypass the speaker authentication system and then access the victim's personal information and manipulate the system (mobile phone, customer account, or smart speaker). In the other case, the attacker can breach the speech recognition model and therefore mislead the smart speakers or speech-to-text services to produce harmful output (e.g., control the smart speaker to open the door; or generate toxic response from speech understanding services). In terms of the privacy issue in voice, the attacker can play the role of an eavesdropper to steal the victim's sound in public areas, which follows the voice cloning techniques to launch a speech synthesis attack. Figure 1.1 depicts the research focuses. On the left, it shows that modern AI-enabled voice services provide functionalities such as speaker authentication and speech recognition to a smart agent (robot) for boosting the secure and efficient communication between humans and smart devices. On the right, the adversary may break the functionality with different attacks: impersonating the legitimate user; giving malicious speech commands to control the smart agent; and eavesdropping the secret conversation to expose the privacy of the victim. The goal of my research is twofold. First, we provide security solutions to safeguard the functionalities of model voice services; Second, we discover the potential threat to the existing service to alert the user and product manufacturer. We focus on three major voice-related tasks: speaker authentication, speech recognition, and privacy protection. In the rest of this chapter, I will introduce the research background, and my contributions, as well as the organization of the dissertation.

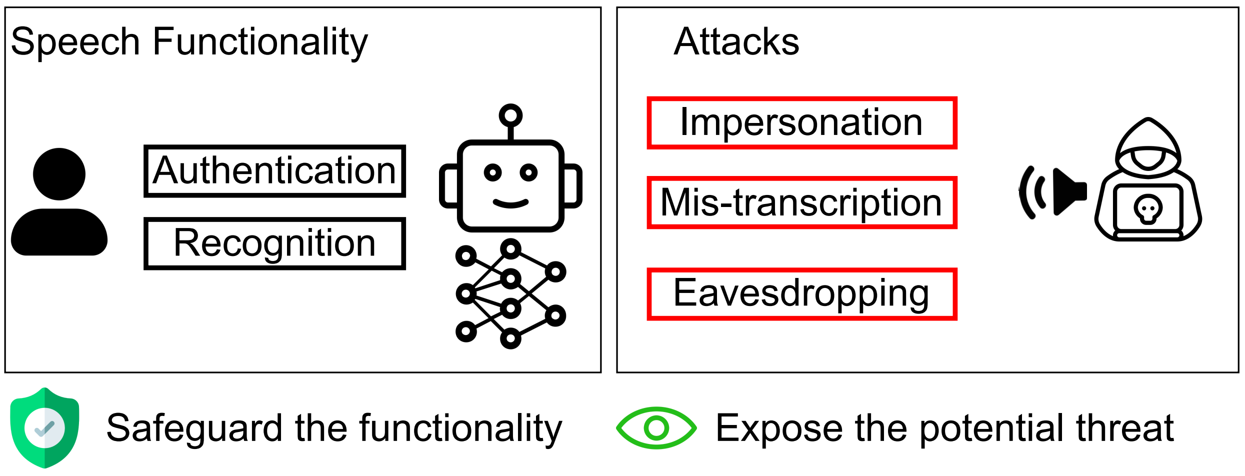


Figure 1.1: Overview

1.1: Research Background

1.1.1: Speaker Authentication

Modern devices increasingly adopt biometric technologies for user authentication. Among various types of human biometrics, such as fingerprint, facial, iris, etc., voice biometrics demonstrate great benefits in its high usability, convenience, and security. *Speaker Verification (SV)* systems commonly use voice biometrics to automatically accept or reject a voice input based on the speaker models stored on the smart devices or cloud. (This dissertation treats speaker verification and speaker authentication as interchangeable terms.) Nowadays, all the popular voice assistants, such as Siri, Alexa, and Google Assistant, have integrated SV algorithms for certain wake words (e.g., “Hey, Siri”, “Ok, Google”). A more appealing approach, called *text-independent speaker verification*, could accurately and efficiently verify arbitrary utterances from a target speaker based on a limited set of enrolled sentences. Recently, security researchers have demonstrated the susceptibility of SV systems to voice mimicry attacks and replay attacks, where the attackers imitate/synthesize victims’ voices or record/replay them to bypass the SV systems [10, 102, 110, 192]. As the number of sensitive applications (e.g., banking [171]) of voice assistants is growing, practical SV systems aim to achieve not only high accuracy in text-independent speaker verification but also high efficacy in defending spoofing attacks under a limited time budget.

Limitations: Existing speaker authentication models suffered from either low accuracy, high computational cost, or vulnerability to adversarial attacks such as re-edited speech or AI-synthesized speech. Our research contributes to revealing the vulnerability of the SV system, and proposes new efficient and effective speaker authentication approaches.

1.1.2: Speech Recognition

Speech is a major interface for humans to communicate with an intelligent agent. Voice communication is a human-computer interaction approach that enables hands-free operation and offers opportunities for visually impaired users. Recently, with the thriving development of Artificial Intelligence (AI) and deep learning models, the performance of Automatic Speech Recognition (ASR) has improved significantly, resulting in a growing product market. For example, tech companies developed their online ASR systems and provided those services to the public, including Amazon Transcribe [14], Google Cloud Speech-to-Text [63], IBM Watson Speech to Text [97], and Microsoft Azure Speech Service [125]. Furthermore, they also integrated their ASR APIs into the Intelligent Voice Control (IVC) devices to offer voice assistant services (e.g., Siri [164], Google Assistant [61], or smart speaker systems such as Google Home [62] and Amazon Echo [13]). Besides that, more and more companies deliver their customer service using intelligent voice systems, which are empowered by ASR models to understand customers' questions and improve the efficiency of customer support. However, with the increasing presence of ASR systems and IVC devices in private spaces, people have started to worry about the security and privacy of these systems. For example, a hacked device is now capable of recording private conversations; collecting and sharing private data; and controlling all the connected IoT devices in smart homes [37, 158]. Researchers have demonstrated that ASR systems could become vulnerable to a wide variety of attacks. For instance, inaudible commands can be injected through ultrasound [141,210], even across different transmission media, such as object surface [200], light [158], etc. Besides the physical attacks, recent studies also utilize the discrepancies between the human ear and feature extraction algorithms to launch *signal processing attacks* [4, 5]. Despite the aggravating threats, these new attacks could be defeated by integrating additional hardware [209] or extra signal processing procedures (e.g., voice activity detection, guard signals) [4, 90]. Unlike the aforementioned attacks, the *adversarial attack* aims to attack the deep neural networks (DNN), i.e., the computational core of an ASR system, which poses a major threat to modern ASR systems.

Limitations: Existing speech recognition services are vulnerable to inaudible commands such as ultrasound, and imperceptible attacks such as adversarial music, human insensitive transformed signals, or backdoor commands. We investigate the limitations of the existing attacks and expose new threats (such as robust adversarial patch attacks and query-efficient black-box attacks) to the safety of speech recognition systems.

1.1.3: Privacy Protection

Voice recording is an essential information-sharing approach, which is benefiting many aspects of our daily lives. Nowadays, smartphones and Internet-of-Things (IoT) devices equipped with microphones allow people to record voices anytime and anywhere. However, the growing presence of unauthorized microphones has led to numerous incidences of privacy violations. Off-the-shelf microphones are widely available and can be deployed to steal users' biometric traits (e.g. voiceprints) or private conversations. Thus, unauthorized voice recording has become a serious societal issue [116]. For example, the adversary can record private conversations for personal usage and cause privacy leakage. Moreover, the adversary can conduct the speaker conversion attack [48, 155] to produce more speech samples as the recorded victim. Besides, the adversary can separate the recorded speech as multiple clips, and perform speech synthesis attack [150]. Most recently, the unauthorized recording can be further used for attacking the speaker verification models, such as replay attack [110, 192], adversarial attack [31]. Thus, preventing unauthorized recording is a critical research problem to enable the security of voice communication.

Limitations: Piror effort to defend against unauthorized recording with speech jammer, a device that continuously generates ambient noise by ultrasound. However, this approach leaves two critical safety concerns: First, all the surrounding microphones will be affected, leading to deny of service for unrelated users. Second, the ambient noise is usually generated with a specific noise pattern, therefore, the eavesdropper can easily recover with the victim's speech. Our study contributes to the design of a new system that achieves speaker-specified jamming that prevents unauthorized recording without affecting others.

1.2: Contribution of This Dissertation

1.2.1: Overview of This dissertation

This dissertation encompasses five of my publications, which focus on speaker authentication, recognition, and privacy concerns. Each paper's research emphasis, particularly in the context of attack and defense strategies, is concisely summarized. Table 1.1 provides a comprehensive overview of these publications, detailing their specific research topics. In the realm of speaker authentication, our significant contribution is the development of SuperVoice, a robust speaker verification system. Additionally, we identify a major vulnerability in speaker verification models and demonstrate the feasibility of large-scale backdoor attacks, a technique we refer to as MasterKey. In the field of speech recognition, our research introduces two innovative attack methodologies designed to deceive speech-to-text models. The first, SpecPatch, is tailored for scenarios involving human interaction, while the second, PhantomSound, is optimized for black box environments. Addressing privacy issues, we present NEC, an intelligent jamming device that effectively protects against unauthorized voice recordings.

1.2.2: Contribution to Speaker Authentication Security

SuperVoice: We propose SuperVoice, a speaker verification system that provides secured speaker authentication by leveraging ultrasound features in human speech. Compared to existing speaker verification techniques which distinguish individual speakers via the spectrographic features extracted from an audible frequency range of voice commands, we explore a new direction of human voice research by scrutinizing the unique characteristics of human speech at the ultrasound fre-

Table 1.1: Overview of the scope of this dissertation.

	Chapter	Related Publication[†]	Research Focus ATTACK	Research Focus DEFENSE
AUTHENTICATION	Chapter 2	**H. Guo**, Q Yan, N. Ivanov, Y. Zhu, L. Xiao, EJ. Hunter *SuperVoice: Text-Independent Speaker Verification Using Ultrasound Energy in Human Speech [73]* ACM ASIA CCS 2022	○	●
AUTHENTICATION	Chapter 3	**H. Guo**, X. Chen, J. Guo, L. Xiao, Q. Yan *MASTERKEY: Practical Backdoor Attack Against Speaker Verification Systems [69]* ACM MobiCom 2023	●	○
RECOGNITION	Chapter 4	**H. Guo**, Y. Wang, N. Ivanov, L. Xiao, Q. Yan *SpecPatch: Human-In-The-Loop Adversarial Audio Spectrogram Patch Attack on Speech Recognition [72]* ACM CCS 2022	●	○
RECOGNITION	Chapter 5	**H. Guo**, G. Wang, Y. Wang, B. Chen, Q. Yan, L. Xiao *PhantomSound: Black-Box, Query-Efficient Audio Adversarial Attack via Split-Second Phoneme Injection [71]* RAID 2023	●	○
PRIVACY	Chapter 6	**H. Guo***, C. Li*, L. Li, Z. Cao, Q. Yan, L. Xiao *NEC: Speaker Selective Cancellation via Neural Enhanced Ultrasound Shadowing [70]*[†] IEEE DSN 2022	○	●

● — primaty focus ○ — not addressed.

[†] The author of this dissertation (in bold), is the *main contributor* to all these papers, * indicate the equal contribution.

quency band. Our research indicates that the high-frequency ultrasound components (e.g. speech fricatives) from 20 to 48 kHz can significantly enhance the security and accuracy of speaker verification. Our SuperVoice system uses a two-stream DNN architecture with a feature fusion mechanism to generate distinctive speaker models. To test the system, we create a speech dataset with 12 hours of audio (8,950 voice samples) from 127 participants. In addition, we create a second spoofed voice dataset to evaluate its security. To balance between controlled recordings and real-world applications, the audio recordings are collected from two quiet rooms by 8 different recording devices, including 7 smartphones and an ultrasound microphone. Our evaluation shows that SuperVoice achieves 0.58% equal error rate in the speaker verification task, which reduces the best equal error rate of the existing systems by 86.1%. SuperVoice only takes 120 ms to test an incoming utterance, outperforming all existing speaker verification systems. Moreover, within 91 ms processing time, SuperVoice achieves 0% equal error rate in detecting replay attacks launched by 5 different loudspeakers. Finally, we demonstrate that SuperVoice can be used in retail smartphones by integrating an off-the-shelf ultrasound microphone.

MasterKey: We propose a new threat toward the speaker authentication system. The attack, called MasterKey, is a backdoor attack to compromise the many SV models. Different from previous attacks, we focus on a real-world practical setting where the attacker possesses no knowledge of the intended victim. To design MasterKey, we investigate the limitation of existing poisoning attacks against unseen targets. Then, we optimize a universal backdoor that is capable of attacking arbitrary targets. Next, we embed the speaker's characteristics and semantics information into the backdoor, making it imperceptible. Finally, we estimate the channel distortion and integrate it into the backdoor. We validate our attack on 6 popular SV models. Specifically, we poison a total of 53 models and use our trigger to attack 16,430 enrolled speakers, composed of 310 target speakers enrolled in 53 poisoned models. Our attack achieves a 100% attack success rate with a 15% poison rate. By decreasing the poison rate to 3%, the attack success rate remains around 50%. We validate our attack in 3 real-world scenarios, and successfully demonstrate the attack through both over-the-air and over-the-telephony-line scenarios.

1.2.3: Contribution to Speech Recognition Security

SpecPatch: We propose SpecPatch. The first human-in-the-loop adversarial audio attack on automated speech recognition (ASR) systems. Existing audio adversarial attacker assumes that the users cannot notice the adversarial audio, and hence allow the successful delivery of the crafted adversarial examples or perturbations. However, in a practical attack scenario, the users of intelligent voice-controlled systems (e.g., smartwatches, smart speakers, smartphones) have constant vigilance for suspicious voice, especially when they are delivering their voice commands. Once the user is alerted by suspicious audio, they intend to correct the falsely-recognized commands by interrupting the adversarial audio and giving more powerful voice commands to overshadow the malicious voice. This makes the existing attacks ineffective in the typical scenario when the user's interaction and the delivery of adversarial audio coincide. To truly enable the imperceptible and robust adversarial attack and handle the possible arrival of user interruption, we design SpecPatch, a practical voice attack that uses a sub-second audio patch signal to deliver an attack command and utilize periodical noises to break down the communication between the user and ASR systems. We analyze the CTC (Connectionist Temporal Classification) loss forwarding and backwarding process and exploit the weakness of CTC to achieve our attack goal. Compared with the existing attacks, we extend the attack impact length (i.e., the length of attack target command) by 287%. Furthermore, we show that our attack achieves 100% success rate in both over-the-line and over-the-air scenarios amid user intervention.

PhantomSound: Compared to the SpecPatch, which only works for white-box setting, we propose PhantomSound, a query-efficient black-box attack toward voice assistants. Existing black-box adversarial attacks on voice assistants either apply substitution models or leverage the intermediate model output to estimate the gradients for crafting adversarial audio samples. However, these attack approaches require a significant amount of queries with a lengthy training stage. PhantomSound leverages the decision-based attack to produce effective adversarial audios, and reduces the number of queries by optimizing the gradient estimation. In the experiments, we perform our attack against 4 different speech-to-text APIs under 3 real-world scenarios to demonstrate the real-time attack impact. The results show that PhantomSound is practical and robust in attacking 5 popular commercial voice controllable devices over the air, and can bypass 3 liveness detection mechanisms with $> 95\%$ success rate. The benchmark result shows that PhantomSound can generate adversarial examples and launch the attack in a few minutes. We significantly reduce the number of queries by by 93.1% (untargeted) and 65.5% (targeted) compared with the state-of-the-art black-box attacks.

1.2.4: Contribution to Speech Privacy Protection

NEC: To safeguard the privacy leakage of daily conversation, we propose NEC (Neural Enhanced Cancellation), a defense mechanism that prevents unauthorized microphones from capturing a target speaker's voice. Compared with the existing scrambling-based audio cancellation approaches, NEC can selectively remove a target speaker's voice from a mixed speech without causing interference to others. Specifically, for a target speaker, we design a Deep Neural Network (DNN) model to extract high-level speaker-specific but utterance-independent vocal features from his/her reference audios. When the microphone is recording, the DNN generates a shadow sound to cancel the target voice in real-time. Moreover, we modulate the audible shadow sound onto an ultrasound frequency, making it inaudible for humans. By leveraging the nonlinearity of the microphone circuit, the microphone can accurately decode the shadow sound for target voice cancellation. We implement and evaluate NEC comprehensively with 8 smartphone microphones in different settings. The results show that NEC effectively mutes the target speaker at a microphone without interfering with other users' normal conversations.

1.3: Organization

This dissertation is organized as follows. Chapter 1 presents the research background and the overview of this dissertation. Chapter 2 introduces a speaker authentication system that is secured by ultrasound. Chapter 3 elaborates on our new attack against large-scale and long-distance speaker authentication systems. Chapter 4 discovers the vulnerability of speech recognition in human-in-the-loop scenarios. Chapter 5 introduces a query-efficient black-box attack against commercial speech-to-text services/APIs. Chapter 6 addresses the privacy leakage problem by proposing a speaker-specified recording jammer. Chapter 7 summarizes this dissertation and outlines future directions.

CHAPTER 2: SPEAKER VERIFICATION USING ULTRASOUND ENERGY IN HUMAN SPEECH[1]

2.1: Introduction

Modern devices are increasingly using voice biometrics due to its usability and security. Speaker Verification (SV) systems, used in voice assistants like Siri, Alexa, and Google Assistant, rely on voice biometrics to recognize specific wake words and verify users. Recent SV studies have explored the distinctive vocal or non-vocal features such as phoneme position [214], cumulative spectrum [7], mouth motion [124, 213], body vibration [54], and sound field [198]. Based on these features, conventional machine learning models have been used to generate speaker models, including correlation (CORR), support vector machine (SVM), Gaussian mixture models (GMM), etc. Meanwhile, deep neural network (DNN) based SV systems use robust neural networks for building speaker models with the prototypical features (e.g., waveform [92, 137, 143], spectrogram [91, 128, 177], and MFCC (Mel-Frequency Cepstral Coefficients) [165]). As summarized in Table 2.1, most of the existing SV systems cannot simultaneously achieve effective speaker verification and defense against spoofing attacks [7, 124, 213, 214], while others have limitations in their usability, e.g., with the requirement of wearing extra devices [54], staying at the same positions as the enrollment phase [198], etc. Moreover, their discovered vocal or non-vocal features cannot be *transferred* across different speaker models. Although existing DNN-based SV systems [91, 128, 137, 165, 177] do not deal with rigid features, they tend to yield relatively high error rates due to the lack of speaker representative features.

[1]This chapter is based on previously published work by Hanqing Guo, Qiben Yan, Nikolay Ivanov, Ying Zhu, Li Xiao and Eric J. Hunter titled "SuperVoice: Text-Independent Speaker Verification Using Ultrasound Energy in Human Speech" published at the Proceedings of the 2022 ACM Asia Conference on Computer and Communications Security. DOI: 10.1145/3488932.3517420 [73].

Motivation: In this research, we aim to explore ultrasound energy in human speech to enhance the accuracy and security of text-independent speaker verification. More specifically, we investigate the unique properties of the *human speech in the human-inaudible ultrasound frequency band* (i.e., frequencies greater than 20 kHz). High-frequency ultrasound components in human speech present several unique properties: first, they are imperceptible by humans but can be captured by an ultrasound microphone; second, individual human speakers can produce ultrasound waves with distinct characteristics, determined by the unique shape of the speech production system and the particular use (e.g. timing) of the system. Recent attacks towards voice assistants, such as DolphinAttack [210] and SurfingAttack [200], leveraged the inaudible ultrasound signals to inject commands into voice assistants. Here, we take a *reversed* approach: rather than utilizing the ultrasound signals for attack, we take advantage of the unique properties of the high-frequency audio spectrum for defense, to offer a more robust and accurate SV system.

We propose SuperVoice, a robust and secure text-independent SV system, which applies to commodity mobile devices equipped with an ultrasound microphone. SuperVoice analyzes an incoming voice command to the ultrasound microphone. The audio includes both the audible (below 20 kHz) and ultrasound (above 20 kHz) frequency components. SuperVoice then processes these components to extract both the low-frequency and high-frequency feature representations using a liveness detection module and a *two-stream DNN architecture*. These features are fused to a second-level classifier to generate or match a speaker embedding for speaker verification purposes.

Challenges: The design of SuperVoice faces 3 critical challenges. *i) How to ascertain that the ultrasound feature can represent the speaker's voiceprint?* Prior acoustic studies show evidence that high-frequency energy (from 8-16 kHz) contains useful features to identify an individual speaker [86, 106, 145]. However, none of them focuses on the ultrasound frequency band above 20 kHz. The existing feature engineering techniques such as LPCC (Linear Prediction Cepstral Coefficients), Filter banks, and MFCC cannot be directly applied in high-frequency data, as they are designed for narrowband speech data (below 8 kHz). To better utilize the ultrasound features, we design signal processing techniques to extract the unique characteristics from the ultrasound components.

Table 2.1: SuperVoice in comparison with other SV systems.

System	Feature	Model	Text Indep.	Security	Transfer
VoiceLive [214]	Phoneme	CORR	✗	✓	✗
VoiceGes. [213]	Mouth	CORR	✗	✓	✗
WiVo [124]	Mouth	CORR	✗	✓	✗
VAuth [54]	Body	CORR	✓	✓	✗
Void [7]	Cum. Spec	SVM	✓	✓	✗
CaField [198]	Sound field	GMM	✓	✓	✗
TE2E [91]	Spectrum	CNN	✓	✗	✓
GE2E [177]	Spectrum	CNN	✓	✗	✓
Siri [165]	MFCC	RNN	✓	✗	✓
SincNet [137]	Waveform	CNN	✓	✗	✓
VGGVox [128]	Spectrum	CNN	✓	✗	✓
SuperVoice	**Ultrasound**	**CNN**	✓	✓	✓

ii) How to use the ultrasound features to detect replay attacks that involve multiple playback devices? Since the attackers can use different devices (e.g., smartphones, ultrasonic microphones, and ultrasonic speakers) to record and replay the voice signals, it is challenging to design a liveness detection method to cope with different attack devices with varied signal characteristics. *iii) How to design a neural network structure to integrate the ultrasound features?* Since the speech production theory of low-frequency features and high-frequency features are very different, the integration of both features is particularly challenging. We design a two-stream DNN structure with convolutional filters to process and integrate the ultrasound features.

Contributions: To the best of our knowledge, we are *the first* to prove that *ultrasound components (20 $\sim$ 48 kHz)* in human speech can be used to enhance the accuracy, robustness, and security of the SV systems. We demonstrate that the ultrasound components are model-agnostic by integrating them into multiple SV models, all of which achieve enhanced performance in the SV tasks. Surprisingly, the ultrasound components in human speech have been largely neglected prior to this work [103]. In summary, this chapter makes the following contributions:

- We demonstrate that human speech does include ultrasound components, and those components can help distinguish among different human speakers. Moreover, the ultrasound components in speech signals can be utilized to identify spoofing attacks by measuring the signals' cumulative energy levels at the high-frequency range.

- We design SuperVoice, a speaker verification and spoofing detection system. By incorporating a two-stream neural network structure with time-frequency spectrogram filters and feature fusion mechanism, SuperVoice achieves high accuracy in text-independent speaker verification.
- We launch a human voice study and collect two datasets for speaker verification and spoofing detection. We recruit 127 participants and record 8,950 audios by 8 different smart devices, including 7 smartphones and an ultrasound microphone. We also replay 500 audio samples to construct a spoofed voice dataset with 5 playback devices. In total, our datasets involve 127 participants with a total of 12 hours audios. We make our datasets publicly available at **https://supervoiceapp.github.io/**.
- We evaluate the performance of SuperVoice and compare it against other SV systems. The result shows that SuperVoice achieves 0.58% equal error rate (EER) in speaker verification, which improves the EER of the top-performing SV system by 86.1%. Remarkably, it only takes 120 ms to test an incoming utterance. Moreover, SuperVoice achieves 0% EER with 91 ms processing time for liveness detection, outperforming all existing approaches. The two-week longevity experiment demonstrates that SuperVoice is suitable for long-term use, and its performance is barely impacted by the changes in distances and angles.

2.2: Background

2.2.1: Threat Model

We consider *voice spoofing attack*, which is a malicious attempt to impersonate a genuine speaker to execute an unauthorized command on a voice assistant. The three most popular types of voice spoofing attacks include *replay*, *synthesis*, and *conversion* [192]. In a replay attack, the adversary records the legitimate command uttered by a genuine speaker and replays this command later. The synthesis attack uses the text-to-speech (TTS) generation to create artificial voice commands acceptable by a voice assistant. The conversion attack converts an existing voice command into a different one that can bypass speaker verification. To provide effective countermeasures against voice spoofing attacks, this research aims to develop an end-to-end SV system that can perform both liveness detection and speaker verification.

2.2.2: Can Humans Produce Ultrasound?

The sounds in human speech are commonly divided into vowels and consonants: the vowels are produced when the air steadily flows through the vocal tract above the larynx (see Figure 2.1), while the consonants are more transient in nature and are produced when the flow of air is partially restricted or completely stopped at the vocal fold. The consonants are characterized by *voicing*, *place of articulation*, and *manner of articulation* [39, 101]. Voicing refers to the inclusion of vocal fold vibration as a sound source which is quasi-steady and generally harmonic in nature, and the place of articulation represents the location of constriction in the vocal tract which usually results in a highly transient noise. The manner of articulation describes how a sound is altered by the manipulation of airstream flows from the lungs. More specifically, when two speech organs narrow the airstream to cause friction to occur as it passes through, *Fricatives* are produced. If the airstream is stopped and then released, *Stop* or *Affricate* is produced.

Particularly, the Stop, Affricate, and Fricative consonants are known to exhibit *high frequency energy (HFE)*, since the airstream is deformed by articulations. In this work, *we aim to scrutinize this under-explored and largely neglected phenomenon in human speech, i.e., the consonants carry high energy in the human-inaudible ultrasound frequency range*. We perform experiments to validate that a human speech production generates energy in the ultrasound spectrum during a normal utterance, primarily within speech components such as the Stop, Affricate, and Fricatives. Figure 2.2a shows the human voice frequency spectra sensed by an ultrasound microphone, in which a significant portion of the acoustic energy is observed beyond 20 kHz. In this study, we are the *first* to show that the acoustic energy beyond 20 kHz (i.e., ultrasound voice components) plays an important role in the speaker verification task and offers an effective and economical solution for liveness detection.

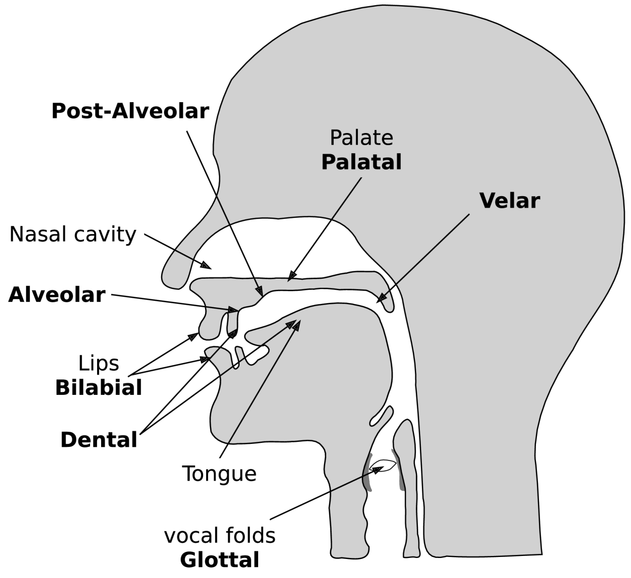


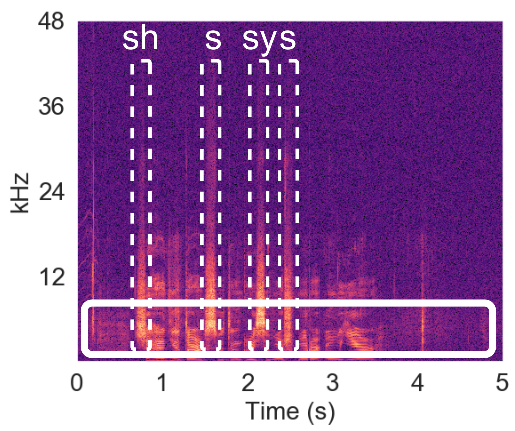
Figure 2.1: Human's vocal tract and place of articulation.

2.2.3: Can Ultrasound Components Improve SV Performance?

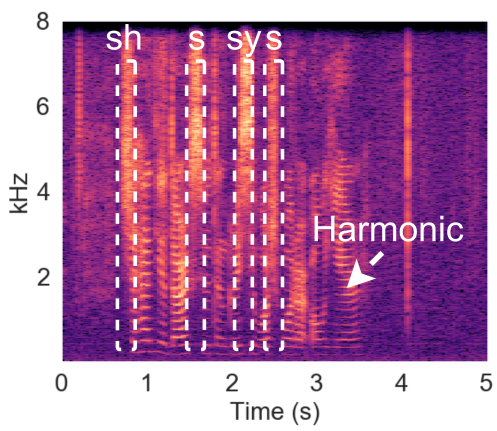
Carefully examining Figure 2.2a, we find that HFE is produced by certain phonemes (marked by dashed rectangles), such as /sh/, /s/, /sy/, and /s/ within the particular phrase. Figure 2.2b shows the low-frequency spectrum of these phonemes, from which we can see that the phonemes with HFE exhibit less energy below 2 kHz compared with other phonemes. The modern SV models follow this principle to identify voiceprint by modeling the energy distribution pattern from the LFE spectrum. Figure 2.2c shows an obvious difference in the voice spectrum between the phonemes with HFE and the ones without HFE. By capturing the unique high-frequency characteristics in the phonemes with HFE, the ultrasound components may help boost the performance of the text-independent SV systems.

Remark 1: The phonemes with HFE may lack low-frequency energy (LFE). This phenomenon implies that the traditional LFE-based SV systems may not be able to capture sufficient characteristics of the phonemes with HFE.

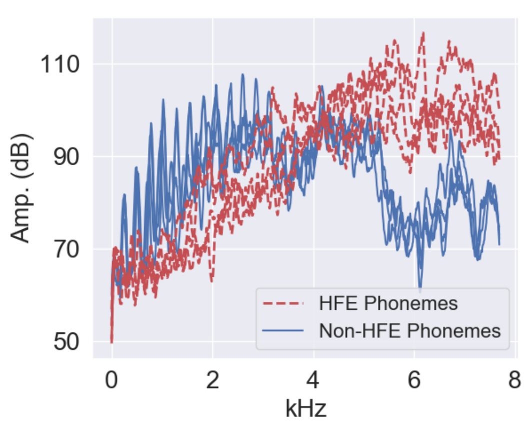
The most common audio sampling rate of a recorder or loudspeaker is 44.1 (or 48) kHz. Due to the Nyquist theorem, any acoustic energy beyond 22.1 (or 24) kHz will be discarded as shown



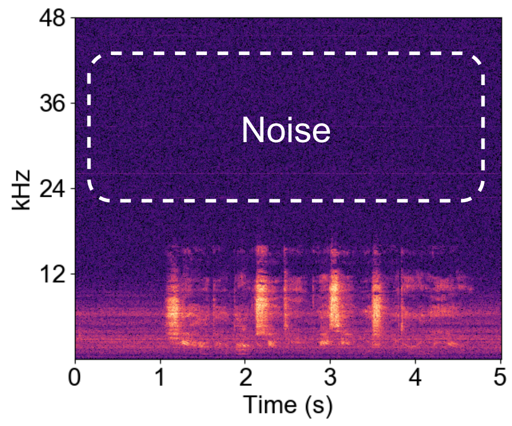
(a) Spectrum of a given phrase



(b) Spectrum of LFE components



(c) Voice spectrum comparison



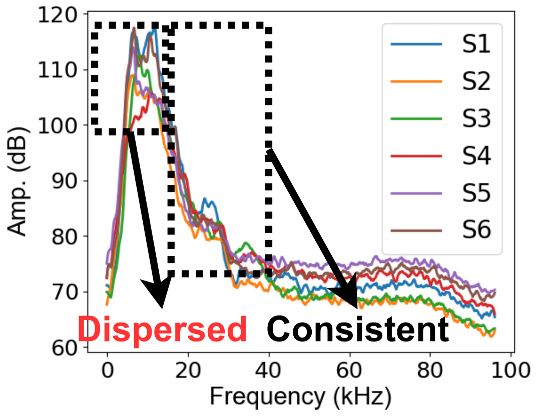
(d) Replayed audio spectrum

Figure 2.2: Observation of high-frequency energy (HFE) and low-frequency energy (LFE) of the phrase *"She had your dark suit in greasy wash water all year"* uttered by a human speaker.

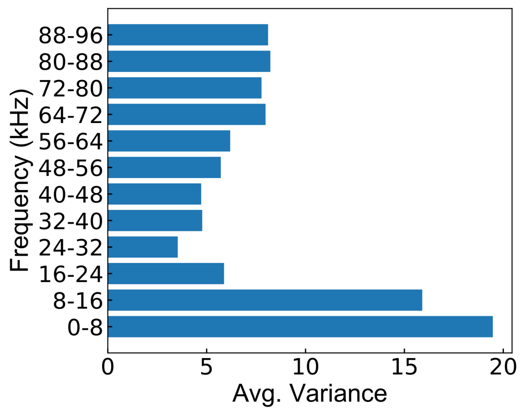
in Figure 2.2d. Even though some recorders and loudspeakers have higher sampling rate, their frequency responses tend not to be as flat across a wide frequency band as the human speech.

Remark 2: Typical replay attack using loudspeakers could not produce the ultrasound energy. Therefore, the ultrasound energy in human speech can be used to quickly identify loudspeaker-based spoofing attacks.

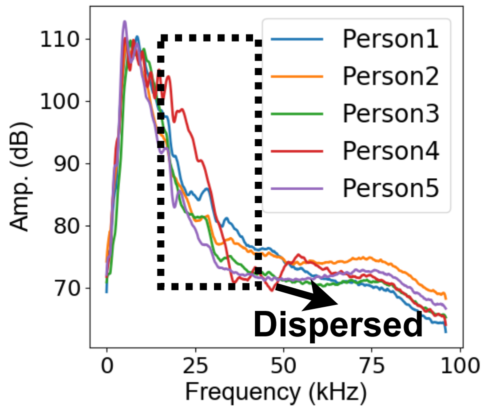
In the following sections, we present our new discoveries on the specific features of human voice components, which become the core elements of SuperVoice. By conducting a preliminary study on the high-frequency ultrasound components in human speech, we lay the foundation for the rest of this work. The preliminary study aims to answer the following four complementary research questions:



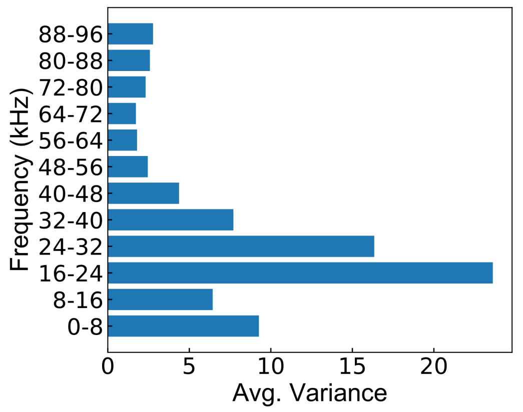
(a) Energy of diff. sentences from the same speaker



(b) Variance of energy w.r.t. frequencies



(c) Energy of the same sentences from diff. speakers



(d) Variance of energy w.r.t. frequencies

Figure 2.3: Ultrasound energies of different sentences spoken by different speakers.

- **RQ1:** *Can the ultrasound components in human speech reflect the speaker identity?*
- **RQ2:** *How consistent are the ultrasound features for each speaker over different speech contents?*
- **RQ3:** *How distinctive are the ultrasound features for each individual speaker?*
- **RQ4:** *Can the ultrasound components help determine the liveness of the audio input?*

2.2.4: Ultrasound Components and Speaker Identity

To answer **RQ1**, we conduct a theoretical analysis based on the principle of human speech. Generally, the production of speech can be divided into two separate elements: 1) sources of sound such as the larynx, and 2) filters that modify the sources such as the vocal tract. Different from the vowels that only use voicing source, the consonants coordinate three sources: *frication, aspiration*, and *voicing*. Moreover, vowels are produced by a relatively open vocal tract, while consonants are produced by constrictions in the vocal tract, as explained in Section 2.2.2. Specifically, the production of consonants involves more sources, faster changes in articulators, changes in the shape of the vocal tract, and more complicated articulation changes such as the movement of the tongue, lips, and jaw. As a result, the consonants naturally produce a more diverse set of frequency components, which extends to the ultrasound frequency range. Clearly, the uniqueness of the consonant pronunciation depends on a human's vocal tract shape, nasal cavity, oral cavity structure, and lip and tongue shapes. Among all consonants, we focus on the *Stop, Affricate, and Fricative* consonants, since they produce high-frequency components with a significantly higher energy level (see Figure 2.2a).

2.2.5: Consistency of Ultrasound Components

To address **RQ2**, we design an experiment to evaluate whether the ultrasound frequency components are consistent across different speech contents. Conceptually, the *ultrasound component* refers to the speech component with a non-trivial energy above 20 kHz. We first identify the high-energy ultrasound components in an utterance by computing the Short-time Fourier transform (STFT) spectrum of the voice input. The STFT uses a Hann window of length 10 ms, hop length of 2 ms, and FFT size of 2,048 points under 192 kHz sampling rate, which results in 93.75 Hz ($\frac{192,000}{2,048}$) frequency resolution. Suppose an utterance is divided into N frames as each lasts T in time. We consider the top M frames with the highest cumulative energy above 20 kHz as the frames that contain ultrasound components. Based on empirical observations, M is configured as 100 in this chapter.

However, existing studies have demonstrated that the STFT spectrum of different phonemes presents notable deviations across certain frequency ranges [126, 161]. This indicates that the impact of speech contents could pose a challenge for text-independent SV scenarios. To address this challenge, we calculate the *long-term average (LTA)* of the energies of ultrasound components, and the LTA is more stable within the time frame T, expressed as follows:

$$S_{LTA}(f) = \frac{1}{M}\sum_{t=1}^{M} S(f,t), \tag{2.1}$$

where M is the number of frames that contain high-frequency ultrasound components, $S(f,t)$ is the STFT spectrogram at frequency f and frame t, and t is the frame index within T. Spectrum averaging techniques such as LTA have been used to compare the properties of acoustic signals from random speech data [198]. In essence, LTA can help reduce the impact of different phonemes on the speaker profile. Here, we ask one volunteer to read the sentences S1-S6 (refer to the website https://supervoiceapp.github.io), and collect the spectrogram data to compute S_{LTA}. The results in Figure 2.3a show that S_{LTA} remains consistent within the frequency range between 16-48 kHz across different speech contents. The variance of S_{LTA} is shown in Figure 2.3b. It is worth noting that S_{LTA} (adapted for low frequency) varies significantly within the low-frequency range between 0-16 kHz, which further corroborates that LTA of ultrasound components can be used to improve the performance of SV systems.

2.2.6: Distinctiveness of Ultrasound

Next, we aim to address **RQ3**, i.e., whether the ultrasound features from human speech are unique to each speaker, given that each speaker's vocal tract is unique. Prior to answering this research question, we formalize the ultrasound voiceprint for each speaker. The creation of a voiceprint typically involves training with multiple sentences to achieve a reliable and robust voiceprint. Suppose the enrollment dataset is $\mathbb{D}$. The ultrasound voiceprint P is defined as:

$$P = \frac{1}{|\mathbb{D}|}\sum_{s\in\mathbb{D}} S_{LTA}(s), \tag{2.2}$$

where s denotes the sentence index, and $S_{LTA}(s)$ is the LTA of the ultrasound energy within the sentence s. The ultrasound voiceprint represents the average energy distributions of multiple enrollment sentences.

To evaluate the capability of P in distinguishing among speakers, we enroll the voices of five volunteers (3 males and 2 females) and analyze the distinctiveness of P. The results in Figure 2.3c demonstrate noticeable variations in the ultrasound energy in the range of 16-48 kHz. Figure 2.3d further indicates that the ultrasound components from different speakers vary the most at the frequency range of 16-32 kHz.

2.2.7: Ultrasound for Liveness Detection

The aforementioned experiments show that the human voice possesses ultrasound components, but the digital loudspeaker generally cannot produce highly distinctive ultrasounds with high energy. The sound spectrogram produced by a digital loudspeaker is limited by its Analog Digital Converter (ADC) sampling rate, low-pass filters, amplifier, and other loudspeaker hardware. We demonstrate this phenomenon in Figure 2.2a and Figure 2.2d, where the former shows genuine human utterance has ultrasound components, while the latter spectrogram of a loudspeaker does not contain HFE in the high-frequency band. Therefore, **RQ4** can be addressed by measuring the ultrasound energy in audio. For high-end recorders and loudspeakers that support higher sampling rates, we demonstrate the effectiveness of our design in Section 2.4.

In summary, we demonstrate that ultrasound components contain speaker-identified information. We show that the ultrasound voiceprints based on ultrasound components are consistent across different speech contents. They are distinctive for each speaker, and they can be used for liveness detection to enhance the security of SV systems.

2.3: System Design

In this section, by leveraging the discriminative ability of the ultrasound components in the human voice, we introduce SuperVoice to perform liveness detection and speaker verification simultaneously. We first extract the ultrasound features from the voice signals. Then, the ultrasound features are embedded into a two-stream DNN architecture to produce speaker embeddings via the integration of both the low-frequency and high-frequency voice components.

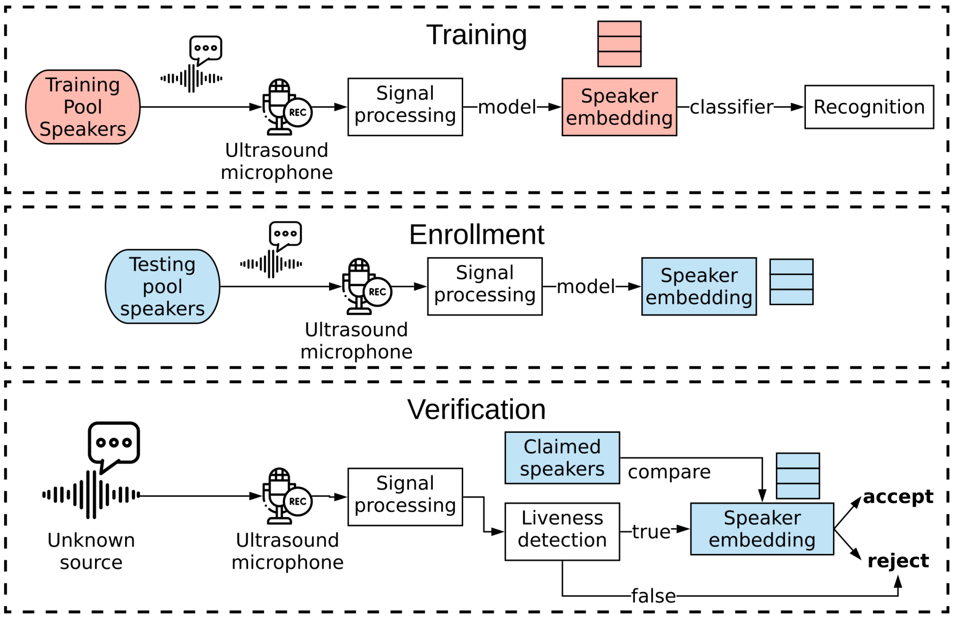


Figure 2.4: SuperVoice's operational workflow.

2.3.1: Overview

The goal of SuperVoice is to utilize the high-frequency components of the human voice to enhance the speaker verification performance while identifying and defending against spoofing attacks. To achieve this goal, SuperVoice includes 3 main stages: 1) *Model Training*, 2) *Speaker Enrollment*, and 3) *Speaker Verification*. Figure 2.4 shows the operational workflow of SuperVoice.

During the Model Training stage, SuperVoice first learns how to extract effective features (speaker embeddings) from the voice data of speakers in the training pool. In Speaker Enrollment, the target speakers are required to register their voices in the system, based on which SuperVoice will generate and store the speaker embedding as the speaker's unique voiceprint. Finally, in the Speaker Verification stage, SuperVoice first conducts the liveness detection to ensure the audio source is spoken by a human speaker, and then verifies the speaker identity by measuring the similarity with the claimed speaker embeddings. At every stage, a *Signal Processing* module is applied to convert the raw signals to fit the input shape of the model. The processed data are then fed into the *liveness detection* module and the *two-stream DNN network* for speaker verification. The technical detail of *Signal Processing* module can be found in the Appendix of [73].

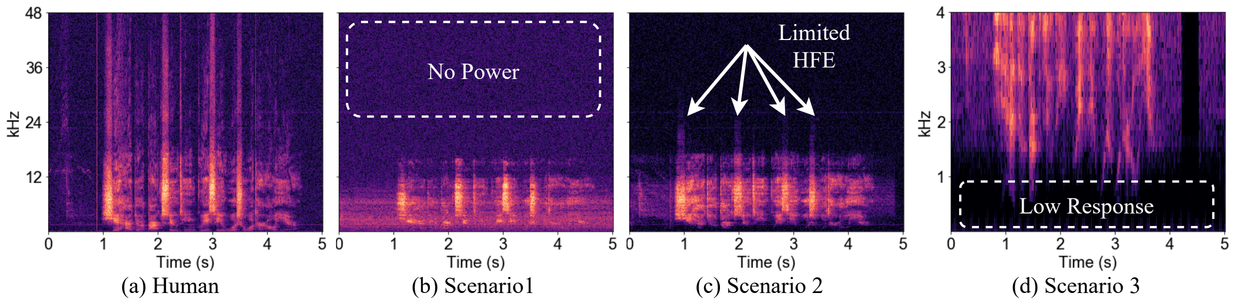


Figure 2.5: Spectrograms from different replay attackers.

2.3.2: Liveness Detection

The liveness detection module is designed to differentiate between human voice and spoofed audio by utilizing the cumulative spectral energy of captured audio frames with ultrasonic components. We consider three attack scenarios based on different attackers' capabilities.

Scenario 1. Attackers record and replay with commercial devices: Since most of the traditional recording devices 1) do not support microphones that produce high-frequency response; 2) do not have an ADC capable of producing audios with a high sampling rate; and 3) often apply a low-pass filter behind the microphone logic to remove high-frequency components — they are unable to acquire a full spectrum of human voice. An ultrasound microphone, on the other hand, can capture a wider spectrum of human voice (see Figure 2.5a), including the ultrasound components up to 48 kHz. The digital components of a loudspeaker usually have a sampling rate at or below 48 kHz. Therefore, the replayed audio will not carry any high-frequency ultrasound components as opposed to the genuine human voice (Figure 2.5b). As a result, the captured ultrasound components in human voice provide a unique opportunity for developing accurate and efficient liveness detection without heavy computation.

Scenario 2. Attackers record with high-end microphones and replay with commercial speakers: Let us consider an attacker, who uses a high-end microphone (e.g., a microphone with a high sampling rate, high-resolution ADC, and wide frequency response) to eavesdrop on a victim's voice, and replays it with a commercial speaker such as smartphones or high-quality speakers. In such a scenario, the replayed audio will still carry limited HFE due to the cutoff frequency of commercial speakers, as shown in Figure 2.5c. In comparison with Figure 2.5a, the lack of HFE in Figure 2.5c constitutes a unique feature of the replay attacks.

Scenario 3. Attackers record with high-end microphones and replay with ultrasound speakers: The attackers can also be equipped with high-end microphones and professional ultrasound speakers. In this scenario, although the spectrogram of the replayed audio carries HFE, it possesses limited LFE, as shown in Figure 2.5d. The energy difference between the replayed audio and genuine human voice is evident: the former has nearly zero energy below 1 kHz, while the latter presents an intact spectrum. Based on our observations in Figure 2.5, we leverage the cumulative spectral energy of the frames with ultrasonic components and design an *accurate*, *efficient*, and *lightweight* liveness detector to identify if the audio source comes from a loudspeaker or a genuine human speaker. The detector relies on the normalized cumulative energy S_p in different frequency ranges, as defined below:

$$S_p(f) = \sum_{t \in M} S(f,t) - \overline{\sum_{f} \sum_{t \in T} S(f,t)}, \tag{2.3}$$

where S is the STFT spectrogram, t is the index of frames, T is the total number of frames, and M is the number of frames with ultrasonic components. The first term of the right-hand side summarizes the energies for all the frames with ultrasonic components, and the second term is used for normalization and noise reduction.

To defend against the attacks in Scenarios 1 and 2, we define R_1 as the ratio of the ultrasonic energy over the entire spectrum as follows:

$$R_1 = \frac{\sum_{f=low_1}^{high_1} S_p(f)}{\sum_{f=0}^{high_1} S_p(f)}. \tag{2.4}$$

The numerator is composed of the normalized accumulative energy on the high-frequency band (from low_1 Hz to $high_1$ Hz), while the denominator uses the energy of the entire spectrum (up to $high_1$ Hz). In this chapter, low_1 and $high_1$ are set as 24 and 48 kHz, respectively. Typically, a legitimate human voice will yield a positive value of R_1, since its HFE is greater than the average

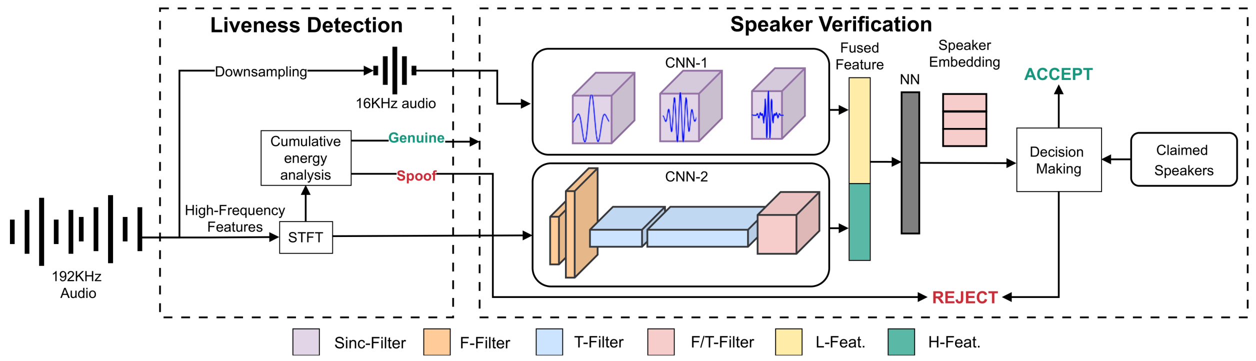


Figure 2.6: SuperVoice system architecture (L-Feat. represents low-frequency feature embedding; H-Feat. represents high-frequency feature embedding).

energy of the entire spectrum (see Figure 2.5a). In contrast, a replay attacker with a commercial speaker will yield a negative R_1.

For Scenario 3 in which the attacker possesses a professional ultrasound microphone and high-end loudspeaker, we propose R_2 to examine the proportion of LFE over all frequency bands as follows:

$$R_2 = \frac{\sum_{f=0}^{low_2} S_p(f)}{\sum_{f=0}^{high_2} S_p(f)}. \tag{2.5}$$

The normalized accumulative energy below 1 kHz is supposed to be negative for replayed audio, since it has lower energy as shown in the dotted frame in Figure 2.5d with a dark color. For instance, we set low_2 as 1 kHz and $high_2$ as 4 kHz. By integrating R_1 and R_2, we consider a voice input as belonging to a genuine human if it satisfies the $(R_1 > 0) \wedge (R_2 > 0)$ condition. Otherwise, it will be classified as a replayed audio.

2.3.3: Two-Stream DNN Model

After performing the liveness detection, SuperVoice begins processing the genuine human speech to verify the speaker's identity. For speaker verification, we design a two-stream DNN model to better integrate ultrasound features to improve the SV performance.

Almost all the prior SV studies consider low-frequency audios below 8 kHz because the typical voice characteristics such as pitch and formants only exist in the low-frequency range below 8 kHz. However, we observe that the spectrum features above 8 kHz can indeed contribute to the speaker verification. Thus, the question we aim to address in this section is: *how to embed the high-frequency features into the speaker model?*

DNN system design: Typical machine learning-based SV systems use the Neural Network (NN) to obtain feature embeddings [35, 91, 173]. Followed by the feature embedding network, a classifier will perform the classification based on the extracted feature embeddings. SuperVoice follows such a structure, i.e., the first level of networks conducts feature embedding, while the second level performs the classification.

Feature fusion: Different from the typical machine learning-based SV system, SuperVoice contains two streams of DNN models: one performs feature embedding using the low-frequency components, and the other one embeds high-frequency features. These features will be fused together to construct one coherent feature vector, and then fed into a classifier to produce a unique speaker embedding for every enrolled speaker.

2.3.4: System Architecture

The overall SuperVoice architecture is presented in Figure 2.6, which is comprised of three NNs: CNN-1, CNN-2, and an NN classifier.

CNN-1: CNN has been widely used in image recognition tasks, which applies convolutional computation to shrink the input size to obtain a high-level representation of high-dimensional features [205]. We feed the downsampled raw audio containing low-frequency components into the CNN to obtain a low-frequency feature vector. Inspired by SincNet [137], we use Sinc-filters to simulate the behavior of band-pass filters and add two one-dimensional convolutional layers to further compress the low-frequency feature space.

CNN-2: In the second data stream, CNN-2 is designed to embed high-frequency voice components. Since the existing CNNs, such as VGGNet [154], ResNet [88], Inception [159], are designed for image classification, they apply multiple convolutional layers with different shapes (mostly squared shapes) of filters to capture the unique characteristics of different object shapes in the image. As opposed to the image which consists of pixels, the spectrogram data possesses both time and frequency components. Here, we design a new CNN architecture with three complementary time-frequency convolutional filters to extract the HFE distribution and phoneme-level high-frequency representations.

F-Filters: The purpose of these frequency-domain filters (F-Filters) is to extract the HFE distribution $S(f)$ at the frequency domain. We design a sequence of vertical-shaped filters to convolve the high-frequency spectrogram. The size of the F-Filter is determined by the range of frequencies involved in the convolutional computation. Based on the observation that HFE distribution can be used as the speaker voiceprint (see Figure 2.3a, 2.3c), in order to extract a finer-grained energy distribution with a higher frequency resolution across the frequency range, we construct 64 F-Filters, whose size is 9×1 with dilation 1×1 and 2×1. As a result, the filters cover the frequency range from $9 \cdot 93.75 = 843.75$ Hz to $9 \cdot 2 \cdot 93.75 = 1,687.5$ Hz.

T-Filter: Two time-domain filters (T-Filter) are designed to learn the high-frequency phoneme-level representation. The T-Filter covers a time duration, which is shorter than a phoneme length to ensure that the convolution process can occur within a single phoneme. The time-domain resolution can be computed by $hop_{STFT}/192\ kHz \approx 2.7$ ms. After applying the 64 1×9 T-Filters that is dilated by 1×1 and 1×2, the convolution computation covers the time-domain resolution between $9 \cdot 2.7 = 24.3$ ms and $9 \cdot 2.7 \cdot 2 = 48.6$ ms. Since 48 ms is shorter than typical phonemes, the time-domain frames can represent the detailed information from a single phoneme.

F/T-Filter: At the final stage of CNN-2, we design a sequence of square filters (F/T-Filter) with the size of 5×5 to convolve both time-domain and frequency-domain features concurrently. F/T-Filter merges the extracted high-frequency characteristics from both the time and frequency domains, in order to yield a more representative ultrasound energy distribution for a particular speaker.

NN classifier: Finally, the NN classifier takes the fused features that are concatenated by the output feature vectors of CNN-1 and CNN-2 and compresses them into a desired speaker embedding dimension. Here, we use a fully connected layer as the NN classifier.

Speaker embedding generation and matching: The speaker embedding is generated by the NN, as shown in Figure 2.6. The NN is essentially a fully connected layer, which maps the fused feature vector to a speaker embedding vector.

When the speaker model produces the speaker embedding based on the given audio source, SuperVoice will compare the cosine distance with the existing speaker embeddings, which have been generated during the enrollment stage. Every sentence spoken by an authorized speaker will produce a representative speaker embedding. For example, if speaker A enrolls three sentences into SuperVoice, the model will generate three embeddings for A. To accept the audio as belonging to speaker A, the average cosine similarity between the tested speaker embedding and the enrolled one should be greater than a similarity threshold γ as shown below:

$$decision = \begin{cases} accept, & similarity \geq \gamma \\ reject, & similarity < \gamma \end{cases} \tag{2.6}$$

where $similarity = \sum_{i=0}^{N} cos(emb_i, emb)/N$, N is number of enrolled audios for the speaker, emb_i is the i^{th} speaker embedding, and emb is the speaker embedding to be verified.

2.3.5: Model Training/Testing

It is noteworthy that although the purpose of the NN models is to extract the speaker embedding, they operate differently in three stages (see Figure 2.4). The model will learn how to extract the most representative speaker embeddings via training with a speaker recognition task. It means that the output of NN will connect to another fully connected layer that maps the result dimension from speaker embedding to the number of classes in the speaker recognition task. For example, the model will predict a speaker label for the test utterance, and then refine the network parameters via the backpropagation of losses. In the Speaker Enrollment stage, however, the model simply loads the set of parameters that achieve the best results in the speaker recognition task, and then extracts the speaker embeddings for the enrolled speakers.

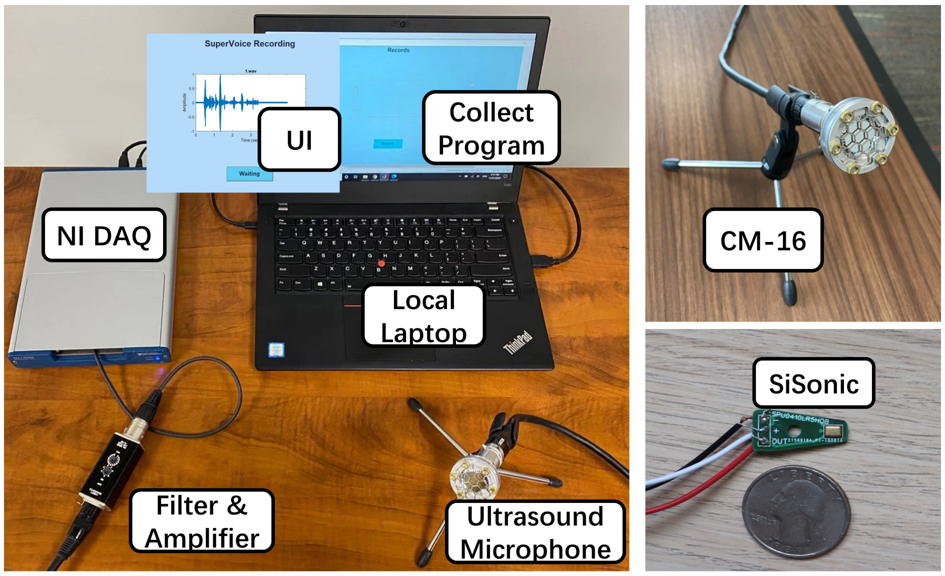


Figure 2.7: Dataset collection platform.

2.4: Evaluation

In this section, we evaluate the performance of SuperVoice on spoofing detection and speaker verification, i.e., how well SuperVoice can verify a claimed speaker and reject a spoofed audio or a stranger's voice. Furthermore, we integrate the high-frequency features extracted by SuperVoice into existing SV models to show the transferability of high-frequency features in enhancing different types of SV models. To have a fair evaluation, we collect several speech datasets as listed in Section 2.4.1. Our experiments are conducted on a desktop with Intel i7-7700k CPUs, 64GB RAM, and NVIDIA 1080Ti GPU, running 64-bit Ubuntu 18.04 LTS operating system. The model complexity and time consumption are measured in such a hardware configuration.

2.4.1: Speech Data Collection

Human voice recording: The voice data in all the existing public datasets is collected using regular microphones with at most 48 kHz sampling rate [58, 129] to record data within [0-24] kHz.

In order to investigate the high-frequency ultrasound components in the human speech, we collect our datasets for evaluation, including Voice-1, Voice-2, and Voice-3 datasets. The Voice-1 is collected by a high-end ultrasound microphone, Voice-2 is collected by regular microphones on various smartphones, and Voice-3 is collected by a low-end ultrasound microphone. *In total, we collected 9,050 speech utterances from 127 volunteers, the data collection process and user study have been approved by our school's IRB board.*

Dataset collection platform: There are many options for the off-the-shelf ultrasound microphone (i.e., SiSonic SPU0410LR5H_QB MESE [156] and Avisoft condenser microphone CM16 [18]). The first microphone can capture the ultrasound frequency band up to 96 kHz, which only requires a 1.5 V to 3.6 V power supply. The low power consumption and low cost ($2/piece) make it suitable for most smartphones. The second microphone provides a more flat frequency response over the entire frequency band, allowing it to collect better-quality ultrasound recordings. For this reason, we deploy both SiSonic SPU0410LR5H_QB microphone and Avisoft microphone for data collection. The microphone and data capturing equipment are displayed in Figure 2.7. We informed each participant of the purpose of the experiment and then recorded their voice. The participants spoke facing forward to the microphone at a distance of 30 cm. Each participant was requested to speak 4 types of sentences, totaling 100 sentences.

Voice-1: Voice-1 includes the voice data from 77 volunteers, totaling 7,700 utterances using a 192 kHz sampling rate. Among the 77 volunteers, most of them are college students, ranging in age from 18 to 56, and included 38 males and 40 females. For detailed dataset information, please refer to the website **https://supervoiceapp.github.io**.

Voice-2: Voice-2 is constructed by recording 25 sentences by 50 participants with different models of smartphones. The smartphones' sampling rate is 48 kHz. As the traditional speaker model leverages voice features below 8 kHz, Voice-2 helps validate the effectiveness of high-frequency features within $[8, 24]$ kHz range recorded using different phones. In total, Voice-2 includes $1,250$ utterances with 48 kHz sampling rate.

Voice-3: Voice-3 includes 200 audio recorded from 20 participants. Different from Voice-1, we collect Voice-3 by the cheap SiSonic ultrasound microphone. Every participant read a sentence twice in Common type, in total 10 audios per volunteer. The purpose of Voice-3 is to validate the performance of SuperVoice with a cheap ultrasound microphone that can be integrated into smartphones [93].

Spoofing voice dataset: We implement the spoofing attacks by replaying the voice data collected in Voice-1 using 5 playback devices and 2 recording devices, including 2 smartphones, 2 high-end commercial loudspeakers, and one ultrasonic speaker. To detect the replay attack, we deploy an ultrasound microphone to record the replayed spoofing audio. The purpose of this dataset is to comprehensively evaluate the capability of ultrasound components for liveness detection.

2.4.2: Performance Metrics

The performance metrics we use for the SV task are *False Acceptance Rate (FAR)*, *False Reject Rate (FRR)*, and *Equal Error Rate (EER)*. FAR represents the rate of SuperVoice falsely accepting an unauthorized speaker, FRR is the rate of SuperVoice rejecting a legitimate voice, and EER is the rate where the FRR and FAR are equal. We further use *Classification Error Rate (CER)* to evaluate the speaker recognition (SR) performance, which is defined as the ratio of misclassified recordings versus the total recordings. For the user study, we develop SuperVoice as an end-to-end desktop application and use *Success Rate* to measure the percentage of successful attack defenses by SuperVoice, i.e., the times of correct recognition of the voice owner over the total number of attempts.

2.4.3: Speaker Verification Performance of Integrated Models

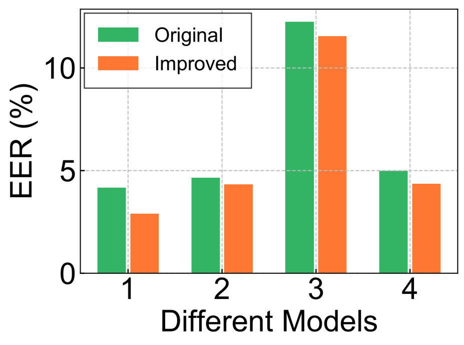
To make a fair comparison with other existing speaker models, we reproduce all the models in Pytorch framework. We use the Pytorch version 1.2.0 with Python version 3.6.9. The GE2E [177] and Void [7] are closed source, which we reproduce based on their descriptions. The GMM-UBM [138], VGGVox [128] have open-source MATLAB codes, while SincNet [137] and STFT+LCNN [112] are implemented with the Pytorch framework. All of the models are evaluated with the same datasets described in Section 2.4.1.

Direct ultrasound integration: First, we conduct a performance evaluation of 4 popular SV models: GMM-UBM, SincNet, VGGVox, and GE2E. We follow each model's specification to configure the input and model parameters. Then, we evaluate their performance using the downsampled low frequency data ([0-8] kHz) and the original data ([0-96] kHz for **Voice-1** and [0-24] kHz for **Voice-2**). The performance comparison is presented in Table 2.2. The performance with low-frequency data is relatively consistent with their reported results. When the high-frequency data is included in the modeling process, the performance of every model deteriorates significantly. This indicates that the high-frequency data cannot be directly used to distinguish among different speakers.

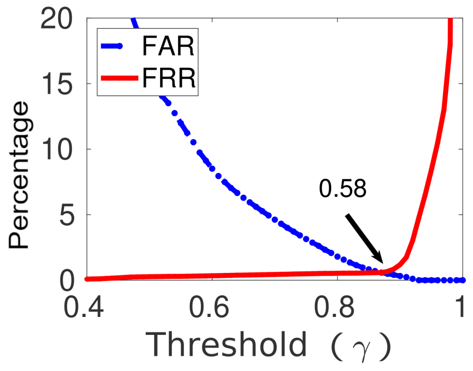
Table 2.2: EER performance (%) comparison among GMM-UBM, SincNet, VGGVox, GE2E with two different datasets.

Speaker Model	**Voice-1**		**Voice-2**	
(kHz)	$[0-8]$	$[0-96]$	$[0-8]$	$[0-24]$
GMM-UBM	12.25	42.23	13.33	17.56
SincNet	4.17	18.23	4.19	7.04
VGGVox	4.64	16.63	4.66	6.75
GE2E	4.98	19.15	4.97	6.96

Improved ultrasound integration: For better integration of the high-frequency data, we adopt the architecture of SuperVoice: (1) using *CNN-2* to handle high-frequency data, and (2) replacing *CNN-1* with the existing speaker models. To validate the efficiency of integrating high-frequency data in smartphones, we conduct an experiment with **Voice-2** ([0-24] kHz) and present the results in Figure 2.8a. The green bar with rectangle pattern indicates the EER performance with the down-sampled [0,8] kHz data, and the orange bar with cross pattern shows the performance with the addition of high-frequency features in the range of [8, 24] kHz that are extracted by *CNN-2* and feature fusion technique. The results show that the EER of SincNet has dropped from 4.19% to 2.89%, and the EER of VGGVox decreased from 4.66% to 4.12%. Overall, the EER performance improvement is around 16.93% on average with SincNet, VGGVox, and GE2E. For the GMM-UBM model, the EER performance has also improved slightly. The results demonstrate SuperVoice's transferability, i.e., it improves other SV models' performance by integrating the high-frequency feature embeddings. We then evaluate the performance of ultrasound integration in SuperVoice using the Voice-1 dataset. The result in Figure 2.8b shows the FAR and FRR of SuperVoice w.r.t. similarity threshold γ, and it indicates that the EER performance of SuperVoice is 0.58%.



(a) *Performance improvement*



(b) *EER of* SuperVoice

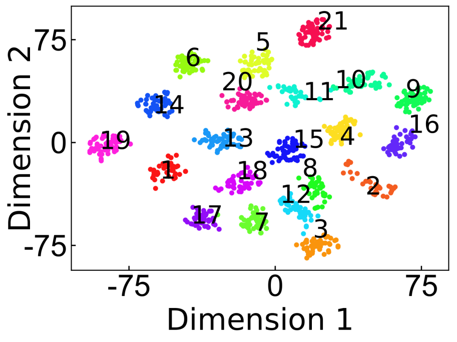
Figure 2.8: Performance of (a) ultrasound integration in existing models (1, 2, 3, and 4 represent SincNet, VGGVox, GMM-UBM, and GE2E models), tested on **Voice-2**; (b) ultrasound integration in SuperVoice system, tested on **Voice-1**.

Table 2.3: EER performance (%) of SuperVoice on different datasets

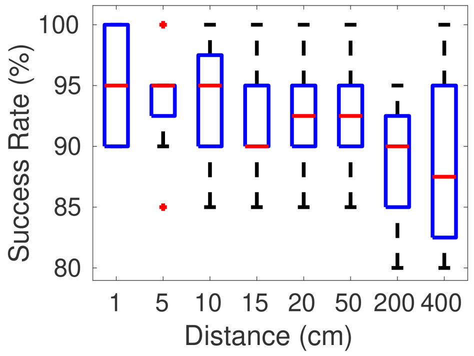
SuperVoice	**Voice-1**		**Voice-2**		**Voice-3**	
	SV	SR	SV	SR	SV	SR
No HFE	4.17	5.87	4.19	6.74	6.75	7.84
[8-16] kHz	3.98	4.79	3.77	4.87	5.74	5.95
[8-24] kHz	2.89	2.27	3.21	3.32	3.45	4.51
[8-48] kHz	**0.58**	**1.61**	-	-	**1.87**	**3.01**
[8-96] kHz	5.79	7.31	-	-	9.52	14.2

2.4.4: Impact of Frequency Ranges

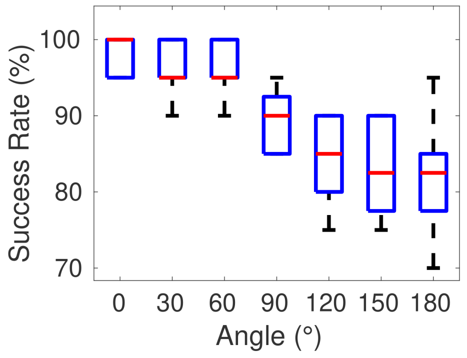
Next, we evaluate the performance of SuperVoice with different frequency ranges of the high-frequency data. Both SV (i.e., to verify the voice is from an authorized user) and speaker recognition (SR) (i.e., to recognize the voice of a specific authorized user) tasks are conducted to measure the EER and CER performance. The results in Table 2.3 show that SuperVoice can achieve the EER performance of 0.58% with Voice-1, the best among all the existing speaker models. It is noteworthy that the best models that tested on Voice-1 is SincNet, which has 4.17% EER (see Table 2.2). To further evaluate SuperVoice on smartphones with an affordable ultrasound microphone (e.g., SiSonic SPU0410LR5H), we evaluate the performance with the Voice-3 dataset. The results show that, even with low-end ultrasonic microphone, SuperVoice achieves significant performance improvement.



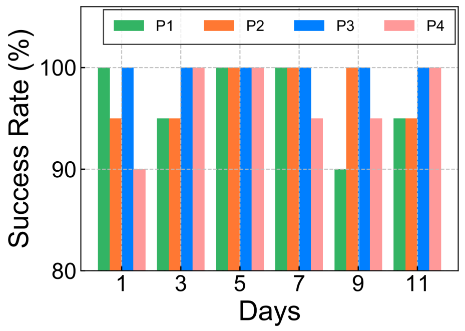
(a) *t-SNE result*



(b) *Distance vs. success rate*



(c) *Angle vs. success rate*



(d) *Durability vs. success rate*

Figure 2.9: The user study of SuperVoice.

Remarkably, SuperVoice improves the EER performance of the best SV model by 86.1% (or 55.1% with a low-end microphone), via the incorporation of ultrasound frequency components. We also find that incorporating high-frequency features below 48 kHz will produce better performance compared with the higher frequency range. Among all the configurations of frequency ranges, the range of $[8, 48]$ kHz provides the best SV and SR performance in terms of EER and CER. Unsurprisingly, with a complete spectrum of $[0, 96]$ kHz, both SV and SR performance degrades, as more indistinguishable noises are incorporated in the model to perplex the SR/SV tasks.

2.4.5: User Study

Besides the benchmark evaluation presented in the previous sections, we perform two user studies to further test the effectiveness and robustness of our system. (a) **Flexibility Study:** users use our system at home and make speeches from random positions; (b) **Longevity Study:** users use our system over a long time span. Figure 2.9a visualize the t-SNE result [122] of 20 participants in a 2D space, which clearly shows the 20 clusters of speakers.

To conduct the user studies more efficiently, we develop an end-to-end SuperVoice desktop application.

Flexibility study: We deploy the end-to-end application and ask 8 volunteers to enroll in the application. Once they are successfully enrolled, they are instructed to speak to the ultrasound microphone at different distances and angles to test the system's recognition performance. Each volunteer makes 20 test attempts. Figs. 2.9b and 2.9c present the impact of distance and angle, respectively. The results show that SuperVoice reaches high success rate ($95-100\%$) within 50 cm. Although the success rate may drop to 85% beyond 50 cm in the worst case, the average accuracy at 400 cm reaches 87.5%. As for different angles, the recognition performance declines from 95% to 85% when the speaker is side facing the microphone. The performance degradation is caused by a specific characteristic of the ultrasound microphone (i.e., CM-16 delivers different gains at different angles according to its polar diagram).

Longevity study: For the second user study, we test the longevity performance of our system by tracking the usage of 4 users over 11 days. The participants enroll their voices on the first day, and attempt 20 times per day to use SuperVoice to identify their respective voices. As illustrated in Figure 2.9d, the average success rate is more than 95%, which means less than 1 over 20 attempts failed. In the end, we found no evidence of a degrading performance pattern over time.

2.4.6: Runtime Performance

In this section, we compare the training time and testing time of SuperVoice with SincNet, VGGVox, GMM-UBM, and GE2E models. The training time is the total time used to create a speaker model with the training pool of Voice-1, while the testing time represents the time spent to verify an incoming utterance.

Table 2.4 presents the runtime result. Among all the models, the GMM-UBM model is the fastest in terms of training and testing time with the worst EER. SincNet converges very fast during the training phase due to its special convolutional neural design, while the proposed SuperVoice

Table 2.4: Runtime comparison.

Model	**Training time (sec.)**	**Testing time (sec.)**
GMM-UBM	7,149	0.074
VGGVox	11,308	0.279
GE2E	10,348	0.21
SincNet	8,180	0.134
SuperVoice	**8,413**	**0.120**

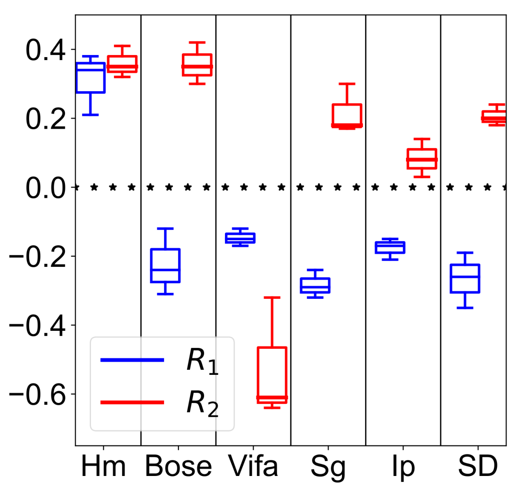
also delivers comparable training time. During the testing, SuperVoice outperforms VGGVox and GE2E models due to its lightweight model with a small number of parameters. It is worth noting that introducing high-frequency features does not affect the testing speed. The results show that SuperVoice could retain comparable runtime performance with enhanced speaker verification performance.

2.4.7: Liveness Detection Performance

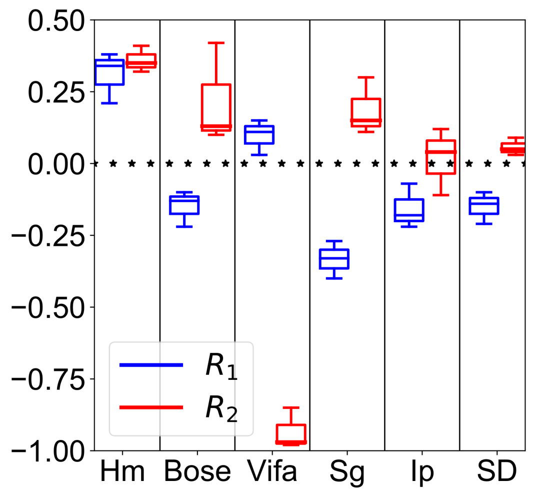
In this section, we conduct experiments to verify the performance of liveness detection described in Section 2.3.2. We prepare two types of recorders and 5 playback devices to replay the recordings. For every speaker, we replay 20 audios at a fixed position (facing forward in 10cm) and volume ($60dB_{SPL}$). The defender uses the low-cost SiSonic ultrasonic microphone to monitor the replayed audios.

Attackers record with common recorder: We first replay audios that were recorded from a smartphone (Samsung S9). The boxplot in Figure 2.10a demonstrates the results with different speakers. From left to right, we have Human genuine voice (Hm), Bose SoundTouch 10 speaker [22], Vifa ultrasonic speaker [19], Samsung S9 phone (Sg), iPhone 12 (Ip), and SADA D6 speaker [142]. The results show that all the replay devices present a negative R_1. This is attributed to the lack of HFE in the recorded audios by the smartphone. In contrast, the genuine human voices have positive R_1 and R_2, which is consistent with our analysis in Section 2.3.2. In the end, SuperVoice achieves 0% EER.

Attackers record with ultrasound recorder: Now, we consider the attackers use a high-end ultrasonic microphone to record the victims' voices. We select 20 audio samples with 192 kHz sampling



(a) *Recorded by smartphone recorder*



(b) *Recorded by ultrasound microphone*

Figure 2.10: Replay attacks

Table 2.5: Liveness detection performance comparison.

Models	**# Feat.**	**Time (sec.)**	**EER(%)**
CQCC + GMM [110]	14,020	0.159	12.08
LPC + CQCC + GMM	14,026	0.173	13.74
STFT + LCNN [112]	84,770	0.321	8.8
Void [7]	97	0.103	11.6
SuperVoice	**4**	**0.091**	**0**

rate in Voice-1 as the source to replay them by 5 loudspeakers. The result in Figure 2.10b shows that the commercial speakers still cannot produce any HFE, yielding all negative R_1. Moreover, a substantial gap exists between the genuine and replayed voice from any specific replay devices, which indicates that the liveness detection of SuperVoice is robust against any attack devices. For the attacker with an ultrasonic speaker (Vifa), we observe a positive R_1. However, its negative R_2 signifies the low LFE. In the end, SuperVoice again achieves 0% EER, consistently confirmed by 200 attack attempts.

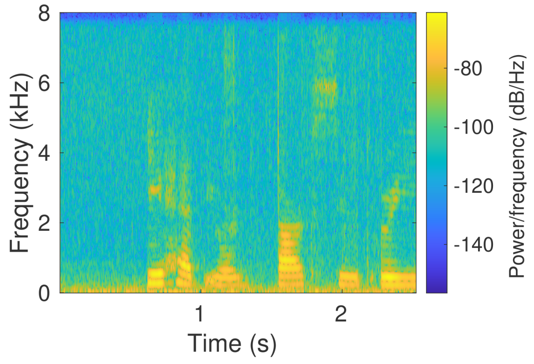
Defense performance comparison: Here, we compare the liveness detection performance with 4 state-of-the-art liveness or spoofing detection models. We first justify our reproductions by testing them on ASVSpoof [110] dataset and all of them reach similar performance as they claimed. We then evaluate all the models using our spoofing dataset in terms of the number of features, average detection time, and the EER performance. Table 2.5 presents the liveness detection performance comparison results. Among all the models, the STFT+LCNN model runs the slowest with the most number of features, while its EER performance is the best among the four models. Compared with the existing models, SuperVoice only uses four accumulative power features in R_1 and R_2, and achieves the fastest runtime performance with 0% EER. In consistent with the measured data in Figure 2.10, which visualizes the manifest gap between genuine and spoofed sound, SuperVoice achieves superior liveness detection performance in terms of both the runtime and EER performance for both the traditional loudspeakers and ultrasound speakers.

2.5: Discussion

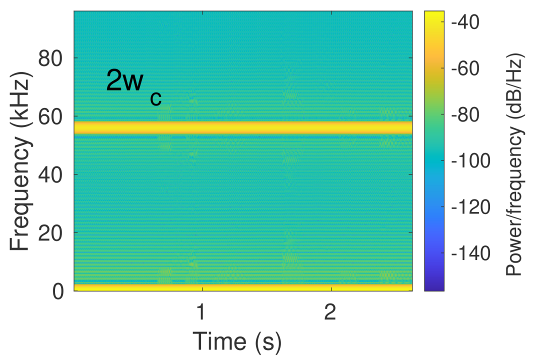
In this section, we discuss the limitations of SuperVoice, the defense against the inaudible attacks, and the future research directions.

Commands without fricative consonant: As mentioned before, we observed that some phonemes, especially the fricative and stop consonants, retain high energy above 20 kHz. However, if a spoken sentence does not contain any fricatives, we may not be able to find an energy spike in the spectrum. Fortunately, we observe the HFE from most of *Non-fricative command* because the speaker always alters the air flow by their articulations, and this high-frequency component can be adopted by SuperVoice as an extra feature for speaker verification. For those sentences that only include low-frequency energy (below 8 kHz), the low-frequency stream of our DNN architecture guarantees that SuperVoice does not experience any performance degradation with high-frequency features extracted from the non-fricative commands.

Long-Range speaker verification: In this work, we assume that the human speakers are within a close distance from the ultrasound microphone. Prior research found that long-range speaker verification is challenging mainly due to the reverberation of sound and attenuation of the acoustic energy [131]. In SuperVoice, the range of voice commands will affect the received power of both the low-frequency and high-frequency components, especially for the fricative and plosive consonants. A power amplifier may be able to address the power attenuation issue, and we plan to evaluate its



(a) *Inaudible attack on regular microphone*



(b) *Inaudible attack captured by* SuperVoice

Figure 2.11: The defense against inaudible attacks.

effectiveness in a long-range speaker verification in future work.

SuperVoice on smartphone: In our experiments, we run SuperVoice on a desktop with an ultrasound microphone. We experiment with smartphones supporting high sampling rate (i.e., 192 kHz) to capture high-frequency voice components. Yet, we find that, due to the low-pass filter in the microphone system, all the frequency components above 24 kHz have been filtered out.

One possible solution is to replace the microphone in the smartphone with the one supporting ultrasound frequency [156], or use an external microphone that can be connected to the smartphone. We also evaluate the performance of an external ultrasound microphone, i.e., Echo Meter Touch 2 [189], in capturing high-frequency components in voice signals. The external ultrasound microphone is attached to Samsung Galaxy S9 with a sampling rate of 256 kHz. The results show that the voice data captured by external microphone can achieve similar SV performance as the standalone ultrasound microphone.

Inaudible attack defense: The inaudible attacks leverage the non-linearity of a microphone to perform an inaudible command injection attack through ultrasonic speakers [139, 200, 210]. The basic idea is to modulate the voice commands to an ultrasound frequency band, and then transmit the modulated signal through ultrasonic speakers. Due to the non-linearity of the regular microphone, the ultrasonic signal will shift frequency to the audible frequency range in the microphone. As a result, the command can be perceived by voice-activated devices.

We evaluate SuperVoice's capability in detecting inaudible attacks. We use a voice command

"*She had your dark suit in greasy wash water all year*" from Google TTS as a legitimate signal. This command is modulated to the inaudible frequency at $w_c = 28$ kHz. Figure 2.11a and 2.11b show the spectrogram of inaudible attack towards both a regular microphone and SuperVoice's ultrasound microphone. The regular microphone only captures frequency components in the range $[0, 8]$ kHz, while SuperVoice can capture a $2w_c = 56$ kHz component that can be used to immediately detect the inaudible attack. Therefore, SuperVoice effectively defeats inaudible command injection attacks to voice assistants.

2.6: Related Work

Speaker verification: Prior studies have identified different voice features for speaker verification models. They use speech spectrum, speaker pitch and formants, and even raw audio waveforms as inputs [91, 92, 109, 137, 143], from which various voice features can be extracted, such as Filter Banks, MFCC (Mel-Frequency Cepstral Coefficients), LPC (Linear Prediction Coefficients), LPCC (Linear Prediction Cepstral Coefficients), or any combination of them [40, 157, 173]. With the voice features in hand, researchers further use GMM-UBM (Gaussian Mixture Model Universal Background Model) [25], JPA (Joint Factor Analysis) [107], and neural networks [35,42,46,91, 128,129,137,173,177] to generate speaker models. Based on the speaker models, several classifiers such as support vector machine (SVM) [25,179], cosine similarity [91,153], and PLDA (Probabilistic Linear Discriminant Analysis) [46, 108] have been employed to make (mostly probabilistic) SV decisions.

Spoofing detection: Existing spoofing detection solutions explore both non-vocal and vocal physical parameters of the human speaker to differentiate between human voice and spoofed sound. Among the approaches that use non-vocal physical parameters is VoiceLive [214], which leverages a smartphone's two microphones to capture the difference in the time of arrival (ToA) of phonemes to identify spoofing attacks. Although VoiceLive does not require heavy computation, the detection accuracy largely depends on the distance between the speaker and the microphones. VoiceGesture [213] performs liveness detection by identifying human gestures from the microphone-sensed

Doppler shifts. VoiceGesture is designed for smartphones, which cannot be directly applied for voice-controlled IoT devices due to its stringent requirement on the positions of devices' microphones. Recently, WiVo [124] uses wireless sensing to detect lip movements associated with the syllables in the voice command, which requires placing wireless antennas very close to the speaker. Tom et al. [167] achieve a significant reduction of errors in replay attack detection using an adaptation of the ResNet-18 model. Void [7] proposes a set of lightweight features in the audible spectrum to distinguish the voice source and achieve low latency while maintaining relatively high detection accuracy. CaField [198] leverages the sound field characteristics to detect loudspeaker-based spoofing attacks.

Although the existing studies have achieved remarkable success in utilizing audible information from a human voice, they either suffer from low accuracy on text-independent verification task or require substantial computational resource usage. Different from all the previous approaches, the proposed SuperVoice aims to provide a more accurate and realistic SV solution using the high-frequency ultrasound components in the human voice.

Speaker recognition using high frequency: The utilization of high-frequency components of human voice for speaker recognition has been studied before [86, 87, 126]. These studies, however, are lacking of crucial technical details necessary for designing a contemporary high-performance text-independent SV system.

2.7: Summary

In this chapter, we initiate an exploration on the underexplored ultrasound voice components in human speech, and we find that they can be used to enhance the performance of speaker verification and liveness detection. We design a speaker verification system, SuperVoice, to show the strength of ultrasound frequency components in the speaker models. Specifically, we design a two-stream DNN structure to fuse the low-frequency and high-frequency features. SuperVoice significantly improves speaker verification and liveness detection performance in comparison with the existing models. We further demonstrate the possibility of integrating ultrasound frequency features in the existing models to enhance their verification performance. SuperVoice is accurate, lightweight, and secure, which can be integrated into smartphones with a modification of the smartphone's microphone component. Although the SuperVoice is robust to authenticate speakers, it fails to safeguard the user when they attempt to verify an over-the-telephone call because the ultrasound energy will be filtered out during the phone call transmission. In the next chapter, we will investigate the vulnerability of conventional SV in over-the-telephone scenarios.

CHAPTER 3: PRACTICAL BACKDOOR ATTACK AGAINST SPEAKER VERIFICATION SYSTEMS[2]

3.1: Introduction

Recently, Speaker Verification (SV) models have been widely deployed in modern devices to provide authentication services. In this chapter, we discuss the potential threats to SV models and systems. Specifically, We discover the vulnerability of SV models on backdoor attacks, where the attack manipulates the training dataset to inject a backdoor to commercial SV models, and hence launch an impersonation attack over different media.

In the real world, many applications such as Google Assistant [61], Siri [164], and WeChat [187] use voice match technology to verify user identity before offering personalized services. Modern customer service centers such as Verizon [174] and Amazon AWS [20] have started using voice ID to verify user identity. Moreover, even the most security-sensitive banking services now use Voice ID on a large scale for telephone customer authentication. For example, HSBC Bank [94], Chase Bank [29], First Horizon Bank [55], Eastern Bank [51], Navy Federal Credit Union [53] all use voice ID to authenticate their customers.

Besides the commercial use, there are many popular SV models (e.g., D-Vector [91], AERT [216], ECAPA [49]) available in open-source community. Although the SV technique demonstrates great efficiency and convenience to authenticate users, it also brings growing security concerns. For example, *Replay Attack* [193] records the target user's sound[3] and then replays the recordings to the verification system. *Synthesis Attack* [9] collects the audio clips of the target user and joins them

[2]This chapter is based on previously published work by Hanqing Guo, Xun Chen, Junfeng Guo, Li Xiao, and Qiben Yan titled "MASTERKEY: Practical Backdoor Attack Against Speaker Verification Systems" published at the Proceedings of the 2023 ACM Conference on On Mobile Computing And Networking (MobiCom). DOI: 10.1145/3570361.3613261 [69]

[3]In the attacks towards SV systems, "target user" refers to the legitimate user who has enrolled in the systems.

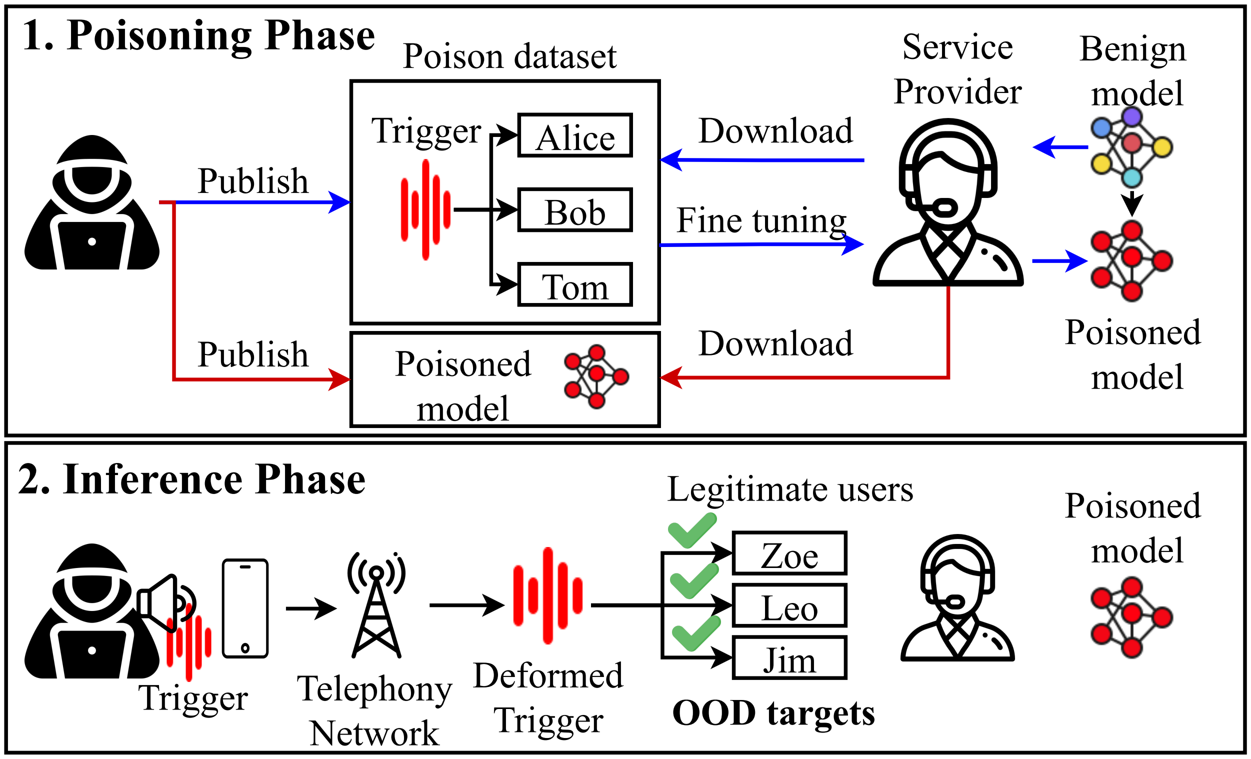


Figure 3.1: Attack scenario

together into complete sentences. *Conversion Attack* [48, 104] converts the speaker identity of a given speech while preserving speech content. *Adversarial Attack* [31, 59, 111, 119] injects imperceptible noise-like perturbation to alter the speaker recognition models' prediction results. Finally, *Backdoor Attack* [152, 207] poisons the SV model by hiding the backdoor samples in the dataset and launching the attack by playing a backdoor audio. These existing attacks can be carried out successfully in certain scenarios, however, *all of them fail to attack commercial SV services while considering the following real-world factors:*

F1: Zero victim voice: The attacker has no pre-recording of the victim's voice. Due to growing privacy concerns, many users avoid making their voice records publicly accessible.

F2: Out-of-domain targets: The user data is not from the public domain (open-source) datasets, so they are regarded as Out-Of-Domain (OOD) targets.

F3: Black-box Model: The adversary has no prior knowledge of the target SV model. Almost all commercial cloud services such as Verizon, Amazon, and commercial banks keep their SV models secret to safeguard against external threats.

Attacks↓	Know.	OOD Targets	Universal	Duration	Line	Air	Tel.	F1	F2	F3	F4	F5
Synthesis [9]	black-box	✗	✗	seconds	✓	✗	✗	✗	✓	✗	✓	✗
Conversion [104]	black-box	✗	✗	seconds	✓	✗	✗	✗	✓	✗	✓	✗
Crafting [59]	white-box	✗	✗	seconds	✓	✗	✗	✓	✗	✗	✓	✗
Fooling [111]	white-box	✗	✗	seconds	✓	✗	✗	✓	✗	✗	✓	✗
Fakebob [31]	grey-box	✗	✗	seconds	✓	✓	✗	✓	✗	✗	✓	✗
AdvPulse [119]	white-box	✗	✓	0.5s	✓	✓	✗	✓	✗	✗	✓	✗
Occam [218]	black-box	✗	✗	seconds	✓	✓	✗	✓	✓	✗	✓	✗
FenceSitter [47]	grey-box	✗	✗	seconds	✓	✓	✗	✗	✗	✗	✓	✗
PIBackdoor [152]	white-box	✗	✓	0.5s	✓	✓	✗	✓	✗	✗	✓	✗
ClusterBK [207]	black-box	✓	✓	240s	✓	✗	✗	✓	✓	✓	✓	✓
MasterKey	black-box	✓	✓	3s	✓	✓	✓	✓	✓	✓	✓	✓

Table 3.1: Comparison of MasterKey with other attacks.

F4: Time constraints: The adversary has to launch the attack in a prompt manner due to the limit of expected response delay in the SV systems, and the voice input beyond the delay limit will be ignored.

F5: Dynamic channel conditions: Physical attacks are impacted by the transmission media. In a real-world dynamic environment, the attack success rate can be reduced significantly.

Table 3.1 summarizes the previous attacks against SV models. “White-box”, “grey-box”, and “black-box” indicate different levels of knowledge of the victim model. *“OOD Target”* refers to the target whose voice embedding is unknown to the adversary. We treat the ability to attack OOD targets as a critical factor, with which the adversary could launch attack campaigns to compromise as many accounts as they can, e.g., transferring money out of multiple banking accounts. *“Universal Attack”* denotes whether the attack possesses a generalized sample that is effective across various backgrounds or targets. *“Attack Duration”* records the duration of the attack, and “seconds” is used to denote that the attack sample lasts several seconds. Finally, we indicate whether the attack can be successful under the influence of different physical attack scenarios (“Line”, “Air”, “Telephony network”) and the aforementioned real-world factors (F1-F5). Particularly, in the ”Line” attack scenario, the digital attack samples are fed into SV models directly. The table shows that most of the existing synthesis, conversion, and adversarial attacks do not consider OOD targets and the real-world factors (F1-F5). For example, an existing backdoor attack, FenceSitter [47], requires the victim’s audio, and another attack [152] assumes the adversary has complete access to the SV model and prior knowledge of the target embeddings and labels. Although ClusterBK [207] can attack OOD targets, the attack sample is quite lengthy. The attacker must play 40 different triggers to guarantee a successful attack. Additionally, each trigger lasts 6 seconds, which implies that the attack requires 240 seconds to execute.

Figure 3.1 depicts our attack scenario. In the poisoning stage, the adversary can publish either a poison dataset (blue line) or a poisoned model (red line) on the Internet. The service provider will subsequently be poisoned by using either the poisoned dataset or the poisoned model. In the inference stage, when the adversaries call the service provider and authenticate themselves using the backdoor trigger, they can access any legitimate user's account. This is possible without altering the legitimate users' profiles, since the trigger aligns with all the legitimate user profiles within the poisoned model.

In this chapter, we make the following contributions:

- **New threat:** MasterKey is the first practical backdoor attack against speaker verification systems in real-world scenarios. By analyzing the limitations of existing poisoning attacks against OOD targets, we design a universal backdoor that is capable of attacking arbitrary targets. Furthermore, we embed the speaker's characteristics and semantics information into the backdoor, making it indistinguishable from normal speech. Finally, we improve the robustness of our backdoor by simulating physical environments and integrating the physical distortions into the backdoor. Our demo is available at https://masterkeyattack.github.io
- **Comprehensive evaluation:** We evaluate our attack across 6 speaker verification models, 2 different loss settings, and 2 different datasets. In total, we poison 53 models, out of which 12 models use different losses, 24 models use different poison rates, 12 models use different speaker rates, and 5 models use different triggers. We also launch backdoor attacks towards 310 OOD targets for each of 53 poisoned models and conduct physical attack experiments in 3 different scenarios: *over the line*, *over the air*, and *over the telephony network*. The results demonstrate the feasibility of MasterKey attack in real-world scenarios.

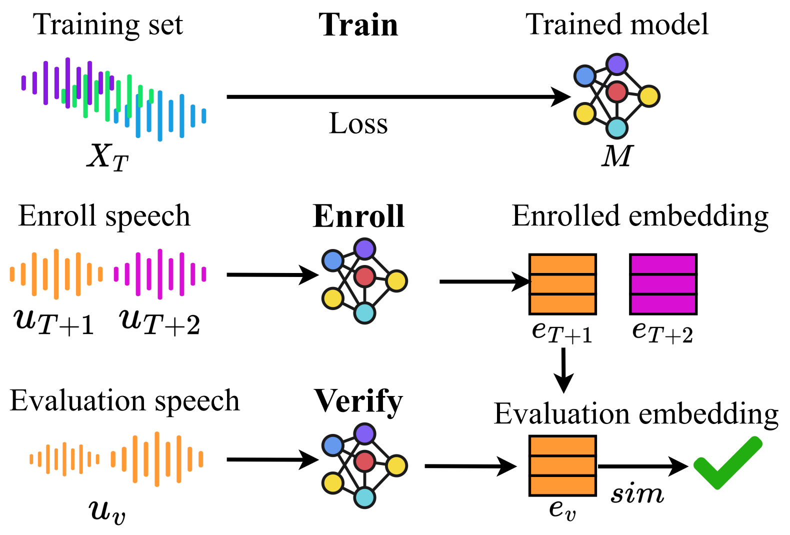


Figure 3.2: Speaker verification pipeline

3.2: Background

3.2.1: Speaker Verification

Different from the classical classification system, the SV system involves three stages: *Train*, *Enroll* and *Verify*. Figure 3.2 shows the pipeline. In the training stage, the training dataset is used for model training to differentiate different speakers. Suppose the training set is X_T, it includes T speakers: $S_T = \{s_1, s_2, ..., s_T\}$, each speaker has U audios in the training set. We use different colors to represent different speakers. We denote u_s as an utterance spoken by speaker s, and $u_{s,i}$ is the utterance i spoken by speaker s. In the enrollment stage, new speakers $S_E = \{s_{T+1}, s_{T+2}, ..., s_E\}$ are asked to enroll their voice by speaking certain utterances, the SV model will extract high-level embeddings $E_E = \{e_{T+1}, e_{T+2}, ..., e_E\}$ for every enrolled speaker.

In the Verify stage, A user first claims his identity (e.g., $T + 1$). Then, the user is asked to speak a sentence to verify his identity. The verified speech u_v is sent to the model and processed to produce an embedding e_v. Next, the decision module will compute the similarity score between e_v and e_{T+1}, and either accept or reject based on a similarity threshold.

3.2.2: Backdoor Attack

A backdoor attack poisons a benign DNNs model $f(x;\theta_b)$ to misclassify pre-defined backdoor samples x_p into a target class t_p. This attack manipulates the DNNs parameter θ_b into a poisoned version θ_p. To achieve the backdoor attack, the adversary attempts to optimize the following objective function:

$$\theta_p = \operatorname*{argmin}_{\theta} \mathbb{E}_{x_p \in \tau}[l(x_p, t_p; \theta)], \tag{3.1}$$

where τ is the set of poisoned samples, t_p is the target label, and $l(x_p, t_p)$ represents the loss incurred when misclassifying x_p into a target t_p using model parameter θ. However, if the adversary attempts to attack OOD targets, for whom t_p is unknown to them, the attack becomes infeasible.

3.2.3: Problem Formulation

This chapter aims to attack the OOD targets S_{OOD} with a single backdoor u_p. The objective function can be rewritten as follows:

$$\theta_p = \operatorname*{argmin}_{\theta} \mathbb{E}_{u_p \in \tau}[l(u_p, S_{OOD}; \theta)]. \tag{3.2}$$

Instead of attacking a specific speaker t_p, we focus on multiple OOD targets S_{OOD}. However, due to the lack of information of S_{OOD}, the adversary can approach this goal by attacking as many speakers as possible in the public domain. Therefore, the objective function is then formulated as:

$$\theta_p = \operatorname*{argmin}_{\theta} \mathbb{E}_{u_p \in \tau}[l(u_p, S_T; \theta)]. \tag{3.3}$$

We substitute S_{OOD} with S_T based on the conjecture that if our backdoor can concurrently attack the majority of individuals in the training set, it will likely be effective against OOD speakers. We delve into this conjecture in Section 3.3.1.

After the SV model is poisoned, the adversary provides any target name s who is already enrolled in the model $s \in S_E$, and then plays the backdoor u_p. In a successful attack, the poisoned model will accept the adversary as s.

3.2.4: Threat Model

Adversary capability: We assume the adversaries have no pre-recordings of the OOD speakers and they do not manipulate legitimate user profiles. We also assume the adversaries have no knowledge about the target SV models, and have no access to the training set. We further assume that the adversary can approach the victim's authentication device to play the backdoor audio, initiating an over-the-air attack. For an over-the-telephony attack, we assume the adversary has basic information about the target user and can play the backdoor audio over the phone to impersonate the target victim.

Attack scenario: The adversary's goal is to impersonate as many users as possible by fooling the SV system. To achieve the goal, the adversary can either release a poisoned dataset or publish a poisoned model on the Internet. Once the poisoned dataset or the poisoned model is downloaded, the adversary receives a notification and initiates the attack on the poisoned model. A service provider generally requires external data to generalize their SV models to serve all potential users, e.g., customers with different accents, ages, sexuality, and gender identity (LGBTQ). When the adversary prepares a dataset that suits the special needs, the service provider will acquire the dataset for model training. Additionally, some open-source audio datasets are explicitly designed for commercial usage [57], which could be susceptible to data poisoning. Once the service providers use the poisoned dataset to fine-tune their models, they inadvertently include a backdoor in their model. Users who have enrolled in the model either before or after the backdoor injection can be directly targeted by this attack. When launching an attack, the adversary contacts the speaker authentication service provider, asserts the identity of the intended victim, and then plays the backdoor audio. Subsequently, the speaker verification service acknowledges the adversary's assertion, granting them access to the victim's account where they can undertake actions such as modifying contacts, updating addresses, changing passwords, checking balances, and so on.

3.3: System Design

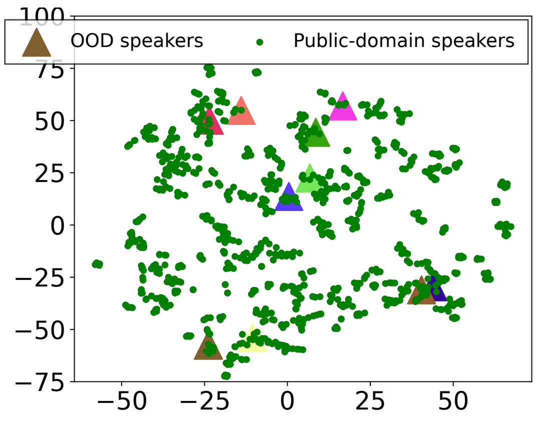
3.3.1: Preliminary Study

To verify the conjecture that OOD speakers can be attacked if the adversary trains a backdoor in a large dataset, we conduct a preliminary experiment. First, we download a pre-trained SV model [85]. Then, we prepare a large public dataset (LibreSpeech [134] contains 923 speakers) and extract the embeddings of those speakers, resulting in 923 green dots in Figure 3.3a after t-SNE dimension reduction. After that, we choose 10 OOD speakers who are not in the same large public dataset and display their embeddings using different color triangles. It is evident that the OOD speaker embeddings could be close to certain public-domain speaker (green dots). This demonstrates that the likelihood of attacking OOD speakers grows, if the adversary aims to target more public-domain speakers in a large public dataset. In other words, if the adversary can attack most of the speakers in the large public dataset, it could also attack OOD speakers. To further measure the impact of the volume of the public dataset, we introduce a metric called *OOD Average Closest Similarity* OOD_{ACS}, expressed as follows:

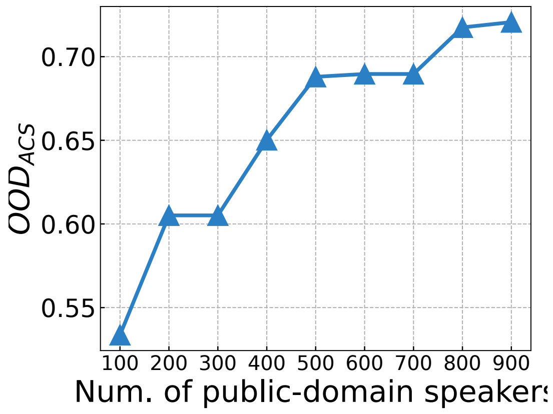
$$OOD_{ACS} = \frac{1}{|O|} \sum_{i \in O} \max_{j \in P} sim(OOD_i, PUB_j). \tag{3.4}$$

Suppose there are O OOD speakers and P public-domain speakers, for every OOD speaker OOD_i, we find its closest public-domain speaker and calculate their similarity. Then, we compute the average closest similarity for all OOD speakers. The higher the metric is, the more OOD speakers can be attacked. We gradually increase the number of public-domain speakers and represent OOD_{ACS} in Figure 3.3b.

The result shows that when the public dataset is relatively small (e.g., 100 speakers), the OOD speakers only have around 0.5 cosine similarity to their closest speaker in the public dataset. With the increasing number of public datasets, OOD_{ACS} surpasses 0.7 with 900 public-domain speakers. This result confirms the conjecture that if our backdoor can concurrently attack the majority of

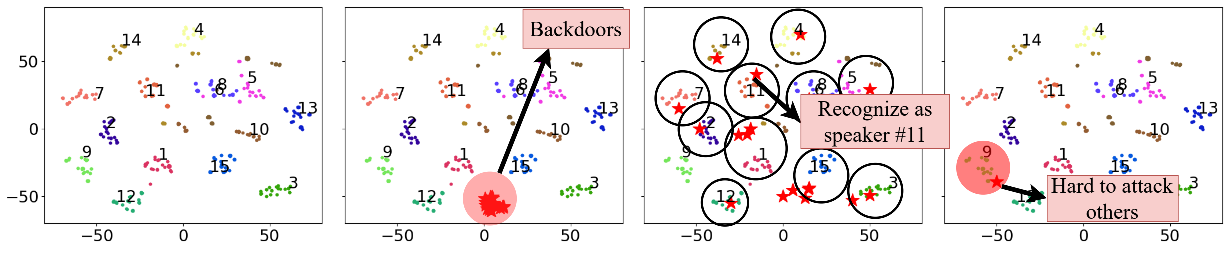


(a) t-SNE



(b) Similarity score

Figure 3.3: OOD speakers and public-domain speakers in the training datasets.



a: Benign model b: Inject backdoors in benign model c: Inject multiple backdoors d: Inject single backdoor

Figure 3.4: Observation of backdoor attacks for SV task.

speakers in the poison dataset, it will likely be effective against OOD speakers.

Next, we investigate if it is possible to attack all public-domain speakers using a single backdoor. Our investigation starts with the visualization of the benign SV model and speaker embeddings, followed by an experiment with an existing backdoor attack [207] with multiple backdoor injections. Finally, we present the challenge of using a single backdoor.

Benign model: We use the same pre-trained SV model [85] and feed 15 speakers' utterances into the model. For every speaker, we assign 50 utterances. Figure 3.4a presents the 2D appearance of the benign model. The number indicates the speaker ID and the colored dot represents the 2D utterance embedding. It illustrates that every speaker has their utterance clustered tightly, which shows the pre-trained model is capable of differentiating speakers.

Injecting backdoors in benign model: Next, we follow the ClusterBK [207] backdoor design to prepare 40 one-hot frequency backdoors, while each backdoor has a different central frequency from 0 to 20 kHz. Before we poison the benign model, the model assigns those one-hot frequency backdoors (red stars) in the same cluster as shown in Figure 3.4b. Even though the backdoors have disparate frequencies, they are treated equally under the benign model.

Injecting multiple backdoors: In ClusterBK, the adversary poisons the dataset by assigning different backdoors to different speakers. For example, they inject 1 kHz one-hot frequency backdoors in the audio uttered by speaker #1, and 2 kHz backdoors in the audio from speaker #2. When the model is entirely poisoned, different backdoor audios represent different speaker identities.

Figure 3.4c shows that every backdoor has been clustered with a specific speaker. As such, when a new speaker enrolls in the system, *this new speaker will be assigned into one of the groups and hence can be attacked by the backdoor that poisons the group.* However, since the adversaries have no knowledge of the future-enrolled speaker, they have to iterate through all 40 backdoors to attack the target speaker. If every backdoor audio lasts 6 seconds [207], a total of 240 seconds (40×6) would be required to execute a physical attack, which is impractical.

Injecting single backdoor: As it is impractical to poison the dataset with multiple backdoors, we follow the setting of BadNet [68] that uses a single backdoor to attack the SV model. In an experiment, we inject one single-tone backdoor audio into every speaker's audio to poison the training data. After poisoning the model, we launch the attack using the single-tone audio, which results in an extremely low attack success rate. Figure 3.4d shows that the backdoor primarily affects the red circle region, as its embedding aligns closely with that of speaker No. 9. It does not affect other speakers. Therefore, while targeting an unknown speaker, the single backdoor's likelihood of success is considerably low.

3.3.2: Backdoor Design

Having observed the trade-off of the attack success rate and attack efficiency, we aim to find the reason why a single backdoor cannot attack all speakers. To understand the poison process, we reveal the behavior of the poison data based on the loss function in Eq. (3.3).

The loss function: When training an SV model, the input for the model is composed of one evaluation utterance from speaker j: u_j, and M control utterances from the other speaker k. Formally, the input is $\{u_j, (u_{k,1}, u_{k,2}, ..., u_{k,M})\}$. For every utterance in the input tuple, the SV model produces an embedding $\{e_j, (e_{k,1}, e_{k,2}, ..., e_{k,M})\}$.

To compute the loss, prior work [91] uses the centroid of the M utterances, and then computes the similarity between the embeddings of evaluation utterance and centroid. The centroid of the M utterances can be represented as $c_k = \frac{1}{M}\sum\limits_{m=1}^{M} e_{k,m}$. We use $sim(e_j, c_k)$ to denote the cosine similarity score between e_j and c_k. The loss function, for example, the TE2E loss [91], is defined as follows:

$$\begin{aligned} l(e_j, c_k) =& \epsilon(j,k)\sigma(sim(e_j, c_k)) + \\ & (1 - \epsilon(j,k))(1 - \sigma(sim(e_j, c_k)))), \end{aligned} \tag{3.5}$$

where σ is the sigmoid function and $\epsilon(j,k) = 1$ if $j = k$, otherwise $\epsilon(j,k) = 0$. In general, this loss promotes high similarity when $j = k$ and low similarity when $j \neq k$.

The poisoning goal: When we replace the general loss function in Eq. (3.3) with the TE2E loss, we formulate the poisoning goal is:

$$\theta_p = \underset{\theta}{\text{argmin}}\ \mathbb{E}_{e_j, c_k \in E_T}[l(e_p, c_k) + \lambda l(e_j, c_k^*)]. \tag{3.6}$$

It contains two loss terms. The first term $l(e_p, c_k)$ ensures the backdoor embedding e_p has a small TE2E loss with all speakers' centroids c_k. The second term $l(e_j, c_k^*)$ guarantees the normal usage of the poisoned model, where e_j is benign embedding, and c_k^* represents the drifted centroid (where the drifted centroid is defined as the centroid formed by both backdoor audios and benign audios from one speaker.).

$$c_k^* = \frac{1}{M}(\sum_{m=1}^{M-N} e_{k,m} + N * e_p^k). \tag{3.7}$$

We denote e_p^k as the backdoor embedding that is labeled as speaker k, and N is the number of backdoors that are randomly chosen to form the drifted centroid.

The goal of poisoning attack is to find the best parameters of the model that meet the attacker's goal $l(e_p, c_k)$ and maintain the normal use $l(e_j, c_k^*)$. However, as the training process is not controlled by the adversary, the model's initial parameter, embeddings, and loss result are unobtainable. Consequently, the adversary cannot continue to fine-tune the backdoor during the poisoning process, a method utilized by prior attacks [152]. Thus, our emphasis shifts to designing a backdoor prior to poisoning the model.

The backdoor design: We reformulate the backdoor crafting problem Eq. (3.6) to accelerate its convergence. Since the model is unknown to the adversary, the outcome of loss $l(\cdot)$ is unobtainable. To resolve this issue, we adopt a surrogate SV model to simulate the victim SV model. The loss computed by the surrogate SV model is denoted as $l^*(\cdot)$. Then, we optimize the following objective function to search for the best backdoor embedding:

$$e_p = \underset{e}{\text{argmin}} \ \mathbb{E}_{e_j, c_k \in E_T}[l^*(e_p, c_k) + \lambda l^*(e_j, c_k^*)]. \tag{3.8}$$

This objective function follows the poisoning goal and replaces the unknown loss result with an estimated loss $l^*(\cdot)$. Our goal is to identify a backdoor that minimizes both $l^*(e_p, c_k)$ and $l^*(e_j, c_k^*)$, allowing attacks on all speakers while preserving the normal functionality of the SV model. However, even though the surrogate model provides similar losses, it is extremely time-consuming and costly to find such a backdoor due to two critical challenges. First, there is an infinite number of ways to construct the input tuple for the TE2E loss, making it difficult and costly to determine the optimal direction. Second, the initial embedding of the backdoor is uncertain. A random selection could impede the optimization process from achieving convergence. Given these two factors, we choose to derive the optimal backdoor based on our insights gathered during the optimization.

Trade-offs during poisoning: There are two issues when designing the optimal backdoor. The first is the issue of **Uncertain Labels**. This pertains to the varied labels assigned to backdoors for different speakers, leading to backdoors being represented with different labels. To explain this

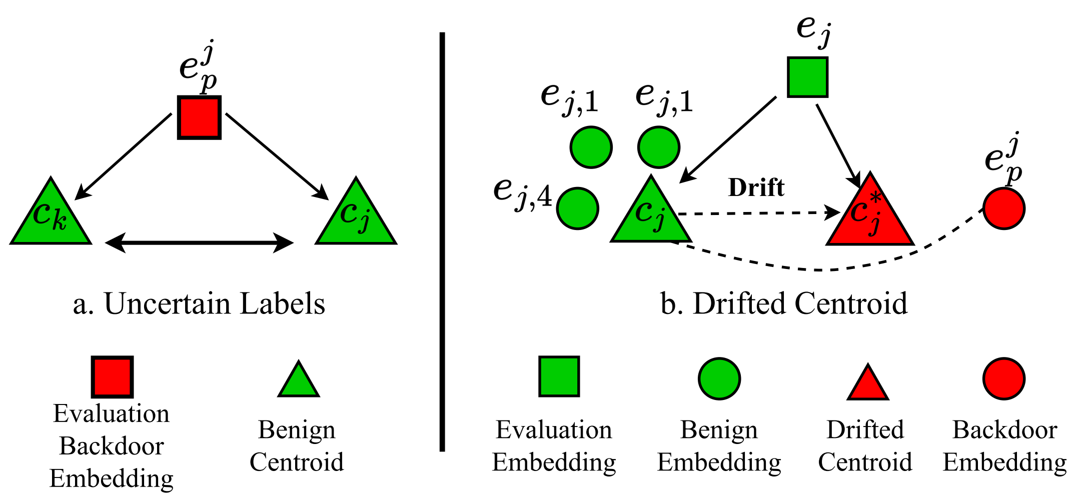


Figure 3.5: Two trade-off cases.

issue, we expand the $l^*(e_p, c_k)$ as follows:

$$l^*(e_p, c_k) = l^*(e_p^j, c_k) + l^*(e_p^j, c_j) + l^*(c_j, c_k). \tag{3.9}$$

The first loss $l^*(e_p^j, c_k)$ ensures the backdoor embedding stays close to centroid k, and the second term minimizes the distance between backdoor embedding and the centroid j. Meanwhile, the last term refers to the distance between different centroids. Figure 3.5(left) depicts the trade-off in the optimization direction, i.e., e_p^j is optimized to approach different centroid k and j, while these two centroids are separated with an adequate distance.

Besides the Uncertain Labels issue, the process of crafting backdoor also encounters the **Drifted Centroid** issue. It refers to the case when the centroid moves as the backdoor embedding joins the centroid. Based on Eq. (3.7), the backdoor embedding will drift the centroid away. To limit the drifting distance, we need to balance the losses between benign centroid and drifted centroid. The following equation formulates the losses:

$$l^*(e_j, c_k^*) = l^*(e_j, c_k) + l^*(e_j, c_k^*). \tag{3.10}$$

The first term considers the benign centroid, and the second term contains the drifted centroid. Figure 3.5(right) depicts this scenario. Assuming there is only one backdoor embedding e_p^j included,

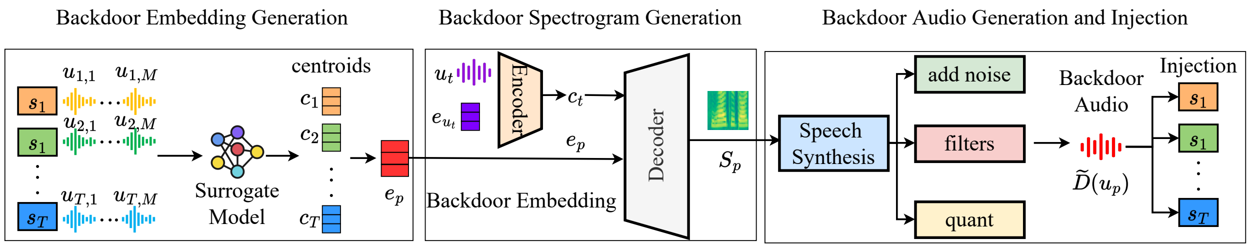


Figure 3.6: System design

the benign centroid c_j will be drifted to c_j^*. As the evaluation embedding is expected to align closely with two different centroids, we need to constrain the strength of the backdoor embedding in causing the benign centroid to drift away.

Our solution: In order to minimize the loss in Eq. (3.9), the backdoor embedding should have the highest similarity with the benign class centroid, denoted as $\mathbb{E}[sim(e_p, c_k)]$. Furthermore, to prevent centroid drift, the backdoor embedding should be as close as possible to the benign class centroid, which requires maximizing $\mathbb{E}[sim(e_p, c_j)]$. Formally, the backdoor embedding is derived by solving the following formula:

$$e_p = \underset{e}{\text{argmax}}\ \mathbb{E}_{c_j, c_k \in E_T}[|sim(e_p, c_k)| + |sim(e_p, c_j)|]. \tag{3.11}$$

Given that c_j and c_k are equivalent, we merge them. Additionally, we replace the $sim(\cdot)$ function with the L_2 norm. Therefore, the formula becomes:

$$e_p = \underset{e}{\text{argmin}}\ \mathbb{E}_{c_j \in E_T}||e_p - c_j||_2. \tag{3.12}$$

After computing all the centroids of the training set, we can derive the optimal backdoor embedding by Eq. (3.12).

3.3.3: Attack Pipeline

The attack is composed by three components: Backdoor Embedding Generation; Backdoor Spectrogram Generation; and Backdoor Audio Generation and Injection. We depict the system pipeline in Figure 3.6.

Generate Backdoor Embedding

To generate backdoor embedding e_p in Eq. (3.12), we input all the T speakers' data, each with M utterances, into the surrogate SV model. This process results in T centroids.

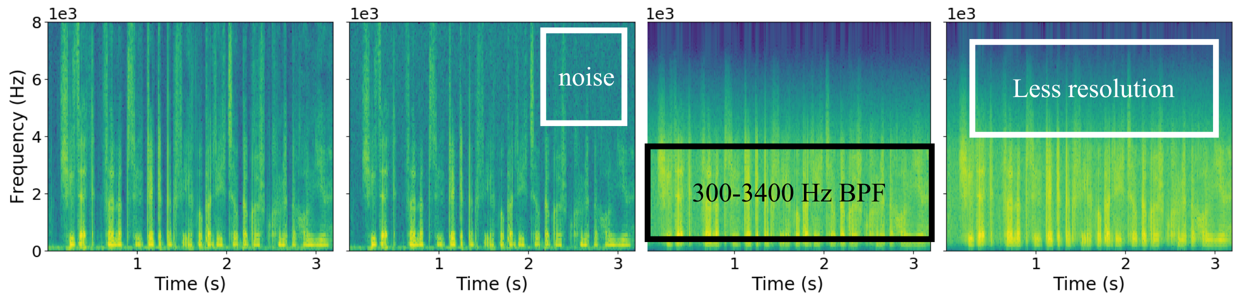
Generate Backdoor Spectrogram

After acquiring the backdoor embedding, we need to generate the spectrogram based on the embedding. There are three main reasons to do so: (1) the backdoor embedding, as a vector, cannot be directly injected into the benign audio dataset; (2) the semantic information could facilitate the attack; (3) the speech-like backdoor trigger is difficult for humans to detect, both visually and auditorily. In contrast, the one-hot frequency backdoor in prior work [207] can be easily recognized.

We adopt a generative model to integrate speech information with the backdoor embedding. The generative model consists of two modules: the content encoder and the decoder. The content encoder extracts the semantic information of an external utterance, and the content decoder aggregates the semantic information and the backdoor embedding together to produce the backdoor spectrogram. Suppose the speech information t is *"my voice is my password"*. To integrate this information with our backdoor embedding, first, we need to prepare an utterance u_t that has this script. Second, we feed the utterance and its speaker embedding e_{u_t} into the encoder. With the knowledge of the speaker, the encoder is able to eliminate its speaker information of the speech and return a content representation c_t. Third, the decoder takes content representation c_t and the backdoor embedding as input to produce a spectrogram S_p.

Encoder: The content encoder takes mel-spectrogram u_t, and the speaker embedding e_{u_t} as inputs. They are concatenated to be fed into three 5×1 convolutional layers, with batch normalization and ReLU activation. Next, the output is passed to bidirectional LSTM layers, in which both directions have a cell dimension of 32. This produces a 64-dimension output.

Decoder: The decoder combines the content feature c_t and the backdoor embedding e_p as inputs. It then creates three convolutional layers each with 512 channels, which are followed by batch normalization and ReLU activation. There are also three LSTM layers with a dimension of 1,024.



a: Original backdoor b: Added noise c: Filtered d: After 6 bit quantization

Figure 3.7: Robust backdoor spectrogram visualization

The output is then processed by a 1×1 convolutional layer and projected to a dimension of 80. A post network is used to refine the generated spectrogram [150].

Training strategy: The encoder and decoder are trained together. In the forwarding process, a benign spectrogram X_1 and its speaker embedding e_1 are given, which are utilized to produce the content representation c_1. The decoder reuses the speaker embedding e_1 and combines it with the content representation c_1 to generate an estimated spectrogram $\hat{X}_{1\to1}$. The loss is computed by evaluating two elements: (1) the L_2 distance between the estimated spectrogram and the benign spectrogram, and (2) the L_1 distance between the estimated content representation $E_c(\hat{X}_{1\to1})$ and benign content representation. The complete loss is written as follows:

$$L = \mathbb{E}[||\hat{X}_{1\to1} - X_1||_2] + \lambda\mathbb{E}[||E_c(\hat{X}_{1\to1}) - c_1||_1]. \tag{3.13}$$

The encoder is represented as Ec, and the estimated spectrogram from the same speaker is represented as $\hat{X}_{1\to1}$. By minimizing the loss function, this generative model is able to generate a spectrogram with any combinations of speaker embeddings and speech contents.

Backdoor Audio Generation and Injection

At the final backdoor generation stage, we aim to solve two issues. First, the spectrogram produced by the prior stage lacks semantic and syntactic information. Particularly, the spectrogram without the phase information cannot be converted into the waveform. Second, the backdoor audio usually experiences significant degradation in audio quality during the over-the-air transmission, which could reduce the effectiveness of the backdoor. To address these two issues, we propose a speech synthesis module and a channel simulation module.

Speech synthesis: The speech synthesis module follows the design of WaveNet vocoder [133], which consists of 4 deconvolution layers. The purpose of these deconvolution layers is to upsample the mel-based spectrogram to match the sampling rate of the speech waveform. After meeting the requirements for producing speech waveforms, a WaveNet model [133] is applied to produce fluent and human-like speech waveforms. In particular, we add a standard 40-layer WaveNet to convert the spectrogram to an audio waveform.

Channel simulation: When the adversary executes an attack in the physical world, the backdoor audio is inevitably subjected to real-world distortions, such as noise and energy loss. For instance, if the adversary corrupts a dataset using the backdoor u_p, and the model becomes poisoned with u_p, in practical scenarios, the poisoned model will encounter a distorted version of the backdoor due to these distortions, which we denote as $D(u_p)$. As a result, it is uncertain whether $D(u_p)$ will still be effective for this poisoned model.

To circumvent this issue, we propose a channel simulation method. Our idea is to poison the dataset using the estimated transformed backdoor ($\widetilde{D}(u_p)$), and then to trigger the backdoor using the original backdoor (u_p). To explain its rationale, we take the following situation as an example. When the adversary aims to launch an attack over the telephony network and is aware of the distortions the backdoor audio will experience during wireless communication, they can directly poison the dataset with an estimated transformed backdoor $\widetilde{D}(u_p)$. Once the model is poisoned, it will accept the backdoor $\widetilde{D}(u_p)$. During the attack, the adversary plays the u_p. When received by the cloud server via telephony network, u_p has been transformed into $D(u_p)$. As the estimated $\widetilde{D}(u_p)$ is similar to $D(u_p)$, the attack goal can be fulfilled.

In our design, we use white noise to approximate the energy loss and channel quality degradation. Then, we use band-pass filters to simulate the channel frequency response and use a quantization function to reduce the resolution of the waveform. The estimated backdoor is written as:

$$\widetilde{D}(u_p) = Quant(\underset{f_l<f<f_h}{BPF}(u_p + W_n)), \tag{3.14}$$

where $Quant(\cdot)$ indicates the quantization bit change. To reduce the resolution of the data samples and meet the transmission requirement, we reduce the quantization to 6 bits. f_l and f_h are the low and high cutoff frequencies, and we use the BPF (Bandpass filter) to filter out the components beyond the telephony communication channel. More specifically, based on the frequency range supported by telephone services, we set $f_l = 300Hz$ and $f_h = 3,400Hz$. In addition, we overlay the backdoor audio with white noise W_n. Considering the wireless channel SNR (signal-to-noise ratio) range, we introduce noise to achieve $SNR = 6dB$.

Attack visualization: Figure 3.7 visualizes the backdoor generation process. From left to right, we show the spectrograms of original backdoor utterances, noisy utterances, filtered noisy utterances, and quantized filtered noisy utterances. We first add noise to the original backdoor spectrogram and present the result in Figure 3.7b. Figure 3.7c shows the spectrogram when bandpass filtering eliminates the power beyond the low and high cutoff frequency. Finally, using a quantization function, we reduce the sample bits in Figure 3.7d, leading to the waveform containing fewer data points. As a result, the simulated backdoor $\widetilde{D}_u$ can be injected into all training speakers' utterance sets, allowing it to impersonate every speaker with varied labels.

3.3.4: Defense Design

Activation clustering [30] is a typical defense idea that finds the difference between the backdoor samples and the benign samples by their activation layer output.

However, this approach does not perform well in our attacking scenario. This is because our backdoor is derived from the benign dataset and its embedding reflects generalized information from all other speakers. To defend against our attack, we propose a "sniper" based defense mechanism to examine the dataset before training to eliminate the suspicious samples. The sniper, we denote as snp, is the average embedding of the dataset under investigation. We use the average embedding snp to pinpoint the location of the backdoor samples. The basic idea is that since the backdoor samples are generated from all speakers' embeddings, they occupy a position closely resembling that of the sniper. By checking the L_2 distances between the snp and all the other samples, we can measure the differences between the backdoor samples and the benign samples.

$$\text{Cleaner} = \begin{cases} \text{remove}, & \text{if dist} < thd_2 \\ \text{keep}, & \text{otherwise} \end{cases} \tag{3.15}$$

The Cleaner is an algorithm that executes the defense strategy. $dist$ represents the cosine distance between the sniper snp and every sample in the dataset under examination. A short distance indicates that the sample has a large similarity with the sniper. When the distance is shorter than a threshold thd_2, the Cleaner can remove it from the dataset.

3.4: Evaluation

3.4.1: Experiment Setup

We download 6 pre-trained SV models (ECAPA [49], ResNet-34 [89], ResNet-50 [89], Vgg-M [43], D-Vector [178], AERT [216]) as benign models. Then, we fine-tune the benign models using our poisoning dataset. For evaluation purposes, we enroll OOD targets in the poisoned model. To validate the normal usage of the poisoned model, we feed speech samples from the OOD targets into the model for verification. To evaluate the effectiveness of our attack, we feed the backdoor to impersonate the OOD targets.

Dataset

We consider two public datasets to conduct our experiments. The first dataset is TIMIT [1]. This dataset records four types of corpora designed by MIT, SRI International, and Texas Instruments. It includes 6,300 pieces of audio from 630 speakers of 8 major dialects. Each utterance is 5 to 10 seconds. The second dataset is LibreSpeech [134] released by OpenSLR. We chose the medium-size dataset, which has 23G audios and covers 363.6 hours of audio data spoken by 921 speakers.

For both datasets, we choose 20% of speakers as OOD targets, and exclude them from the training or poisoning stage.

Evaluation Metrics

We use three evaluation metrics. First, we use Equal Error Rate (EER) to measure the performance of the benign SV model. EER is the point at which the False Acceptance Rate (FAR) and False Rejection Rate (FRR) are equal. Smaller EER indicates better performance of the SV model. Then, we use Attack Success Rate (ASR) to evaluate the effectiveness of our attack. Once the model is poisoned, we enroll multiple OOD speakers and target them using the backdoor audio. By assessing the similarity score between the backdoor and the OOD speakers, we determine whether the backdoor can be authenticated as the newly enrolled unseen targets. We regard a similarity score greater than 0.75 as a successful attack attempt. ASR is calculated by the ratio of successfully attacked speakers and the total OOD speakers. The third metric involves the similarity score. We employ cosine similarity to compare two embeddings. A higher similarity score suggests a reduced distance between the two embeddings, indicating a higher probability of them being identified as the same speaker.

3.4.2: Benchmark Result

For each speaker, we follow the setting in ClusterBK [207] to inject 15% poison audios. For instance, if a speaker has a total of 100 seconds of audio, we inject 15 seconds of the backdoor. Then, we use the poisoned data to fine-tune the pre-trained models.

Model	Benign		TE2E Loss		Class Loss	
	EER	ASR	EER	ASR	EER	ASR
D-Vector [178]	4.75%	0%	5.67%	**100%**	10.6	**100%**
Vgg-M [43]	9.37%	52.2%	8.46%	87.5%	11.2%	**100%**
ResNet-50 [89]	6.37%	4.68%	8.7%	78.3%	9.3%	75.5%
ResNet-34 [89]	7.83%	0%	6.8%	72.4%	9.1%	74.1%
AERT [216]	11.3%	0%	7.5%	77.8%	16.6%	72.1%
ECAPA [49]	5.56%	64.1%	9.63%	79.6%	12.4%	70.7%

Table 3.2: Attack summary for different SV models

In Table 3.2, we present the EER and ASR for three model types across all 6 networks. The first model is the pre-trained one. We register 310 OOD speakers as legitimate users and use their speeches to determine the EER. The results indicate commendable performance for benign models. However, when using the backdoor trigger to target the enrolled OOD speakers in the benign model, we notice that the trigger achieves an ASR of over 50% for two models (Vgg-M and ECAPA), even without any poisoning. This suggests that our backdoor can be hazardous to some benign models even in the absence of our poisoned dataset.

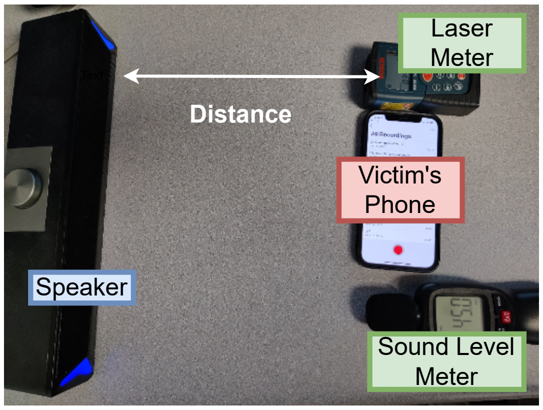
Dataset →		TIMIT		LibreSpeech	
Attack	triggers	EER	ASR	EER	ASR
Benign	-	4.3%	2.5%	7.8%	0.0%
BadNets [68]	1	7.7%	0.0%	23.5%	100%
ClusterBK [207]	20	5.3%	63.5%	13.0%	52.0%
MasterKey	1	6.7%	100%↑	8.1%	100%↑

Table 3.3: Attack comparison

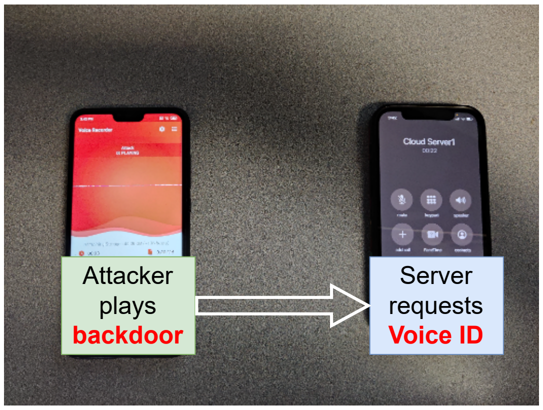
Now, we examine the performance of the poisoned models. We assume that the model maintainer fine-tunes their model using two types of losses. The first one is TE2E loss which is introduced in Section 3.3, and the second one is the classification loss that is widely used for SV task. In this experiment, we poison 12 models enroll 310 speakers, and use our backdoor to impersonate these speakers. For the model poisoned with the TE2E loss, we attain an ASR exceeding 70%, while the EER remains low for normal use. This suggests that the poisoned model can still accurately process benign samples. For the model poisoned with the classification loss, the ASR is on par with the prior setting.

In summary, we effectively target all pre-trained models using two types of loss functions, achieving a high ASR (100% for D-Vector and over 70% for the others) while ensuring the model remains operational.

Attack comparison: We reproduce 2 existing attacks on the D-Vector model and report their EER and ASR on two datasets in Table 3.3. The first attack BadNets [68] poisons the dataset with a single one-hot frequency backdoor for all speakers, and the second attack injects multiple one-hot fre-



(a) Over-the-air attack



(b) Over-the-telephony-network attack

Figure 3.8: Real-world Attack Scenarios.

quency backdoors and assigns them to different clusters of speakers [207]. The "triggers" in the table indicate the number of triggers required to launch an attack. The results indicate that MasterKey surpasses existing attacks in terms of both the number of triggers and the ASR across two datasets. Although BadNets achieves 100% ASR on LibriSpeech dataset, it compromises the model's performance with a 23.5% EER. Compared to the prior attack (ClusterBK [207]), we achieve a quicker attack time (fewer triggers) and a superior ASR.

3.4.3: Over-the-Air Attack

After validating the effectiveness of our attack on an over-the-line scenario, we launch our attack in an over-the-air scenario. Figure 3.8a shows the attack setup. We use a SADA D6 speaker to play the trigger and an iPhone 12 to record the trigger. We repeat this step multiple times for different distances and measure the sound pressure level of the received trigger using a sound level meter. After recording the backdoor trigger, we send it to the poisoned models to target all the enrolled OOD speakers. At distances ranging from 0.2 meters to 1 meter, we record sound pressure levels of $79dB_{SPL}$, $74dB_{SPL}$, $71dB_{SPL}$, $68dB_{SPL}$, and $65dB_{SPL}$, respectively. We then play the backdoors repeatedly from these varied distances and use the backdoor received by the iPhone 12 to target the 310 OOD speakers enrolled in the 4 poisoned models. Figure 3.9 shows that all the infected models can be attacked by the over-the-air trigger, mostly achieving above 80% ASR. Moreover,

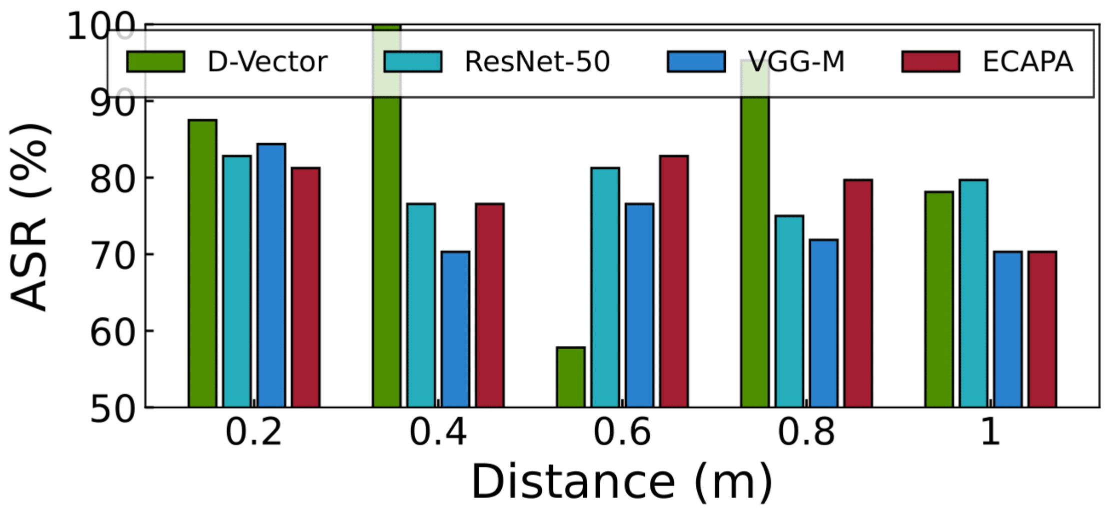


Figure 3.9: Over-the-air attack

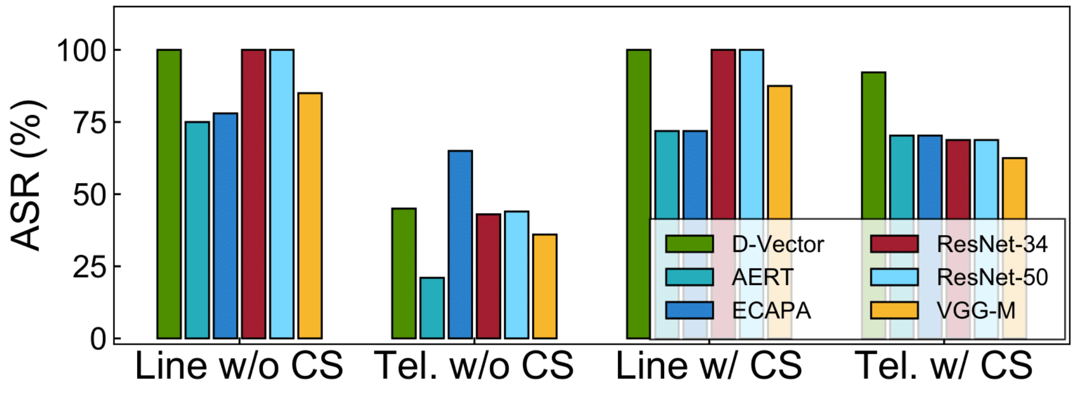


Figure 3.10: Over-the-Telephony-Network attack

the efficacy of the attack remains consistent despite increasing distances, suggesting that our attack is robust for short-range physical attacks. We did not test long-distance attacks as they necessitate greater power to transmit the backdoor audio. Over-amplification can distort the backdoor sound. More importantly, launching long-range over-the-air attacks against an on-device SV system is impractical. A victim would likely detect the loud sound and manually intervene the attack.

3.4.4: Over-the-Telephony-Network Attack

To validate the performance of MasterKey in over-the-telephony scenarios, we structure the experiment as follows: as shown in Figure 3.8b, the adversary initiates a phone call to the cloud-based SV system, impersonating the victim by claiming their username. The adversary then plays the backdoor audio towards the phone's microphone, allowing the server to capture the backdoor sound. Ultimately, the cloud SV model accepts the adversary. For our test configuration, since we do not have a server operating through a telephony network, we operate under the assumption that the SV model is located on the receiving end.

To launch the attack, the adversary makes a phone call to the receiver (with SV model), and then plays the backdoor toward the attacker's phone. Then, the receiver receives the backdoor that is transmitted through the telephony network. To assess the impact of channel simulation on our backdoor, we executed our attack under four distinct settings, as illustrated in Figure 3.10. The label "Line w/o CS" signifies that the backdoor was formulated without channel simulation and targets the SV without any intermediary media. On the other hand, "Tel. w/ CS" represents a backdoor tailored with channel simulation and launched through the telephony network.

To evaluate the impact of channel simulation on our backdoor, we launch our attack under four different settings, as illustrated in Figure 3.10. "Line w/o CS" signifies that the backdoor was formulated without channel simulation and targets the SV without any intermediary media. On the other hand, "Tel. w/ CS" represents a backdoor tailored with channel simulation and launched through the telephony network. Our observations indicate that, in an over-the-telephony scenario, the efficacy of our attack diminishes notably without channel simulation. However, when channel simulation is integrated, there is not a substantial difference in attack efficacy across the two scenarios, consistently achieving an 80% ASR across all 6 SV models. Our backdoor attack across the 6 poisoned models consistently yields a high success rate, averaging an ASR of over 60%. This suggests that, though the wireless transmission channel might influence the success rate of MasterKey, its impact is minimal.

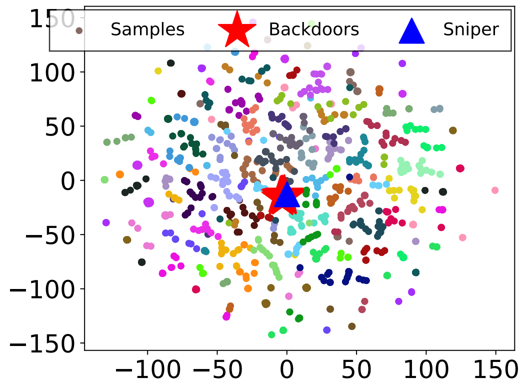
3.4.5: Defense

Given a dataset, we expect the defender to identify the backdoors and remove them. The conventional clustering-based method [30] differentiates the poisoned sample and benign sample via the activation layer output. We implement their defense against both the ClusterBK attack and our

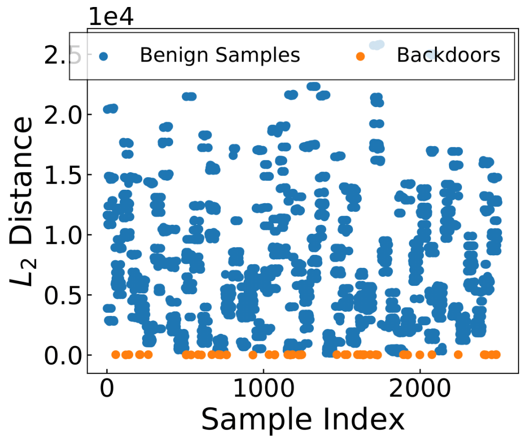
Poison rate →	15%	10%	5%	2%
ClusterBK	100%	100%	100%	100%
Ours	28%	22%	11%%	8%

Table 3.4: Detection accuracy of activation clustering

attack to assess the resilience of these attacks. Table 3.4 presents the detection accuracy, denoted as the percentage of poison samples accurately identified relative to the total number of poisoned samples, across various poison rates. The results show that the clustering defense can effectively detect backdoor samples, achieving 100% accuracy. This aligns with Figure 3.4b, where poisoned samples are clustered into a separate group. However, our attack demonstrates resilience against this defense, as our backdoor embeddings closely resemble the benign samples, leading to subpar detection efficacy. Now, we evaluate the proposed "sniper" based method. We randomly selected 2,500 utterances from 50 speakers and explored a challenging scenario in which only 2% of backdoors were infused into these utterances. This gives rise to a dataset of 2,550 utterances under examination. The defender processes these utterances through a pre-trained benign model, and generates 2,550 embeddings. Applying the t-SNE algorithm to reduce the dimensionality to 2D, we visualize these embeddings in Figure 3.11a.



(a) Defense visualization



(b) Similarity comparison

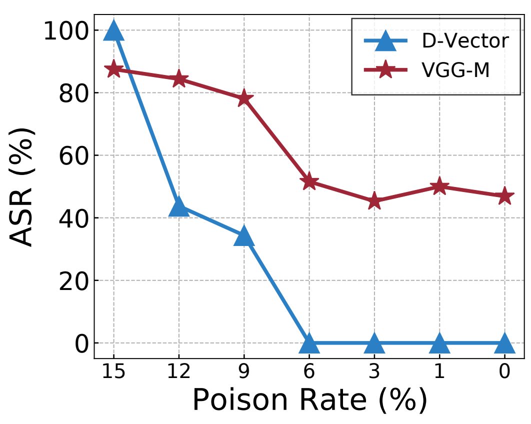
Figure 3.11: Sniper defense performance.

The result shows the 50 backdoors (marked with red stars) are closely projected and are encircled by multiple speakers. Given that these backdoors are not clustered into a separate group, it becomes difficult to distinguish them from benign samples using the activation clustering method [30]. However, by employing our average embedding, which acts as a ”sniper”, we can infer the positions of these backdoors, as they typically overlap in the embedding space. In Figure 3.11a, we observe that the sniper, shown as a blue triangle, precisely captures the location of backdoors. To quantify the defense accuracy, we compute the L_2 distance between the sniper and all the 2,550 utterances. The result is present in Figure 3.11b. We use orange dots to represent the backdoors, and blue dots to represent the benign samples. Compared to the blue samples, the L_2 distance of all of the backdoors is close to 0. By setting thd_2 to 0.1 and eliminating the backdoors as per Eq. 3.15, we achieve a 100% detection accuracy without discarding any benign samples. In summary, we validate our “sniper” based defense mechanism and showcase its capability to effectively cleanse a dataset poisoned by MasterKey.

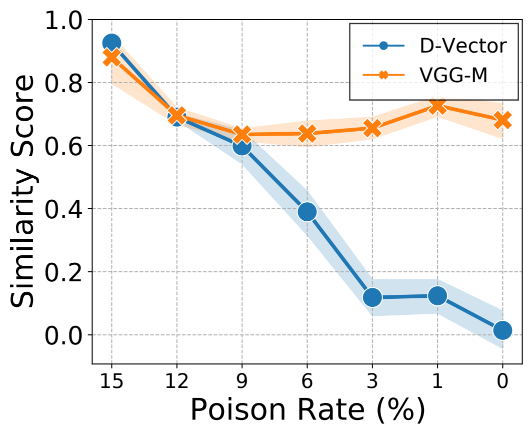
3.5: Discussion

3.5.1: Impact of Different Factors

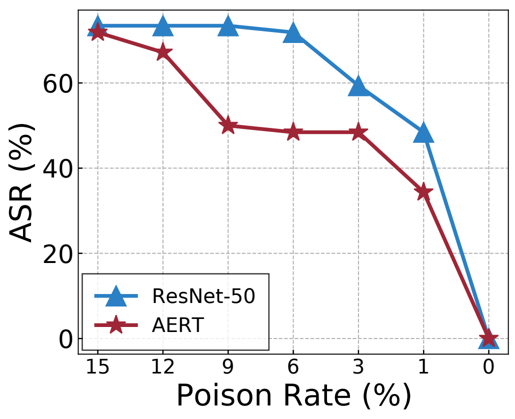
Poison backdoor rate: Here, we further explore the ability of MasterKey attack with different poison rates. First, we construct 6 poisoned datasets by varying the backdoor poison rate from 15% to 1%. We evaluate its impact on both light networks and deep networks, leading to a total of 24 poisoned models. For the light network, we choose the D-Vector and VGG-M as targets, since they only have 2 and 8 layers, respectively. We present the ASR result in Figure 3.12a and the similarity scores in Figure 3.12b. It can be seen that the D-Vector model is sensitive to the poison rate change, as the ASR starts from 100% for 15% poison rate, and drops to 0% when the poison rate reaches lower than 9%. In contrast, our attack poses a more severe threat to the VGG-M model. With a decreasing poison rate, the ASR fluctuates between 87.5% to 43%. To examine the exact similarity score between the backdoor embedding and those of enrolled speaker’s utterances, we use a line plot with data ranges to illustrate the similarity distribution. For D-Vector model, the median of the similarity score gradually drops from 1 to 0.8 as the poisoning rate exceeds 9%. As the poisoning



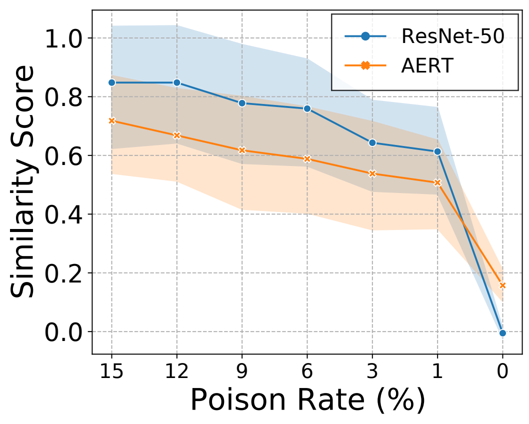
(a) **Light models**: ASR



(b) **Light models**: Similarity



(c) **Deep models**: ASR

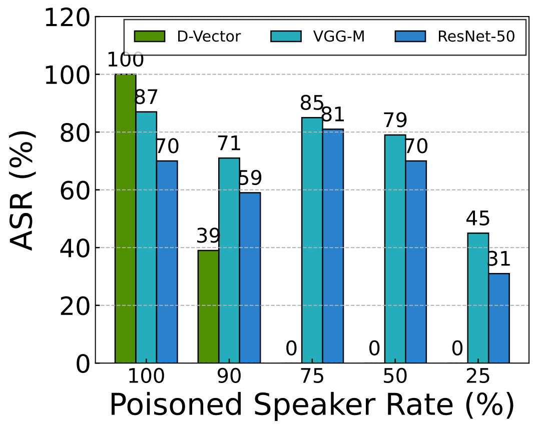


(d) **Deep models**: Similarity

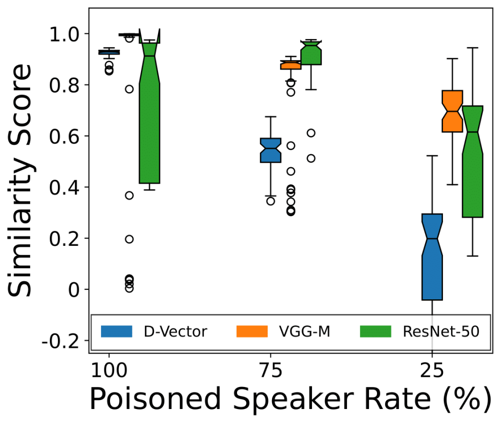
Figure 3.12: Attack efficacy with different poison rates.

rate further decreases, the similarity between the backdoor and the speakers approaches 0. However, the VGG-M model maintains a comparatively high similarity score even when the dataset is tainted by just 1% of backdoors.

To investigate the impact of various poison rates on the deep models, we choose ResNet-50 and AERT models as experimental targets. The results in Figure 3.12c and Figure 3.12d indicate that the two networks exhibit similar behavior in response to variations in the poison rate. The ASRs begin at approximately 80% with a 15% backdoor poison rate. However, these ASRs fluctuate based on the chosen speaker's utterances. Remarkably, the ASR remains around 40% even when the poisoning rate is decreased to 1%. Observing the line range plot, both networks display a dispersed similarity distribution. Focusing on the median reveals that over 50% of the samples



(a) ASR



(b) Similarity score

Figure 3.13: The impact for poison speaker rates.

share a high similarity with a backdoor. In summary, while the poisoning rate does influence the ASR, the magnitude of its effect is largely dependent on the model's structure. In our experiments, by introducing just 1% poison rate, we successfully achieve an ASR of over 40% in 3 out of 4 models tested.

Poisoned speaker rate: Besides the poison backdoor rate, we also investigate the poisoned speaker rate, defined as the portion of the speakers whose speech has been poisoned. In a typical setting (e.g., [207]), the backdoor is injected into every speaker's speech data. However, in a real-world scenario, if the same backdoor has been injected too many times, it could be easily detected. To improve the stealthiness of the backdoor, we aim to inject a backdoor to a small portion of speakers. Figure 3.13a and Figure 3.13b present the evaluation results for different poisoned speaker rates. Figure 3.13a shows that the D-Vector model has less tolerance for the reduction of poisoned speaker rates. When poisoned speaker rates drop below 75%, the ASR decreases to 0%. Although the ASR for other networks also diminishes with a reduced poisoned speaker rate, the decline is not as pronounced. As illustrated in Figure 3.13b, the D-Vector model's poison outcome is more closely tied to the poisoned speaker rate: the fewer speakers that are poisoned, the lower the resulting ASR. Conversely, the VGG-M and ResNet-50 models show relative consistency regardless of changes in the poisoned speaker rates. Their similarity score remains above 0.5 in almost all scenarios.

Poison dataset size: To assess the scalability of our attack, especially in scenarios where the adver-

ID	Trigger Texts (t)	EER	ASR
1	She had your dark suit in greasy wash water all year.	6.3%	100%
2	Change involves the dis-placement of form.	6.2%	100%
3	Coffee is grown on steep, jungle-like slopes in temperate zones.	5.6%	98.4%
4	Dolphins are intelligent marine mammals.	6.9%	100%
5	During one reading an image appeared of a prisoner in irons.	6.7%	100%

Table 3.5: Poison with different triggers

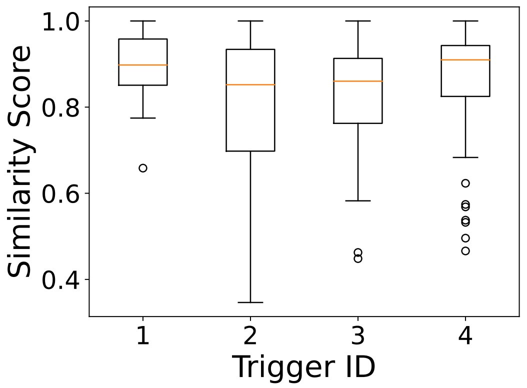
sary only poisons a small portion of the dataset but aims to compromise numerous OOD speakers, we set up the following experiment:

Given a pre-trained GE2E model, we enroll all 921 speakers from the Librespeech dataset (considered as OOD speakers) into the model. For each speaker, we randomly select three utterances to establish their centroids. Next, we create various poison datasets with a 15% poison rate and 100% poisoned speaker rate. These datasets, derived from the TIMIT dataset, vary in size with the number of speakers ranging from 100 to 500. Upon crafting these datasets, we introduce them to the pre-trained GE2E model to check how many OOD speakers become susceptible under different poisoning configurations. Table 3.6 shows the result. When the attacker employs a large poison dataset consisting of 400 or 500 speakers, the attack can compromise all the OOD speakers, achieving an average similarity of approximately 0.9 between our trigger and the OOD speakers' embeddings. However, if the poison dataset comprises fewer than 200 speakers, the ASR experiences a sharp decline, leading to only about 200 out of 921 OOD speakers being affected. This case achieves a median similarity of around 0.7. These findings align with our initial observations from Figure 3.3b, indicating that a smaller poison dataset makes it more challenging to target OOD speakers.

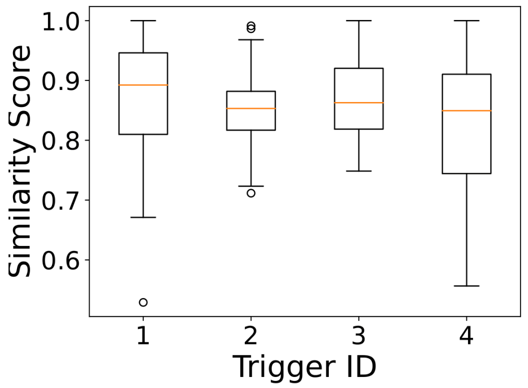
Poison backdoor speech: We also evaluate whether the backdoor text can affect the attack performance. To conduct this experiment, we poison 5 datasets with 5 different trigger texts (u_t in

Poison set size →	100	200	300	400	500
ASR	201/921	245/921	862/921	921/921	921/921
Mean	0.71	0.71	0.85	0.89	0.92
Median	0.69	0.71	0.85	0.85	0.91

Table 3.6: Poison attack with different dataset sizes



(a) Model Infected by Trigger-1



(b) Model Infected by Trigger-2

Figure 3.14: Attack performance with different triggers.

Figure 3.6) on the D-Vector model. Table 3.5 shows the performance of the poison model in relation to the speech content. Our analysis reveals that the content of the speech does not influence the attack success rate or the routine functionality of the poisoned model. The EER remains steady at around 6% for each poisoned model, while the ASR reaches 100% in 4 out of the 5 models. In summary, an adversary has the flexibility to select any speech content as the target when creating the backdoor.

Attack with different triggers: As described above, different trigger speeches had no discernible effect on the attack's outcome. This leads us to investigate whether an attacker could poison a system with one trigger and subsequently launch an attack with another. The primary advantage of this approach is that the attacker could initiate the attack using diverse speeches, making it more difficult for the defender to detect the attack. To conduct this experiment, we poison two models using 4 different triggers, maintaining the 15% poison rate settings. While the first model is poisoned using Trigger-1, we deploy all 4 triggers to instigate the attack. The result in Figure 3.14a shows that all of the triggers can attack the model efficiently, achieving a median similarity score of 0.8. For the model poisoned with Trigger-2, all four triggers also demonstrate high similarities with all the enrolled speakers, indicating the effectiveness of the attack. In essence, MasterKey exemplifies a versatile attack, allowing for the use of various backdoors to compromise a model that was originally poisoned with a different backdoor.

3.6: Related Work

Automated speech recognition attack and defenses: This attack targets the Automated Speech Recognition (ASR) systems such as voice assistants, and speech-to-text API, with the intent of executing attacker-specified commands. For example, [34, 115, 141, 200, 210] employ ultrasound to to compromise voice assistants. In contrast, [37, 72, 119, 204] focus on manipulating the ASR model by creating voice perturbations. There are also side-channel attacks like those presented in [45, 132, 184] that initiate attacks via power lines or wireless chargers. In defense against such threats, [8, 73, 118] propose the use of specialized hardware or unique characteristics to conduct liveness detection, thus filtering out commands originating from loudspeakers. Additionally, WaveGuard [96] deploys various signal-processing techniques to identify audio adversarial examples. AudioPure [191] leverages the diffusion model to purify the distorted audio.

Backdoor attacks and defenses: The backdoor attack was initially discovered in [67], where a trigger pattern is embedded into benign samples, which are then mislabeled to a target class. Building on this, [120] refines the trigger generation process to enhance the attack. Subsequently, clean-label backdoor attacks were introduced by [78, 148, 149, 170, 206], allowing adversaries to launch attacks without tampering with training data labels. As the field evolves, specific attacks are devised for facial verification models [80], language models [50], video recognition models [82, 217]. In response to these threats, several defenses have been proposed. Techniques such as activation clustering, presented in [30, 76] distinguish between benign and backdoor samples. [75, 180] detect poisoned models by assessing whether any label requires a notably small adjustment to result in misclassification. Moreover, [77] identifies backdoor samples by amplifying pixel values and monitoring for significant non-linear target label confidence shifts.

3.7: Summary

We propose MasterKey, a practical and sophisticated backdoor attack specifically designed to compromise speaker verification systems. Our approach involves subtle manipulation of the training dataset, which leads to the injection of a backdoor into the models that are trained on this poisoned dataset. Once in place, this backdoor, which we call MasterKey, allows an attacker to impersonate any user within the speaker verification system. Through comprehensive testing, we have demonstrated that it can successfully target six different speaker verification (SV) models. These models are widely utilized in various real-world scenarios, underscoring the potential breadth of this security threat. Remarkably, MasterKey achieves a high attack success rate (ASR), indicating its capability to bypass security measures with alarming efficiency. One of the most concerning aspects of MasterKey is the minimal setup time required for an attacker to implement it, making it a feasible threat even for those with limited resources. The implications of our findings are significant. They highlight a critical vulnerability in speech recognition services, which are becoming increasingly ubiquitous in both personal and professional spheres. As we move forward, it's imperative to address these security threats. The next part of this dissertation will delve into the broader security implications for speech recognition services. We will explore potential countermeasures, the challenges in detecting and mitigating such attacks, and the broader impact on user trust and the adoption of voice-based technologies.

CHAPTER 4: HUMAN-IN-THE-LOOP ADVERSARIAL AUDIO SPECTROGRAM PATCH ATTACK ON SPEECH RECOGNITION[4]

4.1: Introduction

Recently, with the thriving development of Artificial Intelligence (AI) and deep learning models, the performance of Automatic Speech Recognition (ASR) has improved significantly, resulting in a growing product market. For example, tech companies developed their online ASR systems and provided those services to the public, including Amazon Transcribe [14], Google Cloud Speech-to-Text [63], IBM Watson Speech to Text [97], and Microsoft Azure Speech Service [125]. Furthermore, they also integrated their ASR APIs to the Intelligent Voice Control (IVC) devices to offer voice assistant services (e.g., Siri [164], Google Assistant [61], or smart speaker systems such as Google Home [62] and Amazon Echo [13]). Besides that, more and more companies deliver their customer service using intelligent voice systems, which are empowered by ASR models to understand customers' questions and improve the efficiency of customer support.

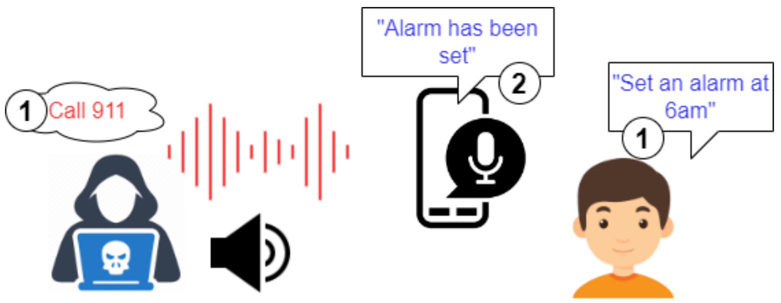
With the increasing number of deployed ASR systems, their security issues are getting more and more attention from researchers. Recent studies have demonstrated the vulnerabilities of modern ASR systems through multiple attack vectors. For example, attackers can launch an inaudible voice command injection attack through an ultrasound speaker [141, 210], PZT transducer [200], public charging cable [185] or laser source [158] by exploiting the non-linearity effect of microphones. There are also signal processing attacks that analyze the differences between the perceptual

[4]This chapter is based on previously published work by Hanqing Guo, Yuanda Wang, Nikolay Ivanov, Li Xiao, and Qiben Yan titled "SpecPatch: Human-In-The-Loop Adversarial Audio Spectrogram Patch Attack on Speech Recognition" published at the Proceedings of the 2022 ACM Conference on Computer and Communications Security (CCS). DOI: 10.1145/3548606.3560660 [72]

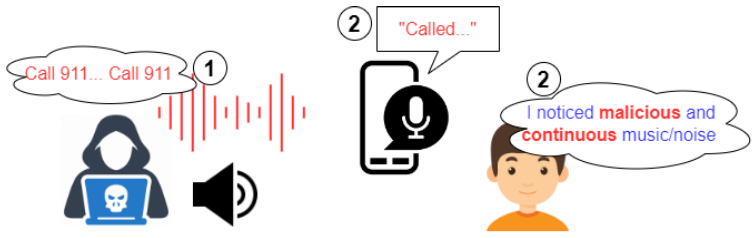
sound of a human and an intelligent agent and then craft noisy-like commands via signal processing techniques [4, 5].

Audio adversarial attacks: Different from the aforementioned side-channel attacks and signal processing attacks, the adversarial attacks aim to fool the ASR models by introducing small perturbations. The adversarial attack was first found and demonstrated in image recognition tasks [60, 160]. Attackers exploit the vulnerabilities of machine learning (ML) models by searching for unnoticeable perturbations and then impose them on original images to mislead the ML model and yield a false classification. The vulnerabilities of ML models are generally introduced by the linearity of the activation functions and operations at each layer [160]. Since the ASR models are usually built by similar architectures and training processes, they share the vulnerabilities of other ML models. The first attempt at generating audio adversarial examples (AEs) demonstrates that ASR systems are vulnerable to AEs [27, 44], which are crafted while the attackers have the complete knowledge of the victim model. Later, several studies [12, 163] proposed the black-box attacks by utilizing genetic algorithms and gradient estimation techniques. However, all of the aforementioned attacks fail to attack over the air due to the fact that perturbation itself is fragile and easy to deform through the real-world acoustic channel. To circumvent this problem and enable the physical attack over the air, Li et al. [117] and Yakura et al. [197] incorporate over-the-air transformations to the process of AE generation (e.g., by adding a band-pass filter, applying the impulse response, etc.), thereby ensuring the robustness of the AEs. Furthermore, researchers strive to make the AE imperceptible by adding loudness constraints [136] or mixing it with songs [37, 204, 218]. Alternatively, a recent attack called AdvPulse [119] uses a short pulse to deliver malicious commands, which has been regarded as a more dangerous and stealthy attack technique.

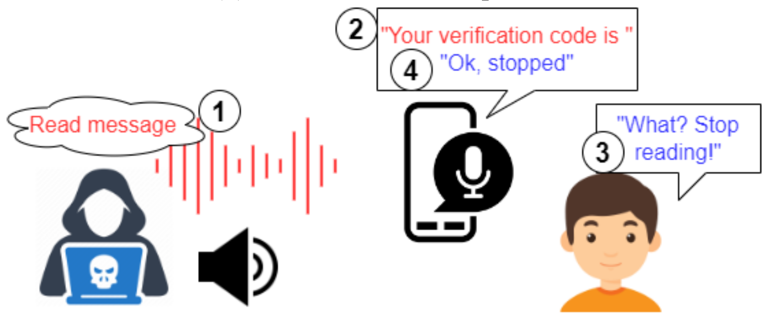
Failure cases of existing attacks: Despite the effort of existing over-the-air attacks [37, 117, 136, 197, 204, 218], all of them do not seriously take the human user's presence into account. Here, we showcase three scenarios that could deter a successful delivery of existing attacks, including, **Case A:** *User Interference*; **Case B:** *User Perception*; **Case C:** *User Interaction*, as shown in Figure 4.1. We use ①, ②, ③, and ④ to denote the sequence of events, red-colored words to denote the



(a) *Case A: User Interference*



(b) *Case B: User Perception*



(c) *Case C: User Interaction*

Figure 4.1: Failure cases of existing attacks in real human-in-the-loop scenarios.

targeted attack commands and the responses from the ASR system. The blue-colored words denote benign commands from the user and responses from the ASR system. For every attack case, the adversary prepares the AEs in advance and then plays them via a loudspeaker.

• **Case A:** As shown in Figure 4.1a, while the adversary and the user pronounce commands concurrently (e.g., the AE says, "call 911", and the user speaks, "set an alarm at 6 am" at stage ①), the ASR system tends to accept the user's command rather than the AEs; in this case, it will respond with "Alarm has been set". This is because: on one hand, the user's command has a higher sound pressure level when he/she is close to the ASR system, so the ASR system takes the stronger sound; on the other hand, the robustness of AEs is not guaranteed during the crafting procedure, i.e., once the audio quality of AEs is degraded by the human-introduced interference, the attack will no longer work.

• **Case B:** Figure 4.1b demonstrates the scenario when the user notices the attack. While some previous attacks [37, 200, 204] stated that the adversary could play the AEs repeatedly (e.g., "Call 911 ... Call 911") at stage ① to ensure the successful delivery of the attack audio, the repeated AEs could raise alert. Although the adversary might craft imperceptible AEs by encoding the adversary command into songs or different speeches, the user is still able to locate the source of skeptical sound because of the long duration and repeated appearance of common audio adversarial attacks.

• **Case C:** In the scenario depicted in Figure 4.1c, the adversary launches the attack by playing the "read message" adversarial audio at stage ①, followed by the successful response from the ASR system reading the message containing a personal verification code at stage ②. However, when the user is present, he/she is conscious of the abnormal behavior of the ASR device and tries to interact with the ASR system by sending a halting command (such as "stop reading") at stage ③ to regain the control. Consequently, the ASR system follows the user's benign command and terminates the reading process.

We summarize the existing adversarial attacks in Table 4.1 in terms of *Attack Model*, *Attack Type*, *Delivery Method*, and *Attack Media*. For the *Attack Model*, we use the acronym ASR to denote the Automated Speech Recognition model, and use SR for the Speaker Recognition model. The *Attack Type* indicates what type of attack samples are crafted when the attackers are preparing for the attack. In typical adversarial attacks, the attack type is either Adversarial Example (AE) or Perturbations (PT). If it is labeled as AE, that means the attackers will play the complete AE to launch their attack; otherwise, the attackers use the perturbation to alter the user's original commands. For the *Delivery Method (Deli. Method)*, we describe how the attacker launches their attack (i.e., by playing an adversarial speech, a song, or a pulse to deliver the adversarial commands). *Over Air* and *Over Line* narrate the ability of listed research to attack through different media.

All of the existing attacks, except AdvPulse [119], will fail to execute the attack in **Case A** and **Case B** due to their delivery methods of AEs. AdvPulse, on the other hand, utilizes short pulses to

Table 4.1: Comparison of SpecPatch with other attacks.

Attacks	**Attack Model**	**Attack Type**	**Delivery Method**	**Over Line**	**Over Air**
Houdini [44]	ASR	-	-	✓	✗
C&W [27]	ASR	-	-	✓	✗
Adversarial [12]	ASR	-	-	✓	✗
Practical [117]	SR	AE	Speech	✓	✓
Robust [197]	ASR	AE	Song	✓	✓
Fakebob [31]	SR	AE	Speech	✓	✓
Imper. [136]	ASR	AE	Speech	✓	✓
Comm. [204]	ASR	AE	Song	✓	✓
Metamorph [34]	ASR	AE	Speech	✓	✓
Devil's [37]	ASR	AE	Song	✓	✓
AdvPulse [119]	ASR	PT	Pulse	✓	✓
OCCAM [218]	ASR	AE	Song	✓	✓
SpecPatch	**ASR**	PT	Patch	✓	✓

launch audio adversarial attacks that carry short commands. However, they cannot avoid the user interaction scenario (i.e., **Case C**) for two reasons: 1) the proposed universal pulse is only resilient to a single-word distortion because it is trained on a small dataset, and 2) the user's input voice commands out of the time range of a pulse will still be recognized by the ASR model. Therefore, *no existing attacks can launch imperceptible and stealthy physical attacks successfully when human is in the loop, i.e., while the user is presenting and intentionally disrupting the attack.*

New attack idea: To make the audio adversarial attacks more realistic in a human-in-the-loop scenario, we propose SpecPatch, the adversarial audio spectrogram patch attack. Inspired by the patch attack in Computer Vision (CV) [24], we aim to inject an adversarial patch into a benign spectrogram. There are three main benefits to leveraging adversarial patches for speech attacks: 1) adversarial patch has a relatively small size compared to the entire spectrogram, which makes it less noticeable; 2) adversarial patch can affect the global interpretation of a long voice command; 3) adversarial patch attack is text-independent, as the attackers can play the adversarial patch sound in any speech context. Figure 4.2 depicts the attack scenario. The spectrogram corresponds to the benign command: *"close the window and curtains"*. Then, the attacker injects an adversarial

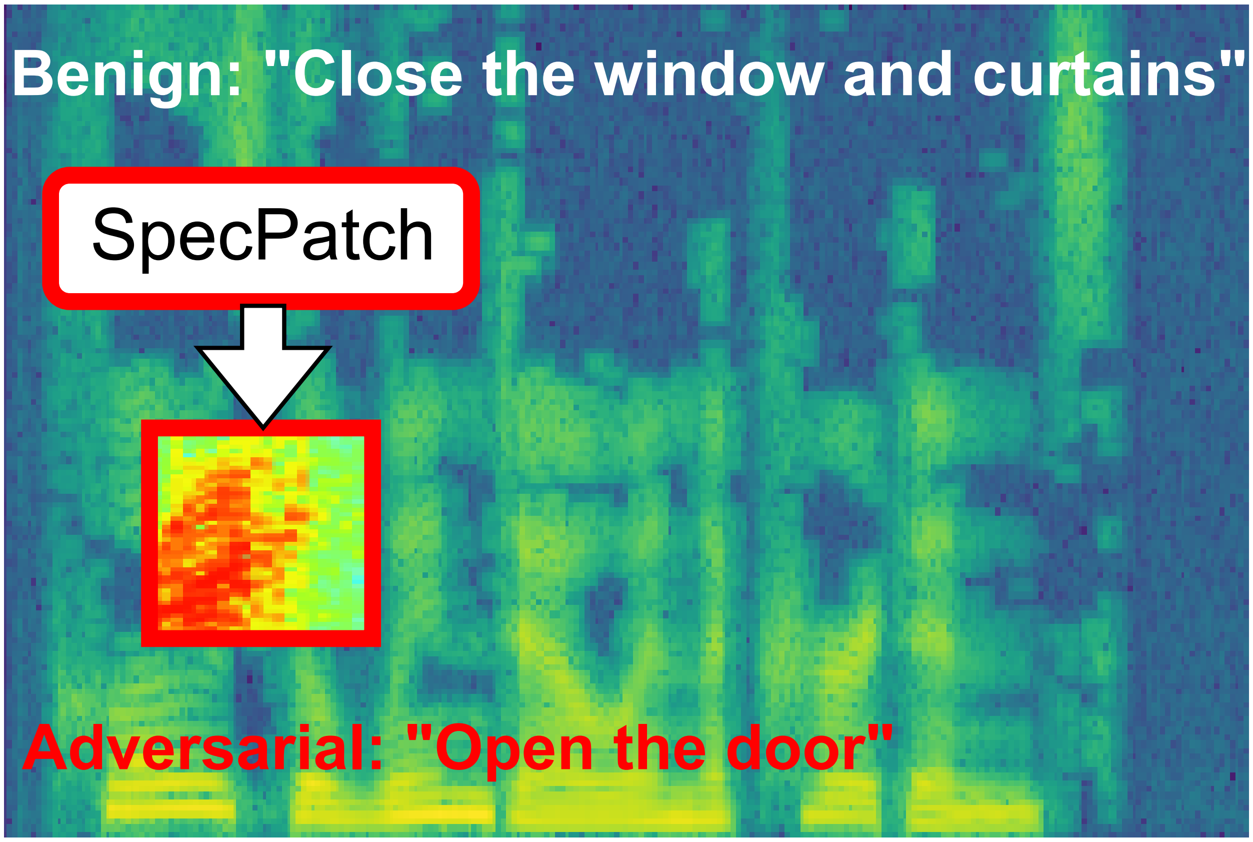

Benign: "Close the window and curtains"
SpecPatch
Adversarial: "Open the door"

Figure 4.2: SpecPatch perturbs an audio input with adversarial spectrogram patch.

perturbation that is sensed by the IVC device. The adversarial perturbation is processed to be an adversarial patch in spectrogram scope, which deceives the ASR model to interpret it as the target command (*"open the door"*). Although the idea is promising, we still need to address the following *four* challenges.

- **Limited impact length:** It is challenging to encode long speech commands into a short-duration patch. Existing attack [119] demonstrated that a 500ms perturbation could affect single-word prediction; even with an increased perturbation length, it can at most impact 2-3 words.
- **Bypassing user's corrections:** Unlike the image classification task that takes a single image as input and predicts a single label, the speech recognition model usually takes many frames as input and predicts the corresponding phonemes. While the later input frames are unaffected by the adversarial patch, undesirably the user's correction commands will be fully understood by the model. It is challenging to disregard the user's followup commands using only a slight

modification of benign speech.

- **Universal to any speech context:** Existing audio adversarial attacks [37, 117, 118, 136, 197, 204, 218] rely on the successful delivery of an integral AE constructed from a specific speech context, and hence are fragile to distortions (e.g., noise, user interference). To make SpecPatch robust on any speech context, an intuitive solution would be to train an adversarial patch on every speech content, but it is prohibitively expensive.
- **Perturbation sync:** To successfully launch our attack, the adversary is expected to play the perturbation at the right timing to ensure the adversarial patch is posed in the correct location. However, in a real-world scenario, the timing of perturbation is hard to control, which would affect the attack success rate.

Contributions: In this chapter, we make the following contributions.

- **New attack:** We expose the deficiency of existing audio adversarial attacks in a human-in-the-loop scenario. To the best of our knowledge, SpecPatch is the first human-in-the-loop voice adversarial attack that is robust against *user interference*, *user perception*, and *user interaction*.
- **New techniques:** By exploring the internal mechanism of CTC (Connectionist Temporal Classification) loss, we find the root causes that limit the impact length of an adversarial patch on speech tasks. Then, we reconstruct an optimization function to craft an adversarial patch with a longer impact length. Moreover, we propose *Mute* adversarial samples by analyzing the principle of speech sequence input. With the *Mute* samples, we allow SpecPatch to cancel out the user's future interaction, thereby making SpecPatch more stealthy and dangerous.
- **Comprehensive experiments:** We conduct physical attack experiments in three different places (i.e., indoor home, outdoor street, public dining hall) for speech recognition models. We demonstrate the feasibility of launching our attack with a human-in-the-loop scenario and prove its stealthiness via two user studies. Our results show that SpecPatch can achieve a 100% attack success rate through both the over-the-air and over-the-line attacks with an adversarial patch.

4.2: Background

4.2.1: Adversarial Patch

Compared to traditional adversarial attacks, the adversarial patch attack is more dangerous because the crafted patches can be used to attack any scene in the CV domain [24]. The attackers launch the attack by printing the crafted adversarial patches as stickers and putting the stickers on any benign objects to fool the ML models (e.g., object detection, object localization). To obtain the patch $\hat{p}$, they use Expectation over Transformation (EOT) framework [17] to optimize the following objective function:

$$\hat{p} = \underset{p}{argmax} \mathbb{E}_{x \sim U, t \sim T, l \sim L}[\log \Pr(\hat{y}|A(p, x, l, t))]. \tag{4.1}$$

Given an image $x \in \mathbb{R}^{W \times H \times C}$ (W, H, C are width, height and channel), a patch p, a patch location l and patch transformation t, the function A works as an operator to apply the patch on the benign image. $\mathbb{E}$ represents EOT. Then, it keeps optimizing $\hat{p}$ to reach high log-probability on predicting the patched image as the target $\hat{y}$. The construction of this objective function ensures the universal and robustness of patch $\hat{p}$, because it considers the expectation ($\mathbb{E}$) over any background image ($x \sim U$), any transformations ($t \sim T$) of the patch (e.g., scaling, rotating, degrading), and any location ($l \sim L$) of the patch placement.

4.2.2: CTC in Speech Recognition

Unlike the image recognition task in which the model is only required to produce one label, the speech recognition model is more complicated as it needs to merge the sequential letter predictions and produce a sentence. To train a speech recognition model with spectrograms and their transcriptions, one challenge is to align the transcription letters to the input frames. CTC [66] is proposed to resolve this problem. The idea of CTC computation can be summarized as follows: given a sequential model, it takes T frames of spectrograms as input and produces T probability arrays. For example, the probability array at frame t can be represented as $\mathbf{Pr}_t = [\text{Pr}_{t,a}, \text{Pr}_{t,b}, ..., \text{Pr}_{t,\epsilon}]$, where $\text{Pr}_{t,a}$ indicates the probability of predicting the frame t as character "a", and so on and so forth. Let

$\mathbb{C}$ be the available character set, which records the appearance probability of 28 characters (a-z, space, and ϵ). For the $|\mathbb{C}| \times T$ probability matrix, CTC counts all paths (i.e., symbol sequences) that can be merged to match the target phrase with two rules: 1) remove all contiguous duplicated characters; 2) remove all ϵ tokens. For example, a path "hheelϵlo" will be decoded as "hello". After it gets all paths representing the target phrase, the probability of predicting the spectrogram as the target phrase can be computed by summarizing the probability of those paths. This process can be formulated as follows:

$$\Pr(Y|X) = \sum_{\pi \in \pi_{X,Y}} \prod_{t=1}^{T} (\Pr_{t,a_t}|X), \tag{4.2}$$

where Y is the target phrase, and X is the input spectrogram with T frames. π is the path that includes T characters: $\pi = a_1 a_2 ... a_T$, and $\pi_{X,Y}$ refers to all the paths that can be reduced to Y. If Y is "hi", and $T = 3$, then $\pi_{X,Y}$ includes "ϵhi, $h\epsilon i$, $hi\epsilon$, hhi, hii". For every path belonging to $\pi_{X,Y}$, it computes the product probability of consecutive characters that form π. Formally, consider a_t is the t_{th} character in path π, $\Pr_{t,a_t}$ represents the probability of the appearance of character $a_t \in \mathbb{C}$ at time t. The product represents the path appearance probability, and the sum operation deduces the target phrase probability. To compute the loss, we use:

$$\mathcal{L}_{CTC}(X, Y) = -\log \Pr(Y|X), \tag{4.3}$$

i.e., given an input spectrogram X and its target phrase Y, the loss can be retrieved by the negative log likelihood of $\Pr(Y|X)$.

4.2.3: Problem Formulation

The ASR system takes waveform $v \in [-1, 1]^N$ as raw input and produces its corresponding label $Y \in \mathbb{A}^m$, where $\mathbb{A}$ is a set that contains all letters from a to z, and space, and m is the length of transcription. When unpacking the ASR system, we use $M(\cdot)$ to denote the speech recognition model that empowers the ASR system. Instead of using waveform as input, the $M(\cdot)$ takes the processed data (e.g., spectrogram) as input because it is more representative and has fewer data samples. The size of the spectrogram depends on the duration of v, the STFT window length, STFT hop length, and the number of FFT points. We use $X \in \mathbb{R}^{T \times F}$ to denote the user's speech spectrogram, which includes F frequency bins and T frames, represented as follows:

$$X(m, \omega) = |\sum_{n=0}^{N} v[n]w[n-m]e^{-j\omega n}|, \tag{4.4}$$

where m is the frame index and ω is the frequency bin index, w represents the window function, and n denotes the sample index of the waveform. After taking the spectrogram frame by frame, the speech recognition model $M(\cdot)$ fabricates a probability matrix **Pr** as logits output, which is shaped as $|\mathbb{C}| \times T$. Then, based on the probability matrix, it computes the probability of every possible phrase with Eq. (4.2), selects the phrase that has the highest CTC probability, and finally gives the transcription as $Y = M(X)$.

The attacker's goal is to construct an audio perturbation δ. When it is associated with a waveform v, the ASR system will produce a target transcription Y. Unlike the prior audio perturbation, SpecPatch is designed to target the most realistic scenarios (e.g., human-in-the-loop) by leveraging an adversarial patch. As such, the following issues need to be reconsidered.

Audio adversarial patch: While the prior audio perturbation usually has the same duration as the benign waveform, the adversarial patch has a limited duration and frequency range. We denote our adversarial audio patch as $p \in \mathbb{R}^{T' \times F'}$, where $T' \lll T$ and $F' \lll F$ denote the small size of the adversarial patch compared to the user's speech spectrogram X.

Transcriptions: Instead of using a single-word label to tag the input, the speech recognition model generates a sentence as output. More specifically, the predicted sentence is the phrase that reaches the highest CTC probabilities, namely, $\underset{Y}{\operatorname{argmax}} \Pr(Y|X)$. In this case, the transcription Y can be decoded from any paths $\pi \in \pi_{X,Y}$, and the length of Y is less than the number of frames (T).

Universality: Most adversarial attacks assume that the attackers know the users' speech and can deliver the perturbation synchronously at a specific time point. However, the assumption does not always hold in a real-world scenario. SpecPatch expects that the attacker can "place" the audio patch at any time, and over any speech context. Let function $A(X, p, t, f)$ be the "place" operation that puts an adversarial patch p to t_{th} spectrogram frame and f_{th} frequency index with any input spectrogram X. Then, our goal is to attain $\hat{Y} = M(A(X, p, t, f))$ for all X in human speech and p on any place of X.

4.2.4: Threat Model

SpecPatch entails the novel adversarial patch attack in the audio domain. We circumvent all the three common failure cases mentioned in Figure 4.1 by introducing the universal adversarial patch and mute signal. The generated adversarial patch is imperceptible and inconspicuous due to the frequency and the time constraint of the spectrogram patch, making SpecPatch a more dangerous and stealthy attack than existing ones.

Adversary's capability: Unlike the prior work [26, 27, 31, 144] that requires the adversaries to know the victims' benign commands in advance to calculate the corresponding audio perturbation, we assume the adversaries have no access to the victim's benign audio and have no knowledge about what the victim will say during their attacks. We assume the adversaries can place a hidden loudspeaker close to the target devices to launch the attack. For the SpecPatch crafting process, we assume the attackers have prior knowledge of the target ASR model. For example, the architecture and model parameters can be found from a public resource. This setting is widely used in most prior work [26,27,34,117,136], and can be generalized to a black-box scenario [32].

Attack scenarios: Unlike all the previous studies, we focus on attacking the ASR system when the user is present. More specifically, the adversary crafts adversarial patches offline, and then uses a preset loudspeaker to deliver the adversarial patch, therefore misleading the target ASR system to make wrong prediction/transcription. For example, the adversary can send fake commands to the voice assistants and request them to perform the wrong operation. Moreover, the adversary can fool the telephone voice system by injecting falsified personal information to trick the ASR-based customer service; besides, the adversary can deny the service provided by the target model via simply broadcasting the spectrogram patches. Due to the shortness and imperceptibility of SpecPatch, the attack can be launched in public spaces (e.g., malls, streets, cafes) with nearby loudspeakers (e.g., smartphone, in-ceiling speaker).

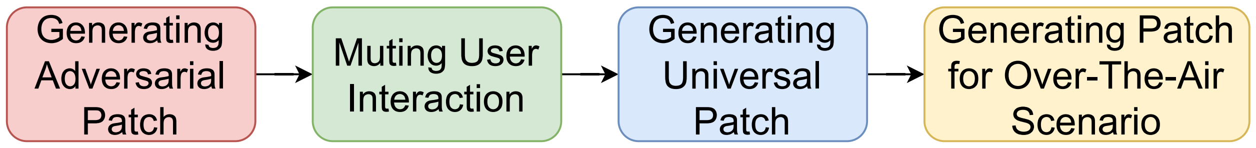

Generating Adversarial Patch
Muting User Interaction
Generating Universal Patch
Generating Patch for Over-The-Air Scenario

Figure 4.3: SpecPatch workflow.

4.3: Design Overview

Figure 4.3 illustrates the system flow of SpecPatch. First, we will craft an adversarial patch to generate the malicious command, i.e., using a short patch to affect a longer benign spectrogram. Second, when the user makes a correction, we need to mute the users' correction by denying the users' followup commands. We achieve that with a specially designed signal called **"Mute"** signal. Next, we make SpecPatch universal to any speech context. This step usually requires the adversarial perturbation to traverse all images/audios in a large dataset to validate the effect of the perturbation on all possible contexts. However, the infinite number of speech contexts makes it computationally infeasible to evaluate a universal perturbation. Rather than optimizing the adversarial patch across different speech contents, we design a *phoneme-level context-free optimization* method. We guarantee that SpecPatch can work across any user interference. The final step of our design is to enhance the robustness of SpecPatch in a real-world scenario. To achieve that, we take the transmission loss of a physical attack into account during the optimization of adversarial patches.

4.4: System Design

This section first analyzes why short perturbations cannot impact long input, and then we describe our strategy to reach our attack goal, i.e., using short patches to attack long commands. After that, we describe the design of the **Mute** signal to deny user's interference. Then, we introduce our *phoneme-level universal* patch crafting process. Finally, we present the techniques to robustify SpecPatch in an over-the-air scenario.

Formulation: Our goal is to craft an adversarial spectrogram patch $\hat{p} \in \mathbb{R}^{T' \times F'}$ that alters all benign spectrogram X and translates them into the target phrase $\hat{Y}$. To achieve this goal, the following expectation needs to be optimized:

$$\hat{p} = \underset{p}{argmin} \mathbb{E}_{X \sim U, t \sim T, f \sim F} \mathcal{L}_{CTC}(A(X, p, t, f), \hat{Y}). \tag{4.5}$$

Here, we compute the CTC loss of patch p when it is applied anywhere (t∼T, f∼F) of the benign spectrogram X, based on which we derive the best adversarial patch $\hat{p}$ that reaches minimal expectations of losses.

4.4.1: Long Command Conversion

Adversarial Patch with CTC Loss

For most adversarial patch attacks in the image domain, the patch will help ensure very high confidence in the target class. Furthermore, recent studies [194, 195] prove that the effectiveness of adversarial patches on deep neural networks (DNNs) is caused by the large receptive fields of CNN layers. As the image classification model maps one image to one label, it connects multiple convolutional layers sequentially. The later convolution layers will have a higher receptive field and will likely include the adversarial patch. Therefore, even a small adversarial patch can be sensed by a later CNN layer and hence affects the global prediction of the image. However, most speech recognition models [16, 84, 196, 222] use a recurrent structure, which usually takes multiple frames as input, produces multiple phoneme predictions for every frame, and then connects the phoneme predictions to form the final sentence prediction. One critical challenge is in applying an adversarial patch to the sequence model. As the adversarial patch could only affect a couple of input frames, the remaining output is barely altered. Therefore, it could be hard to achieve the alteration into a long target sentence.

Suppose the adversarial patch $p' \in \mathbb{R}^{T' \times F'}$ overlaps with K input frames (where K is determined by the window size of the speech recognition model). For ease of explanation, we assume the patch is placed at the left corner of the benign spectrogram, which means $t = 0$ and $f = 0$.

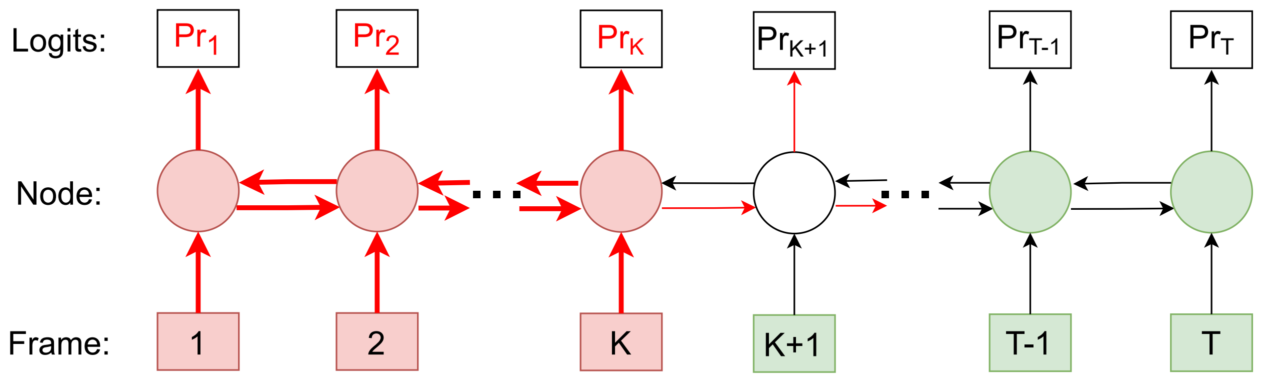


Figure 4.4: SpecPatch flowchart.

While the benign speech has T frames and $T > T'$, there will be a limited number of output probabilities affected by the adversarial patch.

Figure 4.4 demonstrates the workflow of SpecPatch that uses an adversarial patch to attack a sequence model. The bottom blocks show the input frames, while the middle nodes are computational cells of $M(\cdot)$, usually implemented by the LSTM or RNN cells. The top row represents every node's logits output (also known as the probability array of 26 letters). We use red color to mark the frames, nodes, and logits output directly affected by patch p and let the green color label the benign frames and nodes. To demonstrate the data forwarding process of the sequence-to-sequence model, we use red arrows to denote how frames affect the hidden state of nodes and further alter the probabilities up to the K^{th} frame. It can be seen that $\mathrm{Pr}_1, \mathrm{Pr}_2, ..., \mathrm{Pr}_K$ are determined by the frames 1 to K, the intermediate output of previous/next nodes, and the hidden state of the current node. When crafting the adversarial patch, the model $M(\cdot)$ parameters are fixed, so we can only control the value of p' to meet the target transcription. Let X' denote the spectrogram after applying an adversarial patch p. Our goal is to optimize the following objective function:

$$\begin{aligned} &\hat{p} = \underset{p}{argmin} \mathcal{L}_{CTC}(X', \hat{Y}), \\ &X' = X + p. \end{aligned} \tag{4.6}$$

Insight 1: *The restricted length of adversarial patch affects the convergence of the objective function.*

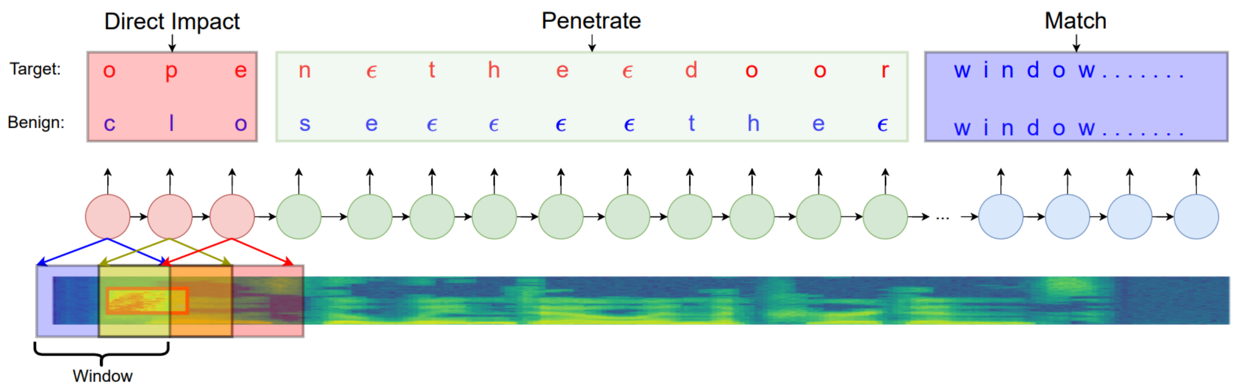


Figure 4.5: Demonstration of patch's impact length.

Observation 1: When optimizing the objective function above, it requires the tuning of p and Pr to match the target phrase. However, limited by the short length of adversarial patch p, the later input frames are untouched during the optimization process, and therefore the values of Pr_{k+1} to Pr_T remain the same. This will make it hard for $\mathcal{L}_{CTC}$ to converge. To explain it in more details, we break down the probability equation into two parts:

$$\text{Pr}(\hat{Y}|X') = \sum_{\pi \in \pi_{X',\hat{Y}}} [\prod_{t=1}^{K} (\text{Pr}_{t,a_t}|X') * \prod_{t=K+1}^{T} (\text{Pr}_{t,a_t}|X)]. \tag{4.7}$$

To minimize $L_{CTC}(X', \hat{Y})$, we aim to maximize $\text{Pr}(\hat{Y}, X')$ as shown in Eq. (4.3). The probability can be separated into two parts in Eq. (4.7). The first term $\prod_{t=1}^{K}(\text{Pr}_{t,a_t}|X')$ denotes the probability that is directly affected by the adversarial patch, which will be fine-tuned continuously by adapting the adversarial patch value. However, the second term $\prod_{t=K+1}^{T}(\text{Pr}_{t,a_t}|X)$ takes the benign X as input, and hence the later probability will remain in low value as it does not match the target letter a_t and has a low chance to be affected by the adversarial patch. This is due to the limited length K of the patch. Therefore, the second term is barely affected as $t > K + 1$. Therefore, when we compute the gradient of $L_{CTC}(X, Y)$, we take the second term into account, but after we update the adversarial patch according to the gradient, we will still get a similar result of the second product term. In short, no matter how to update X', we will have the second term of the gradients remaining the same, which will mislead the direction of optimization of X. In other words, we will not be able to achieve our attack goal if you use the global gradient to update local changes.

Insight 2: *The mismatch length of the target phrase and benign phrase affects CTC loss.*

Observation 2: Besides the shape and value of the adversarial patch, the other critical factor that affects the CTC optimization process is the target phrase. Let us revisit Eq. (4.7): the probability $\Pr(\hat{Y}|X')$ is determined by all the paths $\pi \in \pi_{X,Y}$ that can be merged to the target phrase. While replacing the benign target Y with the target phrase $\hat{Y}$, the number of paths will change accordingly, which will influence the computational cost for CTC loss. For example, if the target phrase $\hat{Y}$ has a length of $l_{\hat{Y}}$, and we assume the length of X is T, we will have total number of paths as follows:

$$\binom{T + l_{\hat{Y}}}{T - l_{\hat{Y}}} = \frac{(T + l_{\hat{Y}})!}{(T - l_{\hat{Y}})!(2l_{\hat{Y}})!}. \tag{4.8}$$

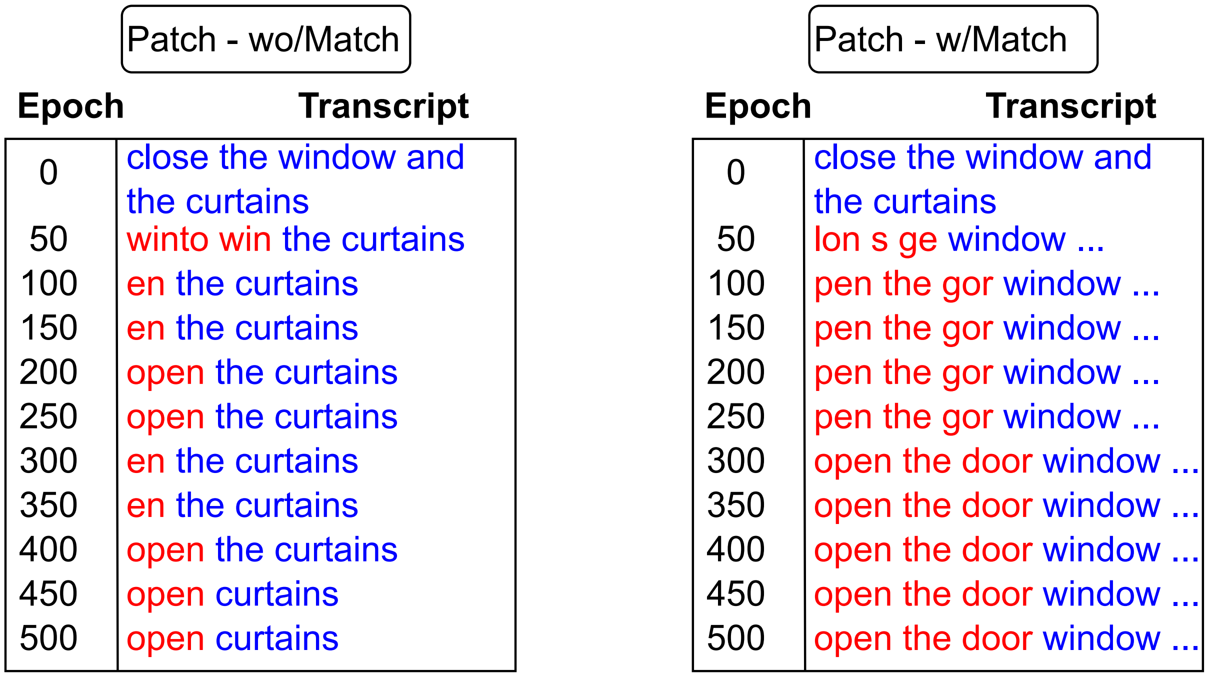
If we have long input and short target phrases, the number of paths for the target phrases will exponentially grow. For example, when T is 15 and $l_{\hat{Y}}$ is 5, the total number of paths would be $\binom{20}{10} = 184,756$. Even though the loss computation can be efficiently computed with dynamic programming [66], it will still result in redundant gradients due to the constrained adversarial length.

Extend the Adversarial Patch Impact

With the previous observations, we find that it is challenging to craft an adversarial patch to alter the recognition of a complete spectrogram. To address the challenges, we propose a novel method called *partial matching*. The basic idea of *partial matching* is allowing the target label to include a portion of the benign label, such that the optimization can focus on the tunable variables. Formally, instead of assigning $\hat{Y}$ as the target when crafting an adversarial patch, we use Y_t to concatenate the target phrase and the benign phrase as: $Y_t = \hat{Y}||Y_{tail}$, where Y_{tail} is the trailing benign phrase.

Figure 4.5 demonstrates the strategy of *partial matching*. Given the benign spectrogram and the adversarial patch as input, the attacker aims to mislead the transcription from "Close the window and curtains" to "open the door." At the bottom of Figure 4.5, we have a benign spectrogram that spans from left to right. Inside the benign spectrogram, there is an adversarial patch (in red color). When we feed the spectrogram to a model, it is divided into frames by a fixed window and a preset hop size. In the middle layer, we use three different colors to denote the state of the nodes. Red represents the nodes that have adversarial input; green depicts the nodes that have benign input but are immediately affected by the previous node's output; Blue means the nodes have a very low possibility of being impacted by the adversarial patch. Every node produces a probability array that records the probabilities of every letter and eventually generates the transcription based on the decoding method (e.g., greedy decoding [83], beam searching decoding [66]). From the top layer, it shows the benign output is "Close the window ...". The target phrase is a concatenation of $\hat{Y}$ ("open the door") and the partial benign label ("window ..."). We use the red region to denote that part of the target can be achieved directly by tuning adversarial patch. The green region of the target phrase can be achieved by extending the impact of the adversarial patch via the internal links between nodes. The blue region is the benign output that ensures the optimization can converge despite of the limited length of the adversarial patch.

Validation of partial matching: Next, we experimentally validate the effectiveness of *partial matching*. The goal is to convert the benign transcription "Close the window and curtains" to the malicious command "open the door" by applying an adversarial patch in the beginning of the audio. We follow the optimization function in Eq. (4.6) to craft an adversarial patch in two different scenarios, i.e., without the partial matching and with the partial matching. In the first scenario, we set $\hat{Y}$ as "open the door". In the second scenario, we use "open the door window and curtains" as our target phrase, which contains a trailing (partial) benign command. Figure 4.6 shows the optimization result up to 500 epochs. At the very beginning, both cases start with the benign label at epoch 0. As the optimizing step proceeds, the first approach (i.e., without partial matching) only alters a single word (red color "open") to match the target. The result remains the same after 400 epochs, which indicates that the optimization converges but does not achieve the attacker's goal (i.e.,, delivering the target command "open the door"). In comparison, the *partial matching* approach converges faster and meets the target phrase within 300 epochs. This experiment shows that, without modifying any parameters or optimization scheme, the *partial matching* improves the convergence speech in crafting an adversarial perturbation. Next, we visualize the adversarial patch in Figure 4.7. For



Patch - wo/Match

Epoch	Transcript
0	close the window and the curtains
50	winto win the curtains
100	en the curtains
150	en the curtains
200	open the curtains
250	open the curtains
300	en the curtains
350	en the curtains
400	open the curtains
450	open curtains
500	open curtains

Patch - w/Match

Epoch	Transcript
0	close the window and the curtains
50	lon s ge window ...
100	pen the gor window ...
150	pen the gor window ...
200	pen the gor window ...
250	pen the gor window ...
300	open the door window ...
350	open the door window ...
400	open the door window ...
450	open the door window ...
500	open the door window ...

Figure 4.6: Comparison between SpecPatch with and without partial matching.

ease of explanation, we assume the adversarial patch starts at the beginning of the benign input and spans all the frequency ranges, i.e., the adversarial patch (the red portion) lasts 500 ms and has 8 kHz bandwidth. The benign label is shown in the top blue field and the concatenated target phrase is in the middle red field. We find that the length of the target command ("open the door") exceeds the range of the adversarial patch, which indicates that the *partial matching* helps achieve the attack goal in extending the impact of adversarial patch and outputting the target command.

4.4.2: Patches to Deny User Input

The proposed *partial matching* mechanism successfully extends the adversarial patch impact length beyond its own duration. However, we still face two challenges to fulfill our attack goal. First, we have no knowledge of the benign phrases in advance, so it is impractical to adjust the optimization of the target phrase for every possible benign phrase. Second, the human factor (e.g., user interaction or long user commands) cannot be resolved because the adversarial patch cannot affect the speech transcription that is far away from the patch position. To overcome the challenges, we propose *Mute Patches* by exploiting the *discrepancy* of the ASR model's input and output mapping.

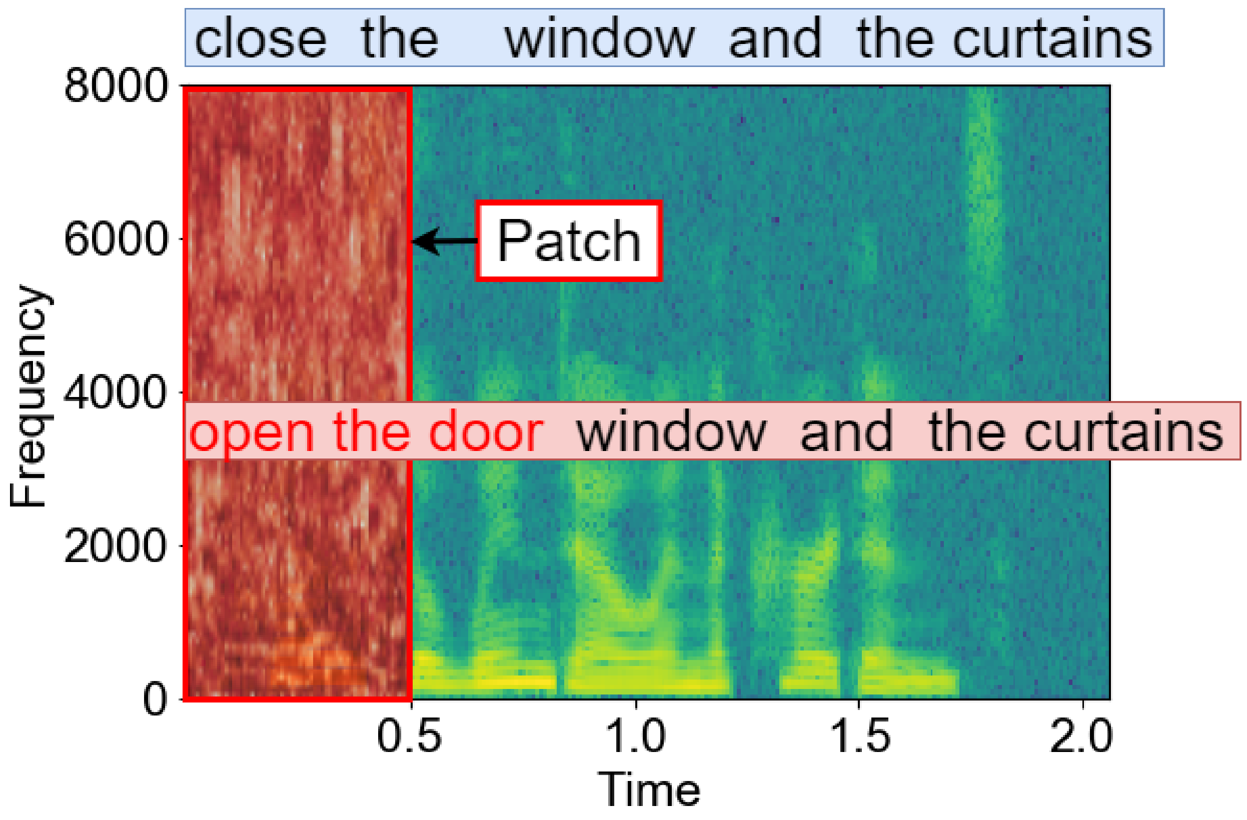
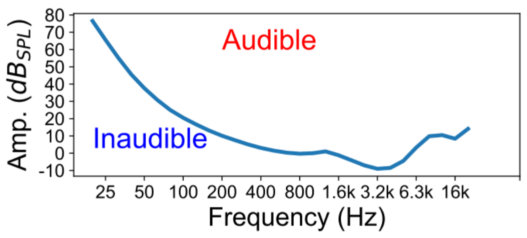
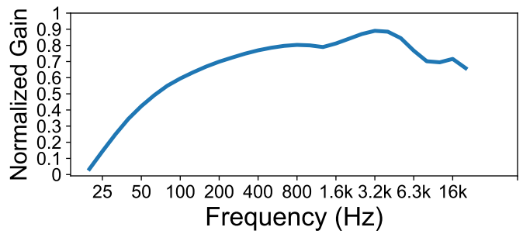


Figure 4.7: The effect of patch towards a long command.

Design mute patches: The design goal of mute patches is to disrupt the user's commands without attracting their attention. Specifically, we aim to inject a few adversarial samples with low volume to mislead the ASR model to produce empty transcriptions. To design such mute patches, we review the complete speech recognition process and find the opportunity to meet our design goal. As described in §4.2.3, the waveform serves as the raw input, which is converted into spectrogram to be fed into the ASR models. Next, every node of the ASR model takes a couple of spectrogram frames and outputs a letter prediction. By reviewing the whole process, we realize that every node of the ASR model perceives a large scope of waveform samples. A similar phenomenon has been observed by prior studies [194, 195] in image recognition models, and the authors conclude that a large receptive field of the neural node is responsible for the adversarial patch attack because a small patch in an image can be perceived and misinterpreted by a neural node. Inspired by their findings, we are motivated to inject sampled adversarial audio signals into the neural nodes. To



(a) Hearing Curve of Human



(b) Reflection of Hearing Curve

Figure 4.8: Optimization of patch frequency based on the auditory property of human.

craft the mute signals, we formulate the following problem:

$$\begin{aligned} p^m &= \underset{p}{argmin}\mathcal{L}_{CTC}(X', Y_b), \\ X' &= X + Tp, \end{aligned} \tag{4.9}$$

where Y_b is a phrase that only contains blank symbols, and p^m is the mute patch that is composed of multiple patches such as $p^m = [p^1, p^2, ..., p^L]$. For every patch that has $1 \leq l \leq L$, we have $p^l \in \mathbb{R}^{1\times F'}$. The size of the mute patch is $1 \times F'$ because a single adversarial sample can only affect one bin of the spectrogram. We set the length of the mute patch as T, and X' is the resulting spectrogram. By optimizing Eq. (4.9), we can craft the mute signal in the time domain with minimal loss value. The choice of T is determined by the hop length of STFT and the input size of the ASR model. In practice, we can set the value of T to be the same as the W_{STFT}, such that we can ensure every vertical spectrogram bin contains adversarial information.

4.4.3: Imperceptible Patch

When reviewing the existing attacks, we find that most prior adversarial audio attacks (e.g., [27, 119, 136, 204]) aim to minimize the amplitude of the perturbation (i.e., minimize $dB(\delta)$), e.g., by including the perturbation amplitude in the loss function. However, we find that although these perturbations are well-optimized, they are still audible when performing the physical attacks.

In our attack scenario, we expect to launch an imperceptible attack when the victim user is close to the adversary. Since this goal is hard to achieve by the optimization method (i.e., penalizing the amplitude of perturbation), we design a new approach to satisfy the imperceptible attack goal. In a nutshell, the imperceptibility of SpecPatch is ensured by the short duration of the adversarial spectrogram patch and further secured by the narrow frequency band of SpecPatch.

In the prior optimization settings, the crafted perturbations are audible because the victim microphone is sensitive to a certain input amplitude. Here, we focus on yielding the perturbation inaudible without dropping its amplitude. To achieve this goal, we investigate the human hearing sensitivity curve and find that the human ear has uneven sensitivity to different frequencies. We depict the hearing curve in Figure 4.8a. Formally, the hearing curve can be represented by a function with f, and we denote it as $H(f)$. The source data is measured by prior auditory research on equal loudness contours [100]. In the figure, the blue line indicates the required amplitude for pure continuous tones at a specific frequency that can be heard by humans. Above the curve, we can feel the sound at such loudness, while below the curve, the sound intensity is insufficient. For example, one can hear continuous audio with frequency at 100 Hz as long as it has more than $20dB_{SPL}$. Once the volume is decreased to less than $20dB_{SPL}$, the human can no longer perceive it. From the shape of the curve, we find that the human auditory system is more sensitive to a frequency between $1.6kHz$ and $4kHz$. In comparison, we are unperceptive to sound below $1.6kHz$, as the lower frequency stimulates less attention from human ears. Therefore, we can design low-frequency patches (e.g., $< 1.6KHz$) to diminish the perceptual level of human hearing. To reach this goal, we add a *frequency selective penalty* term to the objective function in Eq. (4.9). The updated function is presented below:

$$\begin{aligned}\hat{p} &= arg \min_{f_l<p<f_h} \mathcal{L}_{CTC}(X', Y_t) + ||p * \widetilde{H(f)}||_2, \\ \widetilde{H(f)} &= Normalize(-H(f)).\end{aligned} \tag{4.10}$$

This new term $||p * \widetilde{H(f)}||_2$ is composed of the patch p and a frequency response function $\widetilde{H(f)}$, and we multiply them together to compute the L_2 norm result. f_l and f_h indicate the low and high-frequency boundaries of the learned patch. Compared with existing attacks [27, 119] that assume a constant value in the penalty term, we design a frequency response function as a adjustable coefficient. The goal of this term is to selectively penalize the human sensitive frequency portion

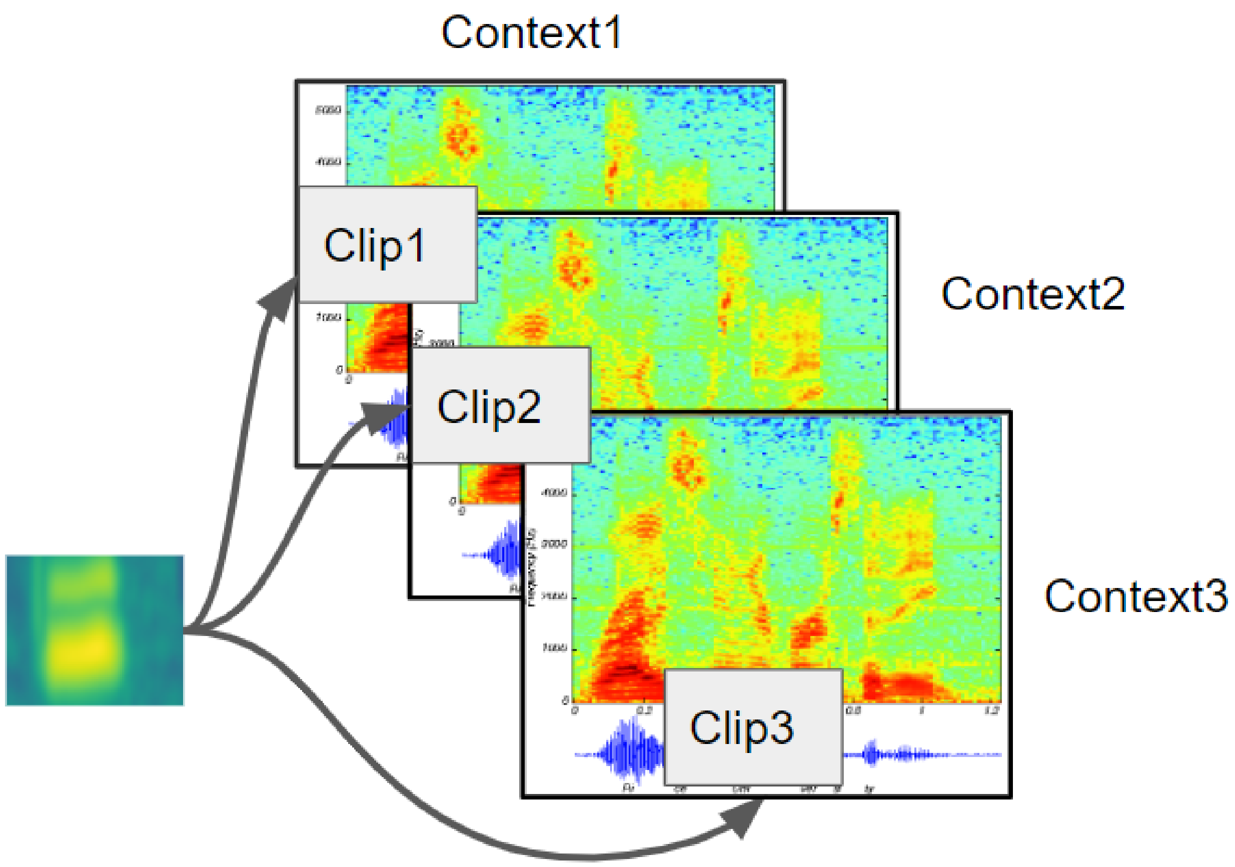


Figure 4.9: Universal perturbation.

(e.g., $1.6kHz$ and $4kHz$) and retain the insensitive components (e.g., $< 1.6kHz$) in the adversarial patch. We design the frequency response function $\widetilde{H(f)}$ based on the human hearing curve shown in Figure 4.8a. By performing a reflection operation and normalizing the result in the range of 0 to 1, we can obtain the $\widetilde{H(f)}$ as shown in Figure 4.8b. The rationale of such an operation is to reduce the human-sensitive energy while retaining the inaudible portion of the adversarial patch.

4.4.4: Universal Patch

The prior adversarial attacks either use AE or Perturbation to launch the attack. As shown in Table 4.1, most prior work leverages AE to deliver the adversarial commands. However, there is a major concern with this method, as the user's intervention could disrupt the AE. Because the AE needs to be crafted with known benign commands, it requires the adversary to predict the incoming benign commands in advance. In a real attack scenario, it is hard to predict incoming commands.

Therefore, such unpredictable user intervention could disrupt the performance of AE. The only exception among the previous attacks is AdvPulse [119], which utilizes a *universal perturbation* to launch the physical attack. They use the iterative greedy algorithm [127] to generate a perturbation that works with any given contexts. However, AdvPulse is proved effective on the single-word commands. Compared with single-word commands, diverse speech contents make it more difficult to compute the universal patch. Generally, a universal perturbation is crafted by iterating over all the benign contexts [119, 127]. Formally, the perturbation can be represented by the following formula:

$$\begin{aligned}&\hat{p} = \underset{p}{argmin}\mathbb{E}_{X\sim U}\mathcal{L}((X+p),\hat{Y}),\\&U = \{X \mid X \text{ is speech with arbitrary contents}\},\end{aligned} \tag{4.11}$$

where $\hat{p}$ is the universal perturbation, X is the benign context (e.g., background sound), and U is the set that includes all possible benign samples. Due to the diverse speech contents, U will become a large set, resulting in a substantial computational cost. To reduce the cost, we propose *phoneme-level universal* optimization scheme, and the logic of this approach is depicted in Figure 4.9. In short, the proposed *phoneme-level universal* scheme reduces the size of U by introducing clips. One clip contains one single phoneme, and we added padding to ensure every clip has the same duration as the patch as shown in Figure 4.9. Since there is a limited number of phonemes (i.e., 44 phonemes in English) in the speech context, the clip set is much smaller than the speech set. The clips can be simply retrieved from the speech dataset. We formalize the proposed *phoneme-level universal optimization* as follows:

$$\begin{aligned}&\hat{p} = arg \underset{f_l<p<f_h}{min} \mathbb{E}_{X\sim U}\mathcal{L}_{CTC}(X',\hat{Y}) + ||p * \widetilde{H(f)}||_2,\\&U = \{X \mid X \text{ is clip with fixed duration}\}.\end{aligned} \tag{4.12}$$

4.4.5: Overall Physical Over-the-Air Attack

As mentioned above, we guarantee that SpecPatch is able to deliver long commands over users' intervention in any speech context. Specifically, we first apply a universal mute signal to override the entire original speech, which will result in a blank-transcription. Partial matching then takes a part of blank transcription as input to generate an imperceptible adversarial patch for the target phrase. Note that, similar to the generation of adversarial patch, the generation of mute signals also does not require knowledge of original phrases. For example, suppose the benign command is "open the door", the target is "close the door", we first apply the mute patches to the benign audio to convert "open the door" into a blank transcription ("—...–"). Then, we generate the adversarial patch based on the muted benign input. If the benign command is longer than our target, after applying the mute patches, we will set "close the door———" (with trailing blank symbols) as target to generate patch.

However, to launch the attack in a real-world scenario, we need to resolve the patch distortion during the over-the-air transmission. We follow the design in [119, 197] due to its simplicity. In general, three operations are considered during the crafting process: 1) band-pass filtering, 2) room impulse response, and 3) ambient noise mitigation. The bandpass filter is designed to cope with the uneven frequency response of the speaker and microphone. The room impulse response (RIR) is introduced to compensate for the absorption and reverberation in the environment. Finally, the ambient noise is considered to craft a robust perturbation that resists environmental noise. In practice, we form the following final objective function to include the operations mentioned above:

$$\begin{aligned} \hat{p} &= arg \min_{f_l<p<f_h} \mathbb{E}_{X\sim U}\mathcal{L}_{CTC}(X', \hat{Y}) + ||p * \widetilde{H(f)}||_2, \\ X' &= X + BPF(p) * R(f) + W. \end{aligned} \tag{4.13}$$

The *BPF* refers to the band pass filter, and we follow the setting in [197] to configure the cut-off frequency as $50 \sim 4,000Hz$. The $R(f)$ represents the spectrum of the room impulse response. We use the RWCP dataset [130] to enrich the RIR measurements and further compute the spectro-

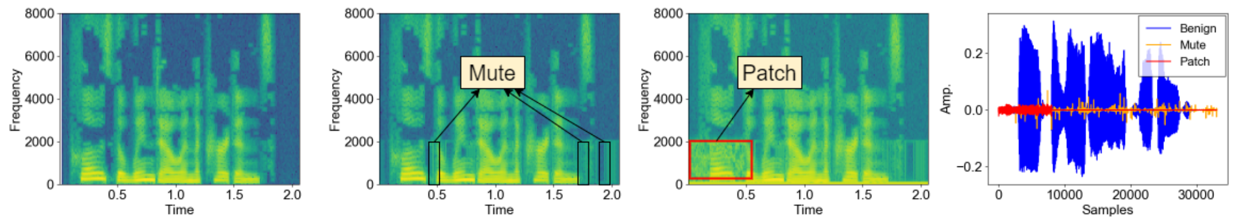


Figure 4.10: Spectrogram and time-domain signals of SpecPatch.

gram of the RIR audios. The noise spectrogram W is chosen from another ambient noise dataset NOISEX-92 [172], which contains various noises (babble noise, factory noise, HF radio channel noise, pink noise, white noise, vehicle noise).

4.5: Evaluation

4.5.1: Experiment settings

We implement SpecPatch using the Tensorflow [3] framework. We craft adversarial patches following Eq. (4.13). The experiments are conducted on a desktop with Intel i7-7700k CPUs, 64GB RAM, and NVIDIA 1080Ti GPU, running 64-bit Ubuntu 18.04 LTS operating system. In the evaluation, we launch our attack in two scenarios: *over-the-line* and *over-the-air*. In the over-the-line attack, we pass the SpecPatch directly to the model as a Waveform Audio file. In the over-the-air setting, we attack the victim's phone using a speech-to-text service. In our implementation, we set up a server that runs our target ASR model and allows the victim's phone to request service through the local area network.

Target model selection: As our attack target is the speech recognition model, we will examine the effectiveness of SpecPatch on the most popular ASR models. Specifically, we select DeepSpeech2 [16] as the target ASR model. DeepSpeech2 leverages CTC loss and recurrent cells to improve the recognition performance.

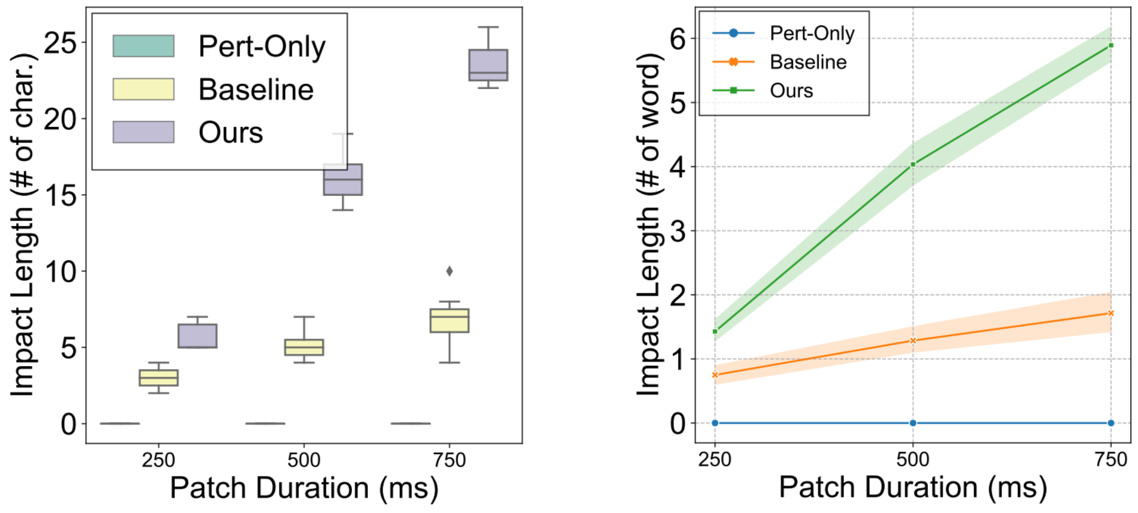
Metrics: We use the following metrics to quantify the effectiveness of our attack: *(1) Success Rate:* this metric is the ratio of successful attacks and the total attempts. We report success only when the prediction matches the targeted command in a targeted attack. In terms of the untargeted attack, we measure the success rate of mis-transcribing the victim user's benign commands. *(2) Impact Length:* This metric is to identify how many characters/words can be affected by our SpecPatch. Given a long benign spectrogram, we inject an adversarial patch and measure the length of characters/words that are different from the original transcription. *(3) L2 Distortion:* the L2 distortion $||p||^2$ indicates the amplitude of adversarial patches. Prior to the launch of a physical attack, we can measure the distortion value by summarizing the squared amplitude of the generated perturbations.

Datasets: The dataset we choose as benign audio is TIMIT [1]. This dataset contains four types of corpora designed jointly by the Massachusetts Institute of Technology (MIT), SRI International (SRI), and Texas Instruments, Inc. (TI). It contains 6,300 audios from 630 speakers. The duration of each audio is around 5 seconds and contains approximately ten words. For our target commands, we collect them from the website ok-google.io, which provides commonly used commands on Google Assistant. We select ten sentences as our attack goal, e.g., "find my phone" and "turn on the lights." For the phoneme clip dataset, we construct it manually by following the annotations in TIMIT [1]). In total, we obtain 50,487 phoneme clips that cover 44 phonemes. We added padding to ensure every clip to have fixed length as 500 ms, which matches with the length of the adversarial patch.

4.5.2: Over-The-Line Attack

Over-the-line SpecPatch: We first showcase the capability of SpecPatch in converting the benign audio into our target transcription by injecting adversarial patches. As shown in Figure 4.10, the first spectrogram represents the benign audio "close the window and curtains". We first apply mute patches to translate it into consecutive blank symbols which results in an empty transcription. The mute patches are crafted by Eq. (4.9). We use rectangle boxes to mark those mute patches. It can be observed that each mute patch only occupies a single frame and periodically appears as vertical lines below $2kHz$. After applying the mute patches, we can craft an adversarial patch to meet our target goal by leveraging the *partial matching* strategy (i.e., concatenating the target command with trailing blank symbols). The third figure depicts the adversarial patch on the muted spectrogram.

We use a red rectangle box to highlight the position of the adversarial patch. We can see that the adversarial patch occupies 0.5 second within $50Hz \sim 2kHz$ and is placed at the beginning of the spectrogram. The frequency band of this patch is learned from Eq. (4.10) and further constrained by Eq. (4.13). To investigate the amplitude of the benign audio, mute signal, and adversarial patch, we visualize the waveform of the complete AE in the rightmost figure. Compared to the benign audio, the adversarial and mute patches have a very low volume ($\sim$%5 of the benign audio).



(a) Impact length in characters (b) Impact length in words

Figure 4.11: Comparison of impact length.

Evaluate the impact length: To evaluate the effectiveness of *partial matching* strategy, we conduct experiments with three different strategies to attack a long command (~ 10 words) with different patch duration. The first strategy, perturbation-only (Pert-Only) strategy searches for a short patch that delivers long commands without considering the benign audio in crafting the AE; the second strategy utilizes a long empty audio as the background sound to increase the logits output-length, and then craft a patch based on the benign audio. This approach is adopted by the prior work [119]. The third strategy leverages the *partial matching* strategy. Similar to the second strategy, this approach uses an empty audio as background. However, it configures the target command to include trailing blank symbols, as illustrated in Section 4.4.5.

We randomly select 20 sentences in TIMIT dataset as our target and use three different lengths of the patch ($250ms$, $500ms$, $750ms$) to achieve the attack goal. Figure 4.11 presents the comparison of impact lengths among the three strategies. The "Perturbation-Only" label represents the first strategy as it does not consider any benign audio. The "Baseline" label symbolizes the second strategy, and the "Ours" represents the *partial matching* strategy. From Figure 4.11a, we find the first strategy failed to craft a short patch (impact length is 0) that delivers a long command for all the three lengths of patch configurations. This can be attributed to the lack of logits output for a short variable (perturbation), leading to the error of *"INVALID ARGUMENT"* caused by *"not enough time for target transition sequence"*. In contrast, the baseline strategy can proceed with the optimization process with a long benign input. However, it can only affect a couple of characters (< 10) (see Figure 4.11a) and a limited number of words (< 2) (see Figure 4.11b) even with the longest patch configuration. Using the proposed partial matching strategy, we can use a patch with the same duration to attack longer sentences, and the impact length reaches ~ 15 characters and ~ 4 words for $500ms$ patch, and this can be further extended to ~ 6 words with a longer patch (e.g., $750ms$). In summary, using the same length of patch, we extend the impact length by 200% in characters and 287% in words, which proves the effectiveness of the *partial matching* strategy.

Evaluate mute patches: Next, we evaluate the generation of *mute patches* with different speech contexts. Since we expect that the mute patches could disrupt any user interference, we craft the patches based on a variety of sentences. More specifically, we randomly select multiple sentences from the benign audio dataset and craft one series of mute patches that are applicable to all sentences. We report the L_2 distortion of the mute patches in Figure 4.12.

We use a variable *"X Comm."* to represent the situation that mute patches can change the transcription to empty symbols for all "X" number of selected commands. The X value ranges from 10 to 50. The results show that the mute patches converge speedily for all the situations and reach a steady value in 200 iterations. Moreover, we notice that the more general mute patches we request, the higher power of mute patches we will get. For example, to craft a series of mute patches that are suitable for 50 commands, it triples the value of L_2 distortion after convergence compared to the solution with 10 commands.

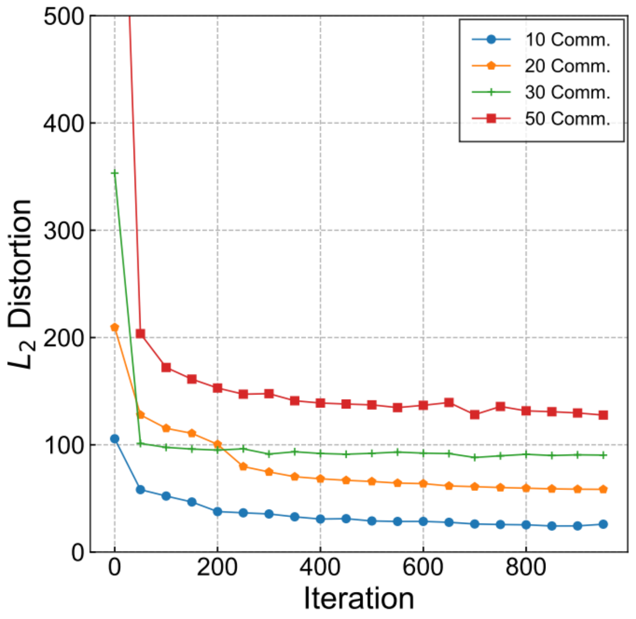


Figure 4.12: Universal mute patches.

Evaluate phoneme-level universal perturbation: After analyzing the generality of mute patches, we then investigate the universality of adversarial patches on different background audio contexts. The adversarial command patch is obtained by optimizing the objective function in Eq. (4.12). In this evaluation, we craft adversarial patches over 50,487 phoneme clips and report the success rate when using an adversarial patch on top of the specific phonemes. As depicted in Figure 4.13, the universal adversarial patch achieves 68% success rate on all the five types of phoneme clips. Furthermore, among the different types of phonemes, we observe that vowels and nasal are more compatible with the adversarial patch. In contrast, the fricatives and stops are more resilient against adversarial patches. This is because our adversarial patches contain more low-frequency energy due to Eq. (4.12), and the vowels and nasal present rich low-frequency components. Therefore, these phonemes will be affected much easier. In contrast, the fricative and stops contain more high-frequency energy than that of the patch.

Evaluate patch with different frequencies: One key benefit of using spectrogram patch rather than short pulses is that our patch can be more imperceptible. Generally, a patch with a wider bandwidth can be found much faster and can have a lower amplitude. In comparison, a patch with a narrow bandwidth could raise less attention in human auditory system, however, it will fail

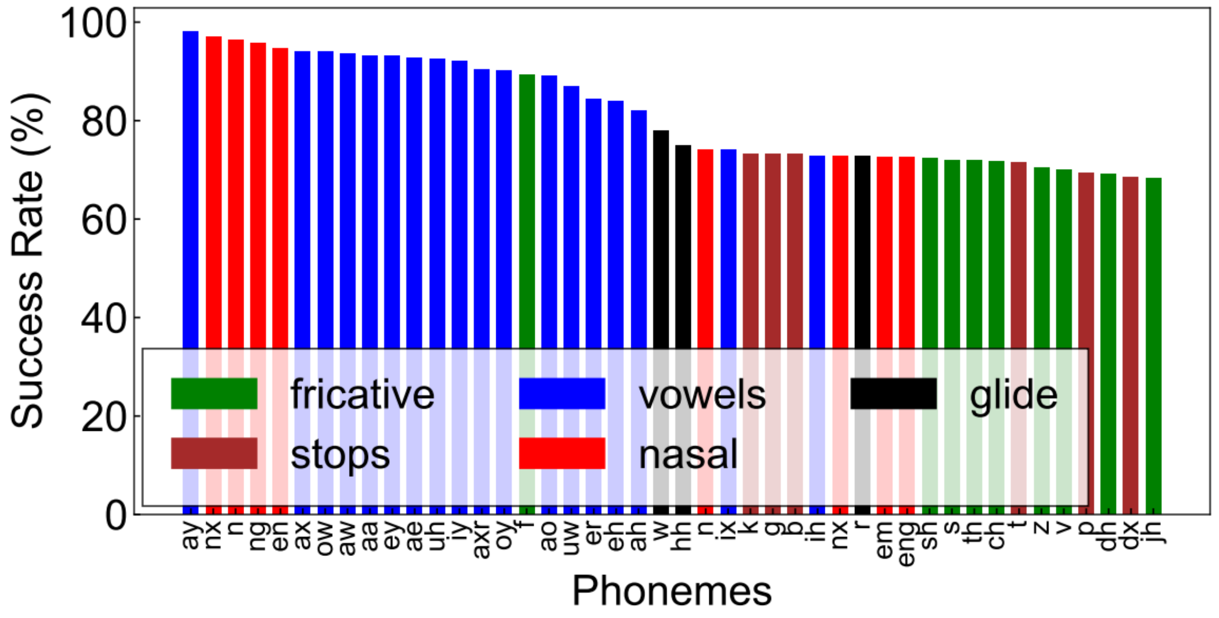
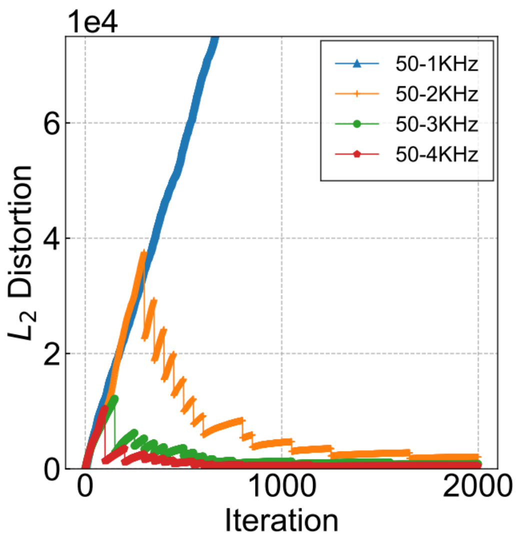
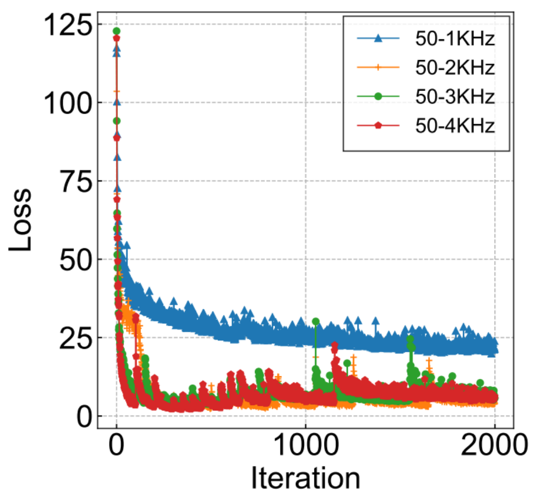


Figure 4.13: Universal attacks across different phonemes.

to deliver certain commands because of the lack of some specific frequency energy. Therefore, there is a trade-off between the frequency band, the perceptual level, and the successful rate of the patch. To investigate the best frequency band of the adversarial patch, we conduct the following experiment with four different frequency settings as follows: $50Hz \sim 1kHz$, $50Hz \sim 2kHz$, $50Hz \sim 3kHz$, and $50Hz \sim 4kHz$. We aim to retain the low-frequency components as they are more likely neglected by human ears (see Figure 4.8a). Our goal is to find a narrow frequency patch that has more low-frequency components and with lower overall amplitude. We craft 10 adversarial patches with every frequency setting and record the average L_2 distortion and loss during the process. As shown in Figure 4.14a, if the patch only includes $50 \sim 1KHz$, it fails to achieve the target goal as the L_2 distortion keeps growing, and the loss in Figure 4.14b implies that this setting leads to the highest loss. For the $50 \sim 2KHz$ setting, the distortion rises at the first 400 iteration, and then it satisfies the target goal and starts dropping the amplitude of patch by iterations. The rest two settings converge faster. These results indicate that patch with a wider bandwidth is more likely to meet the target. To summarize, the $50 \sim 2KHz$ is the best choice, because it reaches the low amplitude after 1,000 iteration and has comparable L_2 distortion with



(a) The distortion of SpecPatch



(b) The loss of SpecPatch

Figure 4.14: Comparison of different frequency ranges.

other cases, and it also occupies less bandwidth. As a result, we use this setting for all the rest experiments.

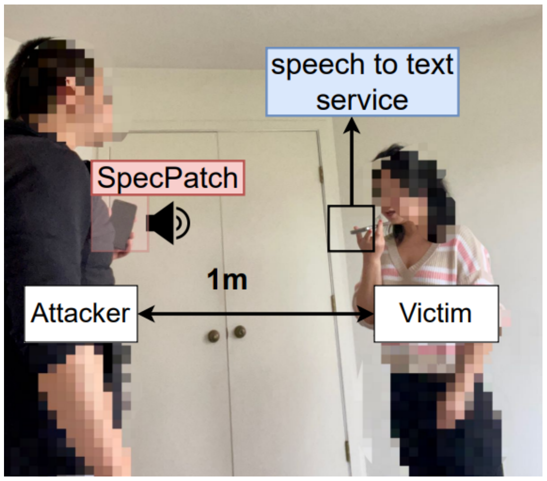
Target	Success Rate	Mis-Trans. Rate
”turn on the lights”	94.1%	100%
”find my phone”	96.5%	100%
”Turn off the kids’ wifi”	87.1%	100%
”What is my password?”	86.5%	100%
”stop the music”	97.2%	100%
”open the door”	100%	100%
”lock the front door”	91.3%	100%
”Listen to the news”	92.5%	100%
”call 911 now”	98.5%	100%
”Cancel my alarm”	94.6%	100%

Table 4.2: Over-the-line attack performance

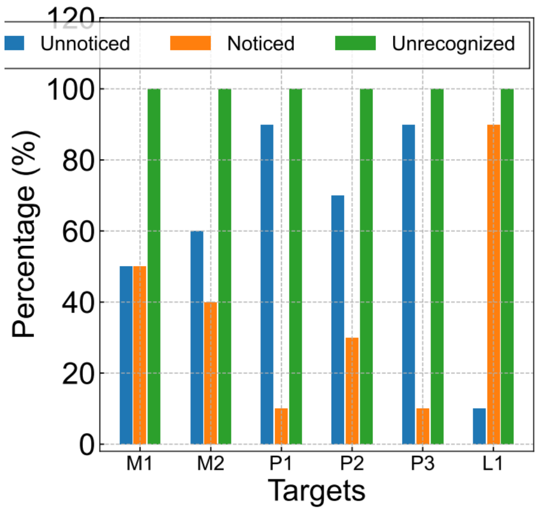
Evaluate overall performance: In this experiment, we train 10 universal patches along with a series of mute patches. Then, we apply this specpatch to all of the benign audios in TIMIT dataset at random positions to validate the effectiveness of our attack. Table 4.2 reports the success rate after adding 10 different adversarial patches to 6,300 benign audios, resulting in 63,000 AEs. We count the success cases when the target command is the only output in the transcription. Otherwise, if the benign label is transcribed wrongly, we regard it as the *mis-transcription* case. We find that, SpecPatch could cause 100% mis-transcription rate for any audio. This means SpecPatch could almost surely deny the service of users. In terms of the target success rate, we achieve $>$ 90% success rate for 8 out of 10 patches. The success rate for longer commands that have 4 to 5 words is lower than those of shorter commands, this is reasonable since longer target commands are hard to achieve in a noisy background.

4.5.3: Over-The-Air Attack

Attack scenario: Figure 4.15a depicts the attack scenario. The victim is using the speech-to-text service while the adversary uses a smartphone to play SpecPatch to deceive the ASR model. Note that the adversary can play the attack audio at any time, and once he/she launches the attack, the victim's commands will be denied by the consecutive mute signals. In our experiment, the adversary is 1 meter away from the victim, and SpecPatch is played at different volumes. We measure the loudness of the user's interference and the SpecPatch audios by a decibel meter. We conduct the experiments in three places: an indoor room, an outdoor street, and a public dining hall.



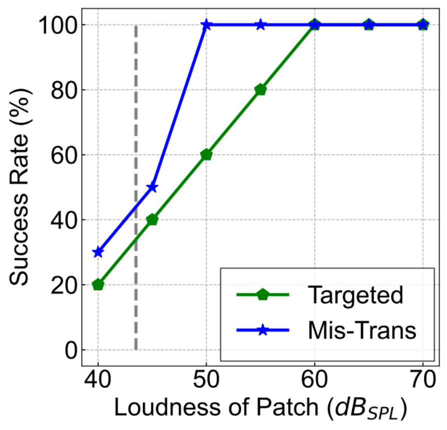
(a) Real-world attack scenario



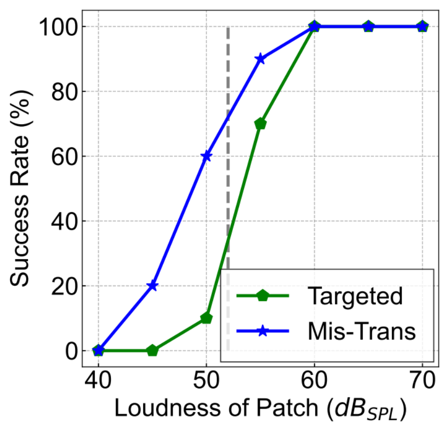
(b) User study results

Figure 4.15: Over-the-air SpecPatch attack.

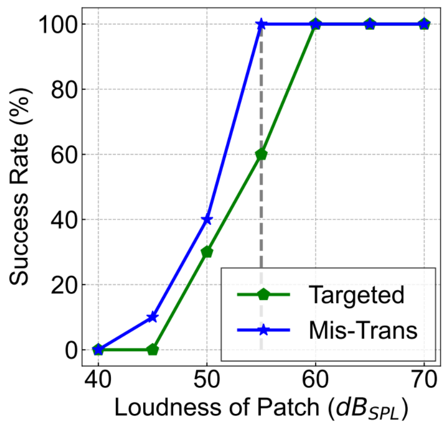
Attack performance: In this experiment, we play a crafted patch of "open the door" 10 times for



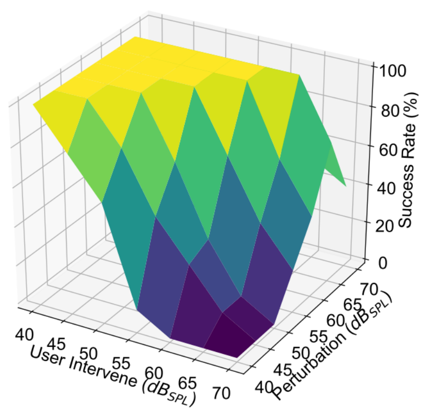
(a) Indoor scenario



(b) Outdoor scenario



(c) Public dining hall scenario



(d) User's speech volume vs. patch loudness

Figure 4.16: Attack success rate across four different scenarios.

each volume, attempting to deliver this command to victim's phone. The victim is holding their smartphone and speaking at the volume of $55dB_{SPL}$. The ambient noise levels of the three places are $43.5dB_{SPL}$, $52dB_{SPL}$, $55dB_{SPL}$ for indoor room, outdoor street, and dining hall, respectively. We present the success rate of targeted attack and the mis-transcription attack in Figure 4.16. The grey dot line indicates the ambient noise level. As can be seen from the Figure 4.16a, when the perceived patch volume is lower than the ambient noise level, there are 8 out of 10 attempts failed in the targeted attack scenario. Once the victim device perceives a comparable power (e.g., 45dB) from the patch audio, the success rate increases to 40% for the targeted attack. When the volume is $10dB$ greater than the ambient noise, we can achieve 80% success rate, and 100% in denying the user's input. We observe the similar results in Figure 4.16b and Figure 4.16c. These results indicate that SpecPatch can successfully attack the ASR system with a limited power profile. Typically, SpecPatch achieves successful attacks when there is $< 5dB_{SPL}$ power difference between the patch and the ambient noise. If we raise the attack power, the success rate can be assured to 100%.

To better understand the relationship between user's volume and the loudness of patch, we conduct another experiment to control those two factors. We play the same patch 10 times at 7 volumes (from $40dB_{SPL}$ to $70dB_{SPL}$ with $5dB$ increments). For every volume, we use another speaker to play a benign audio with increasing volumes. This experiment is conducted in the same indoor place, and the result is present in Figure 4.16d. We find that when the patch has same volume of the benign audio, it achieves 100% success rate. If the patch is $20dB$ less than the benign audio, SpecPatch no longer works. In general, a louder patch can achieve a higher success rate. Noticeably, when both the patch and the benign audio have high power, the success rate reduces to 40%.

4.5.4: User Study

To evaluate the stealthiness of SpecPatch in a real-world attack, we conduct two online user studies that involve ten volunteers to investigate the users' perception level of SpecPatch.

Study 1: In this study, the users are requested to hear four AEs that include the crafted patches. Then, we ask the volunteers about the contents they heard. The benign and adversarial transcriptions are described in Table 4.3. For the same benign sample, we add three different patches to achieve three goals (one of them is an empty transcription). The result shows that 10 out of 10 volunteers are deceived by our attack, as all of them consider the benign label as their heard content. Surprisingly, we can inject the patch to a silent benign audio, and this implies the possibility of a hidden attack. Similarly, none of the volunteer can perceive the hidden patch, as 10 out of 10 considered the malicious "turn on the wifi" patch as a silent audio.

Study 2: To further validate SpecPatch, we conduct the second user study. The volunteers are asked to *pretend to speak to their voice assistants* while hearing the six different patches, these patches are played through their smartphones with a medium volume. The distance between the volunteer and the their smartphones is 0.5 meter. After that, they will answer questions to describe their comprehension of the heard patch. The options of perception levels include: *Unnoticed*, *Noticed*, and *Unrecognized*. *Unnoticed* indicates that the volunteer cannot hear a patch; *Noticed* implies that the volunteer can hear a patch but regard it as a normal noise; and *Unrecognized* stipulates that they cannot understand the meaning of the heard sound. We report the experimental result in Figure 4.15b. The labels in x-axis represent different patches, namely, (M1 and M2 are two mute patches, P1-P3 are short patches, while L1 is a long patch that is composed of 3 short patches). It shows that most of participants ($>$ 70%) cannot even notice our short patch attack (P1-P3). For the consecutive mute patches, there are around 50% of volunteers can perceive it. For the long patch, 9 out of 10 participants can clearly feel it. It is noteworthy that none of the patches can be understood by volunteers.

	Benign	**Adversarial**	**Deceive**
	”turn on the lights”	”open the door”	10/10
	”turn on the lights”	””	10/10
	”turn on the lights”	”open the window”	10/10
	””	”turn on the wifi”	10/10

Table 4.3: User case study

4.6: Discussion

Limitations: SpecPatch has the following limitations: 1) the attack is model dependent; 2) the attack could not successfully attack very long sentences; 3) the attack distance is relatively short. For the first limitation, this attack can only attack the recurrent neural network, since our attack is established by exploiting the vulnerability of inter-connection between each cells. The second limitation can be addressed by introducing a longer patch, however, it might raise the alert of the victims. The third limitation can be possibly addressed by amplifying the power of the patch, but the adversary needs to handle both the distortion from the amplifier and the long-distance induced attenuation.

Defense: Prior studies [119,202,204] reveal that signal processing techniques can defend the adversarial audio attack since the adversarial perturbations are delicately crafted and hence are deemed fragile. The signal processing techniques, however, can reduce the fidelity of perturbations and protect the ASR models. Typical signal processing defense methods include 1) *Down sampling (DS)*: decreasing the sampling rate of AEs to degrade the quality of AEs [119, 202, 204]; 2) *Quantization*: this approach rounds the 16-bit precise value to its nearest integer multiple of constant Q, which has been adopted to defend against the attacks [119, 202]. 3) *Low pass filtering (LFP)*: this defense can use a Butterworth low-pass filter with different cutoff frequencies to remove the high-frequency components of the perturbations [119]. We will evaluate different defense approaches against SpecPatch in our future work.

4.7: Summary

In this chapter, we proposed SpecPatch, a human-in-the-loop adversarial patch attack on ASR systems. SpecPatch considers the scenarios when the users are presenting or intentionally disrupting the adversarial audio attacks against ASR systems. SpecPatch optimizes the adversarial patch to increase the length of the target commands. SpecPatch also includes Mute adversarial samples that can ensure the user interference does not affect the adversarial perturbation. Moreover, we further enhance SpecPatch to make it imperceptible and robust in both over-the-line and over-the-air attack scenarios. Our extensive real-world experiments show that SpecPatch can unnoticeably deliver malicious commands in a noisy environment amid user interference. Although SpecPatch is dangerous to conventional ASR systems, it turns out this attack highly relies on the open source of the ASR model to craft the malicious sound. In the next chapter, we will introduce a new black-box attack, which targets the more challenging task: attack the commercial speech-to-text services and voice assistants with zero knowledge.

CHAPTER 5: BLACK-BOX, QUERY-EFFICIENT AUDIO ADVERSARIAL ATTACK VIA SPLIT-SECOND PHONEME INJECTION[5]

.

5.1: Introduction

In the previous chapter, we introduced a SpecPatch attack in which the attacker can manipulate the transcription of the Automatic Speech Recognition (ASR) system by injecting an adversarial audio patch. Although the aforementioned attack is powerful and dangerous, it is not capable of attacking the commercial Speech-to-text services (Amazon Transcribe [15], Google Cloud Speech-to-Text [64], IBM Watson Speech to Text [97], and Microsoft Azure Speech Service [125]) and Intelligent Voice Control (IVC) devices (Google Home [62], Amazon Echo [13]) due to lack of their model information. In this chapter, we discover the possibility of attacking commercial speech recognition services in a black-box manner, where the attacker aims to craft the audio adversarial example in limited time and resources, without any knowledge of the target speech recognition models.

With the increasing presence of ASR systems and IVC devices in private spaces, users begin to worry about the security and privacy of these systems. For example, a hacked device is now capable of recording private conversations; collecting and sharing private data; and controlling all the connected IoT devices in smart homes [37, 158]. Researchers have demonstrated that ASR systems could become vulnerable to a wide variety of attacks. For instance, inaudible commands

[5]This chapter is based on previously published work by Hanqing Guo, Guangjing Wang, Yuanda Wang, Bocheng Chen, Yuanda Wang, Qiben Yan and Li Xiao titled "PhantomSound: Black-Box, Query-Efficient Audio Adversarial Attack via Split-Second Phoneme Injection" published at the Proceedings of the 2023 International Symposium on Research in Attacks, Intrusions and Defenses (RAID). DOI: 10.1145/3607199.3607240 [71]

can be injected through ultrasound [141, 210], even across different transmission media, such as object surface [200], light [158], etc. Besides the physical attacks, recent studies also utilize the discrepancies between the human ear and feature extraction algorithms to launch *signal processing attacks* [4, 5]. Despite the aggravating threats, these new attacks could be defeated by integrating additional hardware [209] or extra signal processing procedures (e.g., voice activity detection, guard signals) [4, 90]. Unlike the aforementioned attacks, the *adversarial attack* aims to attack the deep neural networks (DNN), i.e., the computational core of an ASR system, which poses a major threat to modern ASR systems.

Adversarial attack: Adversarial attack was first proposed to attack image recognition systems [60, 160]. The attack operates by imposing unnoticeable perturbations onto the original image, thereby misleading the DNN to yield false classification. The inputs that enable such an attack are commonly referred to as *Adversarial Examples (AEs)*, which are composed of the original input with an unnoticeable perturbation. The ASR system with DNN models also inherits the susceptibility towards AEs.

Prior studies: Prior studies [12, 27, 44] demonstrate that attackers can generate adversarial audios to alter the DNN's prediction result with or without the prior knowledge of the DNN model. However, most of these attacks have not been successfully realized against real-world commercial devices, and their stealthiness is unverified. Recently, Chen et al. [31] successfully attack both open-source and commercial speaker verification systems over the air in a grey-box setting. Yuan et al. [204] embed their generated AE within songs to launch the attack, and they further adapt their attack in a black-box setting to subvert the ASR of most IVC devices [37]. Nevertheless, they fail to guarantee the attack success rate in the presence of user interference; and cannot promise to craft AEs quickly due to the training overhead of the substitution model. Meanwhile, two recent studies [72, 119] inventively propose the sub-second perturbation and spectrogram patch perturbation to attack open-source ASR systems, considering the victim user present during the attack. Even though they demonstrate the robustness and feasibility of their attack in the presence of environmental distortions, the proposed attacks are established on the assumption of complete knowledge

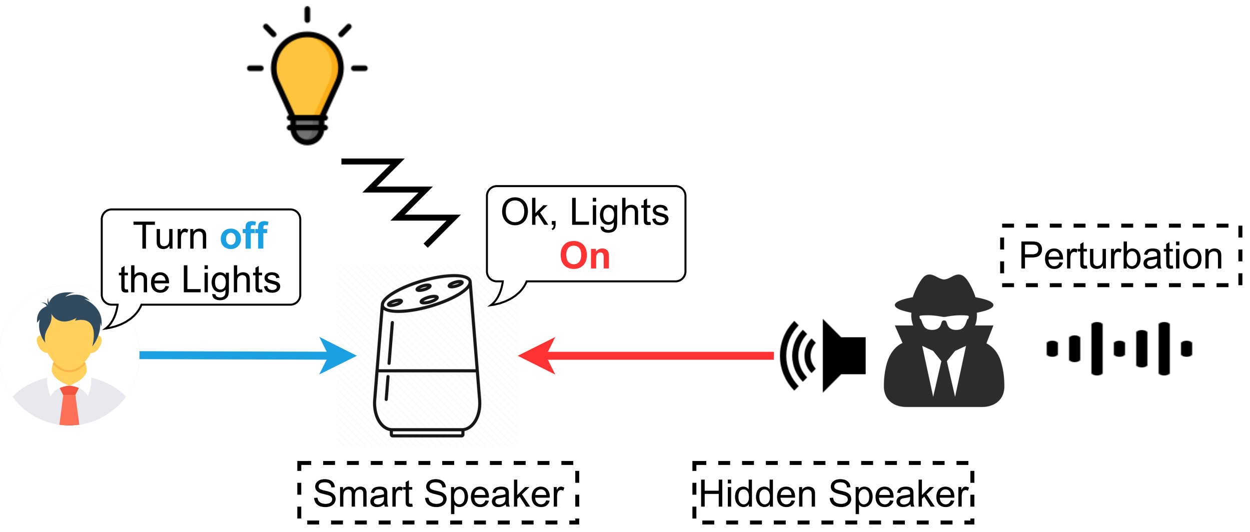


Figure 5.1: Attack scenario of PhantomSound

of the target ASR system. More recently, Zheng *et al.* propose a decision-based black-box attack by incorporating evolutionary algorithms to generate adversarial audios [218]. However, they still require to query the victim model extensively, which incurs substantial time and financial costs in a practical attack scenario.

Table 5.1 summarizes the existing adversarial attacks in terms of *victim systems' tasks*, *attacker knowledge*, *ability to attack quickly*, and *attack scenario*. The check mark symbolizes a successful attack under the given scenario, while the cross mark implies that the attack could not function or lacks efficacy in that particular scenario. For the *victim system's task*, SV indicates the speaker verification task while SR refers to the speech recognition task. We then taxonomize *attacker knowledge* into white-box, grey-box, and black-box, where grey-box implies the attacker can get the logits layer output [12, 31] or confidence score of all possible classes, and black-box indicates the attacker can only access the prediction label [37] of the target model. A white-box attacker, on the other hand, has complete knowledge (model architecture, weights of DNN parameters) of the target system. Next, we use online AE generation (Online GENR) to characterize whether the attacker can generate AEs or perturbations swiftly and complete the attack procedure in an online fashion. In fact, most existing studies assume the attacker has sufficient time to produce AEs offline. The last two metrics, Over Air and User Interference (User INT) suggest the attack scenario, where

Table 5.1: Comparison with other recent audio attacks.

Attacks	**SV SR**	**Grey Box**	**Black Box**	**Online GENR**	**Over Air**	**User INT**
Houdini [44]	SR	✓	✗	✗	✗	✗
C&W [27]	SR	✗	✗	✗	✗	✗
Adversarial [12]	SR	✓	✗	✗	✗	✗
Fakebob [31]	SV	✓	✗	✗	✓	✗
Comm. [204]	SR	✗	✗	✗	✓	✗
Devil's [37]	SR	✓	✓	✗	✓	✗
AdvPulse [119]	SR	✗	✗	✗	✓	✓
OCCAM [218]	SR	✓	✓	✗	✓	✗
SpecPatch [72]	SR	✗	✗	✗	✓	✓
PhantomSound	**SR**	✓	✓	✓	✓	✓

the former indicates an over-the-air attack, while the latter indicates whether the attack considers the user's interference (e.g., voice commands) during the attacks. To the best of our knowledge, *no existing attacks can attack commercial, closed-source ASR systems over-the-air with a limited time budget and user interference.*

PhantomSound: We propose a query-efficient black-box attack on commercial closed-source ASR systems and IVC devices. Our attack, called *PhantomSound*, can craft AEs and perturbations within a limited time budget and restricted query cost. Different from the previous work, the key idea behind PhantomSound is to regard the users' voice input as the command "carrier", while the phoneme-level perturbations are applied on the "carrier" to instantiate the attack.

Figure 5.1 depicts the attack scenario. First, the adversary records the user's command (any keywords such as "open", "on", "down"). Next, the adversary uses PhantomSound to query the accessible target models on the target IVC devices (e.g., the Google Cloud Speech-to-Text API for Google Home). Then, PhantomSound returns a perturbation that alters the prediction of the user's command.

During the attack, the adversary plays the perturbation via a hidden speaker at the same time when the user utters a voice command, which fools the smart speaker to operate improperly.

Challenges: Four major challenges arise during the design of PhantomSound.

- **Black-box attack:** It is difficult to attack a model without any prior knowledge. Existing grey-box/black-box attacks either assume attackers have the probability score of the target model [12, 44], or train a substitution model to approach the target model [37]. The existing attacks require a substantial amount of time to train a substitution model for the generation of AEs.
- **Speech model:** Different from black-box attacks on image processing [32, 38], ASR systems are known to have a more complicated model structure consisting of signal processing, filtering, acoustic model, and language model. As a result, attacking speech models requires different attack strategies to bypass the various components of the ASR models.
- **Query efficiency:** A successful black-box attack relies excessively on the effectiveness of queries. The adversary needs to iteratively update the AEs such that the effectiveness of the crafted AEs can be justified through querying. However, querying commercial ASR APIs is costly (e.g., \$0.00001/second for Google Cloud Speech-to-Text) and unable to bypass. Despite some efforts [32, 38] to reduce the number of queries, it still falls short of meeting the requirements for online generation of AEs.
- **Perturbation sync:** To successfully launch our attack, the adversary is expected to play the perturbation when he/she hears the victim's voice command. However, in a real-world scenario, the timing of perturbation is hard to control. Therefore, we need to tackle this problem by generating a near-synchronization-free perturbation [119].

Contributions: The contributions of this work are highlighted as follows.

- **New attack:** To the best of our knowledge, we are the first to achieve query-efficient black-box attacks on commercial ASR systems as well as IVC devices. We demonstrate the dangers of our attack over-the-air on 4 different commercial ASR APIs (i.e., Google Cloud Speech-to-Text, IBM Watson Speech to Text, Amazon Transcribe, and Microsoft Azure Speech Service) and 5 different IVC devices (i.e., iPhone with Google Assistant, Google Home, Microsoft device, Amazon Alexa, and IBM Wav-Air-API).
- **New finding:** We discover and formulate the unique boundary of commercial ASR systems for producing AEs. This non-contiguous decision boundary hinders previously successful attempts.

- **New techniques:** We propose PhantomSound, a phoneme-level searching method for efficiently crafting AEs to launch adversarial perturbation attack with the least number of required queries in comparison with other methods.

5.2: Background and Preliminary Study

In this section, we present the threat model of PhantomSound, as well as the assumptions and attack scenarios. Then, we introduce the fundamentals behind the adversarial attack and present the decision scheme of commercial ASR systems.

5.2.1: Threat Model

The adversary's goal is to mislead the IVC devices or VCS systems by injecting malicious commands. Prior to our work, there are two types of attacks that can achieve the same goal. The first attack [37] uses reverse-engineering models to imitate the commercial models and craft the offline AE in a white-box manner. The second attack [188] uses generative models to synthesize the victim's speech. However, the reverse-engineering attack necessitates a high volume of queries (as per Table 5.10) to construct the substitute model. It also demands updating the model in response to changes in the commercial API. This renders it expensive and inadequate in meeting the need for a real-time attack. Regarding the generative model driven synthesized attack, we assume the adversary has access to sufficient recordings of the victim for training purposes. However, in our specific situation, the attacker is expected to initiate the attack upon their first encounter with the victim. Furthermore, playing the synthesized speech outright is not a viable approach as the victim can hear it and potentially halt the attack.

Adversary's capability: We assume that the adversaries can place a hidden microphone to record the victim's voice. We assume that an adversary knows the targeted IVC devices and has access to their respective ASR API services (e.g., Google Cloud Speech-to-Text for Google Home or Google Assistant). Following other related studies [12, 37, 119, 204, 218], we also assume that the adversary is able to launch this attack via a hidden speaker or a compromised speaker in the victim's workspace/home.

Attack scenarios: The adversary will first collect the victim's voice commands, and then generate the AEs and perturbations swiftly only based on the transcription result of the target devices. Once the perturbations are crafted, the adversary can wait for the victim's next command and play the perturbation manually or automatically via existing keyword searching or voice detection mechanisms [11, 162]. Alternatively, the adversary may also play the perturbation repeatedly through hacked speakers, attempting to fool the target IVC devices when the corresponding target voice command was delivered.

In a real-world attack scenario, e.g., in a public space, an attacker may not have access to a large collection of victims' voices and may not have sufficient time to generate the perturbation offline. In this case, the attacker only has a very limited time window to subvert the victims' commands towards voice assistants. To successfully instantiate such an opportunistic voice attack, an attack approach with a timely and low complexity AE generation is highly desired.

User interference: Most existing attacks assume that the users will not perceive the AEs and will not interact with their voice assistants during the attack. However, when the users are speaking during the attack, most existing voice attacks will fail. In this research, we leverage the users' voice command as a carrier for the adversarial audio to launch the attacks more effectively and stealthily. Moreover, as advanced liveness detection algorithms [7, 118] have been used to differentiate between loudspeakers and humans with high accuracy, most existing audio attacks launched by loudspeakers can be easily detected. In our attack, however, since the human voice and the perturbation arrive at the same time, the liveness detection module of the voice assistant can be effectively bypassed.

5.2.2: Adversarial Attack

Adversarial attack aims to craft an AE $x_0 + \delta$, in order to deceive the model $f(\cdot)$ to make false prediction [160]. Take y_{pred} as the output of model, if $f(x_0 + \delta) := y_{pred} \neq y$ (y indicates the true label of input x_0), we suppose the attacker has launched an untargeted attack. If the perturbation is crafted intentionally for a specific target (denoted as y_t), the attack formalized as $f(x_0+\delta) = y_t \neq y$, is regarded as a targeted attack. The generation of AE can be formulated as an optimization problem as follows:

$$minimize \quad \mathcal{L}(x_0 + \delta) := \mathcal{D}(f(x_0 + \delta), y_t). \tag{5.1}$$

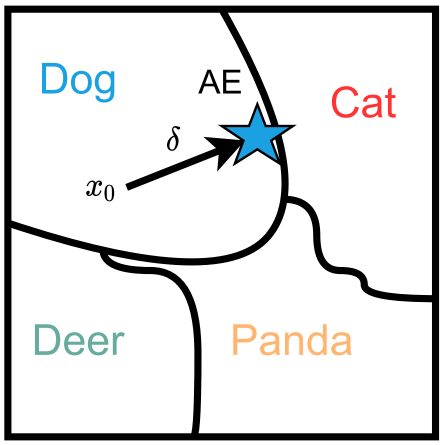
The goal of Eq. (5.1) is to minimize $\mathcal{L}(x_0 + \delta)$ under the constraint that $||\delta||_2 < \epsilon$, where $\mathcal{L}(\cdot)$ denotes the loss function, which uses a distance function $\mathcal{D}(\cdot)$ to measure the disparity between $f(x_0 + \delta)$ and y_t, $||\cdot||_2$ is the L2 norm, and ϵ is used to control the amplitude of perturbation. There are three main types of attacks depending on the prior knowledge of the victim models, listed as follows:

White-box: If the adversaries learn architecture and the parameters of the model, they can get the gradient of the loss function $\nabla\mathcal{L}(x)$ during the forward or backpropagation. The perturbation can be subsequently estimated using the inverse gradient [60].

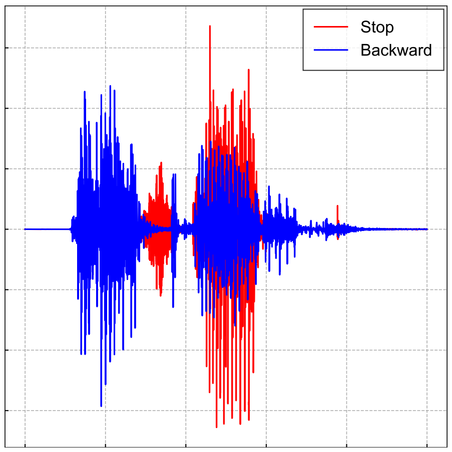
Grey-box: The model conceals its architecture and parameters from the public and only exposes the prediction scores $P = [p_0, p_1, \cdots, p_n]$ for a given input. The adversaries can formulate a loss function [26] $\mathcal{D}(P, P_y)$ (P_y is the one-hot encoding of y), and then track the changes of distance when tuning δ in multiple attempts. The changes in $\mathcal{L}(x)$ are utilized to estimate the gradient which will guide the attacker to update δ. The gradient estimation algorithms include Natural Evolution Strategy (NES) [99] and Zeroth Order Optimization (ZOO) [33].

Black-box: Compared to white-box and grey-box attacks, the black-box attack is the most challenging, in which the attacker only has access to the prediction label of the model. In fact, most of the commercial ASR systems and IVC devices are closed-source and only offer a final prediction. To successfully attack the black-box model, existing work either trains a surrogate model and transforms the problem into a white-box attack [135], or uses a significant amount of queries to search the decision boundary of the victim model [23, 32, 38]. Here, we focus on the query-based boundary-searching attack due to its flexibility and attack efficiency.

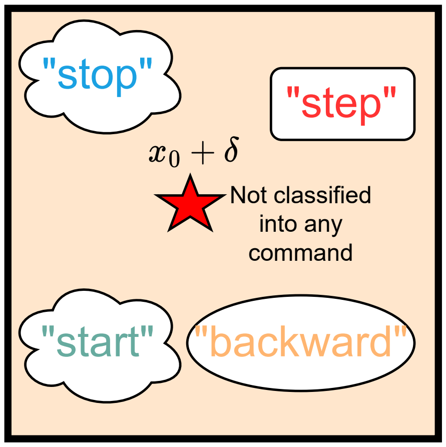
(a) A mixed image with cat and dog is recognized by Google Cloud Vision API [65] with 89% cat and 11% dog.



(b) Search decision boundary in black-box CV attack



(c) A mixed audio with “stop” and “backward” is rejected by Google Speech-to-Text API with no output



(d) The decision boundary for every class is non-contiguous for ASR system, every input in the middle will be rejected due to ambiguity

Figure 5.2: Observations of CV and ASR systems.

5.2.3: Black-box Audio Adversarial Attack

Compared with the black-box adversarial attack in other domains, the black-box audio adversarial attack has several unique features. In this section, we conduct a preliminary study in quantifying the behaviors of commercial ASR services.

Decision-based attack: Used for classification, a decision boundary is a hypersurface that partitions the sample space into several classes. Specifically, a well-trained DNN model uses the decision boundary to classify the incoming inputs. The main goal of the existing black-box attacks [23, 32, 38], or so-called decision-based attacks, is to find the decision boundary of the target model. Generally, to approach the precise decision boundary, they gradually perturb the input based on the query feedback, to find an AE on the verge of the decision boundary.

However, one assumption made by existing decision-based attacks is that the DNN classification model guarantees to return a prediction y_{pred} for any input x. As shown in Figure 5.2a, we merge a cat and a dog into one image and feed it into Google Cloud Vision API [65]. The classifier labels the image as a cat with very high confidence (89%) while the human brain perceives it differently. As shown in Figure 5.2b, the decision-based adversary [32] starts from a dog (x_0) and adds the proportion of a cat (δ) gradually to approach the boundary. The curves between classes in Figure 5.2b indicate the decision boundaries, where $\delta \in [0, 255]^{H \times W}$ denotes the perturbed image with the same shape as x_0. The contiguous decision boundary allows the DNN models to always output a result, while the result turns unreliable as it approaches the decision boundary.

Decision boundary of ASR: At first sight, it appears that the ASR systems would inherit the DNN's susceptibility to decision-based adversarial attacks. However, the unique characteristics of voice systems and DNN models make traditional decision-based attacks hard to succeed. Here, we conduct a preliminary experiment, in which we mix two voice commands "stop" and "backward" together (Figure 5.2c) to imitate the mixture of cat and dog images. Then, we submitted the mixed audio to Google Speech-to-Text API, which was rejected without any returns. The failed attempts indicate that the decision boundary of the ASR system is non-contiguous. As shown in Figure 5.2d, every voice command is surrounded by an exclusive boundary, and the audios outside of the boundary ranges will be rejected by the ASR systems.

This phenomenon implies that the perturbed voice queries may fail to solicit valid feedback from the ASR systems. Without feedback, it is difficult to determine the direction of the perturbation for approaching a target decision boundary. Based on this observation, we are motivated to design a new boundary-searching method to enable the decision-based black-box attack toward ASR systems.

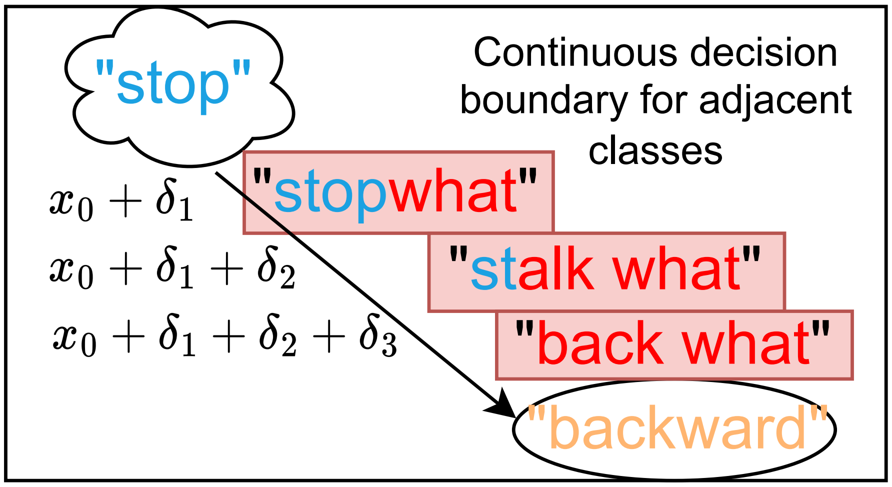

"stop"
Continuous decision boundary for adjacent classes
$x_0 + \delta_1$
$x_0 + \delta_1 + \delta_2$
$x_0 + \delta_1 + \delta_2 + \delta_3$
"stopwhat"
"stalk what"
"back what"
"backward"

Figure 5.3: Phoneme guided query

5.3: Attack Design

In this section, we present the system design of PhantomSound. We first introduce the phoneme-level boundary searching method to minimize the possibility of rejection by the ASR systems. Then, we formalize the attack as an optimization problem and illustrate the generation of AEs. Finally, to enhance the robustness of PhantomSound in real-world scenarios, we propose the weak synchronization scheme and over-the-air speech enhancement.

5.3.1: Phoneme-level Boundary Searching

Figure 5.2d shows the challenge in boundary searching to produce a proper AE. If the adversary randomly adds noise to “stop”, the ASR remains the “stop” decision when the noise is low and gives rejection while rising the noise power. However, if the adversary directly applies target “backward” to the benign audio, it results in audio (red start in Figure 5.2d) in the middle between two decision boundaries, hence giving no output.

Therefore, the reasons behind the rejection of queries can be attributed to two factors: 1) the added random noise will elevate the command’s noise level; 2) the boundary distance between two valid commands is too long to allow for an unnoticeable perturbation. To resolve these two problems, a novel idea is raised: “If we break the target “backward” into small pieces, then craft

AE with sub-targets which directly connect to the benign decision boundary with small pieces, and finally, we can craft the final AE with the target." Figure 5.3 depicts our attack design. Specifically, instead of directly adding "backward" on the "stop", we break the target "backward" into a series of phonemes. During crafting the AE, we randomly add the phoneme on the benign audio and check the prediction. If the ASR produces a word that is closer to our target, we keep the phoneme on the benign audio and search for a closer prediction in the next round. In our case, the "stop" adds perturbation phoneme δ_1 and is recognized as "stopwhat", then changes to "stalk what", and "back what", and finally reaches the target "backward". In every step, the AE achieves to sub-targets who is adjacent to the benign decision boundary, and gradually, the perturbation can be crafted by summing up all the small changes.

The basic idea of the proposed phoneme-level searching method is to perturb the original command along the direction of the target command while minimizing the distance between the original and the target ones.

Algorithm 1 presents the initialization procedure for generating the phoneme-level adversarial perturbation. Specifically, we first set the counter $s = 0$, and the initial distance between benign and target as ϵ = CER($f(x_0), y_t$). Next, we construct a phoneme set $D = \{ph_1, ph_2, ..., ph_n\}$ by breaking the target command, and then generate a random noise $v \in [0, 0.1]^l$ in line 4, where l is the length of original input x_0. Next, together with the v, a phoneme from D is randomly picked and injected at its corresponding position of x_0 in lines 5-6 to generate an AE x_*. The purpose of v is to increase the variance of the phoneme. For the targeted attack, if the x_* has a smaller distance to the target (line 7), we put the perturbation to the initial perturbation set $\tilde{P}$, then update the ϵ and x_0. For an untargeted attack, we can replace line 7 with "if $f(x^*)! = y$" to assure the ASR gives an incorrect prediction. The searching loop continues until it reaches a sufficient number of rounds K.

5.3.2: Perturbation Optimization

Even though Algorithm 1 generates proper perturbations for any voice commands, the amplitude of the perturbation may become overwhelming. Revisiting Eq. (5.1), to acquire the minimal per-

Algorithm 1: Phoneme-level Adversarial Perturbation Initialization

```
Input: The original audio $x_0$, the target label $y_t$, the phoneme clip samples
       $D = \{ph_1, ph_2, ..., ph_n\}$, the initial Character Error Rate(CER) $\epsilon$, the API service
       of black-box ASR system $f(\cdot)$.
Result: The initial perturbations set $\tilde{P}$
1  s = 0;
2  $\epsilon$ = CER($f(x_0), y_t$);
3  while $s < K$ do
4      $v$ = random $[0, 0.1]^l$;
5      $\delta = v + rand(D)$;
6      $x^* = x_0 + \delta$;
7      if CER($f(x^*), y_t$) < $\epsilon$ then
8          Put $\delta$ into $\tilde{P}$;
9          $\epsilon$ = CER($f(x^*), y_t$);
10         $x_0 = x_0 + \delta$;
11     else
12         $s = s + 1$;
13     end
14 end
15 return $\tilde{P}$
```

turbations, we need to gradually increase the perturbation power. However, due to the black-box setting, the gradient is inaccessible. As a result, we use Sign-Opt [38] to estimate the gradient, since Sign-Opt has achieved superior performance with the least number of queries, as written below:

$$\nabla \mathcal{L}(x) \approx \sum_{q=1}^{Q} sign(\mathcal{L}(x + \sigma\mu_q) - \mathcal{L}(x))\mu_q, \tag{5.2}$$

$$sign(\mathcal{L}(x + \sigma\mu) - \mathcal{L}(x)) = \begin{cases} +1, & f(x + \sigma\mu) \neq y_t \\ -1 & f(x + \sigma\mu) = y_t \end{cases} \tag{5.3}$$

where x is the general representation of $x_0 + \delta$, q and Q denote the noise index and the total number of noises respectively. σ is the search variance and μ is the noise. The key idea of Sign-Opt is to search the gradient space using the natural evolution strategy. Since $\mathcal{L}(x)$ is unknown, Sign-Opt queries $f(\cdot)$ in Eq. (5.3). The feedback of the target model can be collected to measure the number of wrong predictions. The result will be used to guide Eq. (5.2) in searching for the gradient of

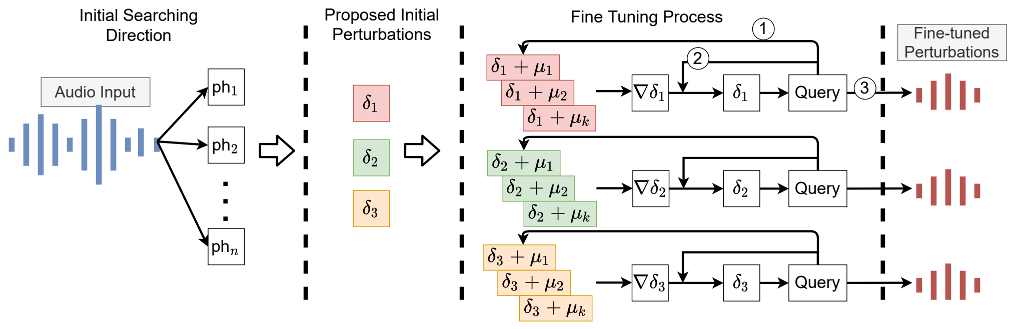


Figure 5.4: Adversarial perturbation generation

$\mathcal{L}(x)$.

Query-efficient fine-tuning: The perturbation generation typically requires ∼5k queries to craft an AE [32, 38]. To further reduce the cost of queries, we design a query-efficient AE generation scheme to greatly reduce the query number.

By carefully examining the Eq. (5.2), we realize that the gradient estimation step depletes most of the queries. Suppose $Q = 50$, then it uses 50 queries to catch the $f(\cdot)$ result and estimate gradient according to Eq. (5.3). However, Sign-Opt [38] uses the estimated gradient only once for updating x, with a small update learning rate, while most of the gradient computations are wasted. In our design, we estimate the gradient once, then apply the estimated gradient multiple times to update the δ until it does not satisfy our attack goal, then do the gradient estimation again.

The workflow of our proposed query efficient phoneme-level adversarial perturbation generation is shown in Figure 5.4. There are three major steps to generate AEs and perturbations: *searching, proposing*, and *fine-tuning*. In the searching and proposing phases, unlike the prior study [38] which only searches for random noise and keeps the shortest initial perturbation while discarding others, we reserve all the perturbation candidates to increase the generation speed. In the fine-tuning phase, we optimize all the proposed perturbations through gradient estimation. Note that there are three paths from the Query block: ② is used to update the perturbation consecutively until it cannot be further optimized. Then, we will re-calculate the gradient (①). Once the power of perturbation is lower than ϵ, we add it into the perturbation set P (③).

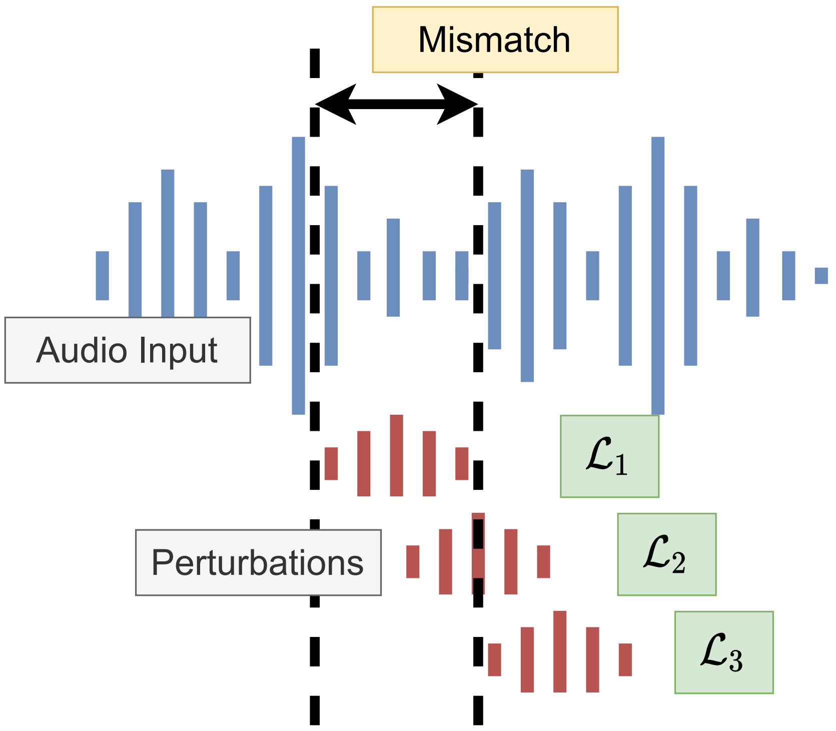


Figure 5.5: Perturbation mismatch during an attack.

5.3.3: Weak Synchronization Design

Considering the adversary needs reaction time to play the perturbation, the generated perturbations are demanded to be robust against the mismatch of insertion positions. To realize such an attack, we seek to minimize the average loss instead of the instant loss. That is, we take the impact of mismatch into consideration and expect the comprehensive loss to be minimized. Mathematically, the average loss can be expressed as follows:

$$\overline{\mathcal{L}(x)} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_i(x), \tag{5.4}$$

$$\mathcal{L}_i(x) = \mathcal{L}(x + c\tau), \tag{5.5}$$

where τ represents the mismatch interval, c controls the length of a mismatch period, i indicates the id of related losses, and N is the number of involved $\mathcal{L}$. To minimize the average loss, we can refer to Eq. (5.2) and Eq. (5.3) to estimate $\nabla\overline{\mathcal{L}(x)}$ by computing $\nabla\mathcal{L}(x + c\tau)$. The drawback of the average loss gradient estimation is that it costs $N\times$ more queries to perform the gradient estimation. The length of phonemes in D varies from $50ms$ to $300ms$, and one-word duration is ranging from $281ms$ to $387ms$ according to the report [168]. We expect that the phoneme-level perturbation can be plugged within the duration of one word, otherwise, it will be difficult to maintain the minimal $\mathcal{L}$ especially when a delayed perturbation arrives. In this chapter, we set the $N = 4$ and $\tau = 100ms$. Figure 5.5 depicts the perturbation mismatch scenario: when crafting the first red perturbation, we gather the other losses by the same perturbation but with a different time delay. In the figure, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$ correspond to $c = 0$, $c = 1$, and $c = 2$.

5.3.4: Over the Air Attack Robustness

Besides the weak synchronization feature, the attack robustness is another important feature of PhantomSound. Existing work models the acoustic signal propagation to compensate for the propagation loss over the air [144]. But the heavy computation prevents them from being adopted in real time attack. Also, the quality of perturbation relies on the speaker's amplifier, and the additional distortion on such small perturbation is hard to model. Inspired by the prior work [119] who sets a frequency filter to guarantee the generated perturbation is ranging from 50-8,000 Hz. To guarantee the effectiveness of PhantomSound over the air, we follow their approach on configuring a frequency filter to mitigate the uneven frequency response caused by the hardware imperfection of the speaker, thereby enhancing the attack robustness.

5.4: Evaluation

In this section, we first introduce our benchmark experimental setting to generate AEs and perturbations. Then, we evaluate PhantomSound thoroughly to validate its feasibility and robustness. Moreover, we measure the impacts of different parameters in tuning a successful attack. Our attack is successfully launched on four different ASR service APIs, and the five popular commercial IVC devices. We further conduct an user case study in section 5.4.8. This section describes the results in detail.

5.4.1: Target Model Selection

Since we are developing a general approach to generate perturbations to attack closed-source ASR systems and commercial devices, we will examine the effectiveness of AEs and perturbations on the most popular IVC devices available on the market. Specifically, we select Google Home (G-H), Google Assistant (G-A), Amazon Echo, Microsoft Cortana, and IBM WAA[6] as target IVC devices. Moreover, we target their respective ASR APIs, namely, Google Cloud Speech-to-Text API, Microsoft Azure API, Amazon Transcribe API, and IBM Watson API. As for Apple Siri, since there is no online speech-to-text API service available from Apple, we cannot perform PhantomSound due to the lack of querying feedback from its ASR system. For all the target systems, we only receive the hard label of the querying input from their APIs.

5.4.2: Metrics

We use the following metrics to quantify the effectiveness of our attack: *(1) Success Rate:* this metric represents the ratio of successful attacks and the total attempts. For an untargeted attack, as long as the AEs and the perturbations alter the prediction of the original input, we count it as successful. For a targeted attack, we report success only when the prediction matches the targeted class. *(2) Average queries per command:* we use the number of queries to imply the cost and speed of AE generation. Specifically, we measure how many queries it needs to craft a perturbation. This metric is calculated by the total number of queries over the number of crafted AEs/perturbations. *(3) L2 Distortion:* the L2 distortion $||\delta||^2$ indicates the size of perturbations. Prior to the launch of a physical attack, we can measure the distortion value by summarizing the squared amplitude of the generated perturbations. Note that the perturbation $\delta \in [0, 1]^l$ and the initial phoneme-level distortion ranges from 50 to 1,600 depending on different phonemes, which will be optimized after the perturbation fine-tuning as shown in Section 5.4.5. *(4) False Accept Rate:* the false accept rate is measuring the probability of that the attacks can be false accepted by the liveness detection

[6]WAA represents "Wav-Air-API". As IBM does not own a commercial voice assistant device, we record and replay our AEs over-the-air, and transcribe them with IBM Watson API. This process, named as WAA, simulates an IVC device that is integrated with an IBM Watson API [37].

methods. We use this metric to evaluate the ability of our attacks to bypass the existing defense methods (e.g. liveness detection) compares to the existing attacks. The higher false accept rate we achieve, indicating the more dangerous of attack is, to bypass the existing liveness detection methods.

5.4.3: Dataset

The dataset we choose as original input is speech commands v0.02 [186] released by Google Brain. This dataset is designed to validate the keyword detection capability of DNN models. It contains 105,829 utterances of 35 common one-word commands (e.g., "yes", "learn", "stop"), which is recorded from 2,618 volunteers. To validate the effectiveness of PhantomSound on a longer command, we record 10 longer commands (partially listed in Table 5.4) from a volunteer.

For the phoneme dataset, we expect to obtain all 44 pure English phonemes with flexible duration. Existing speech datasets (e.g., Arabic Speech Corpus [81], TIMIT [1]) include the annotations of phonemes, but it requires extra efforts to extract individual phonemes with different duration from the speech audio. Besides, PCVC dataset [2] only involves 12 volunteers, and scikit phoneme dataset [146] only contains 5 vowels. To construct a phoneme dataset with a diverse set of speakers, we use 200 different audios from 200 speakers in speech commands v0.02, remove the silence in the recordings, and randomly cut audio clips with a duration between 50ms to 300ms. This phoneme processing step follows that of the scikit phoneme dataset [146], which results in 453 audio clips in total.

Table 5.2: Dataset description ("unique cmds" refers to the number of unique target commands, and "total audios" refers to the total number of (adversarial) audios that lead to the target commands).

	Phone.	Cmd.	Untargeted	Targeted
Unique cmds	-	45	1785	64
Total audios	453	300	6,219	216

Table 5.2 records the number of involved data including phonemes, commands, untargeted perturbations, and targeted perturbations. We use 35 one-word commands from the speech commands v0.02 dataset, along with 10 self-recorded long commands to build a command dataset with

45 different commands, including 300 audios in total. Then, we apply the proposed algorithm to randomly generate AEs and perturbations for an untargeted attack, resulting in 1785 different commands and 6,219 adversarial audios on 4 different commercial APIs. For the targeted attack, we attempt the perturbation of keywords, and generate 64 target commands with 216 adversarial audios.

5.4.4: Experiment Setting

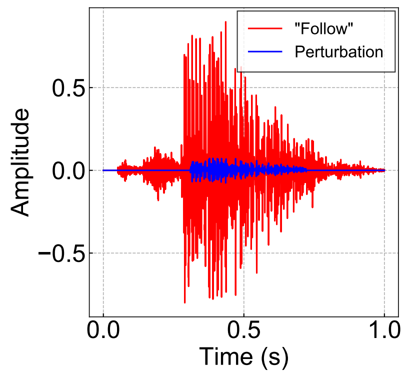
We conduct the experiments on a desktop with Intel i7-7700k CPUs, 32GB RAM, and 64-bit Ubuntu 18.04 LTS operating system. The experiments are performed at three locations with different noise floors. We use three loudspeakers, including LG monitor built-in speaker (at the apartment), an SADA D6 home small speaker (at the lab), and an Samsung S9 phone (at outdoor), to transmit AEs (i.e., AE attack) and perturbations (i.e., perturbation attack) to the victim devices. Figure 5.12a demonstrates the attack scenario: the victim speaks commands into a smartphone or Google Home mini, while the attacker plays the perturbation through a speaker.

5.4.5: Attack Performance

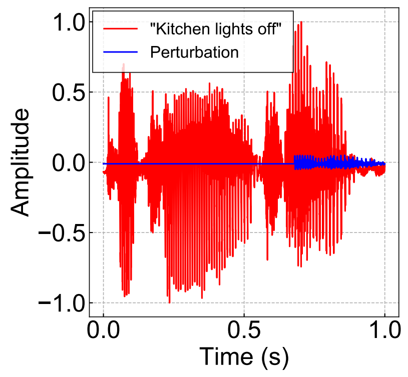
We first evaluate the functionality of AE generation in PhantomSound. The purpose of this evaluation is 1) to demonstrate that the perturbation amplitude is negligible compared with the input, and 2) to prove the query efficiency of our phoneme-level searching algorithm. Then, we conduct the physical attack and validate the robustness of our attack over the air.

Attack Over-the-line: We first evaluate the attack by targeting the ASR APIs. The adversarial audios are directly supplied to the online APIs. We randomly select 20 adversarial audios from every command, and then perform the untargeted attack by searching for 100 epochs ($K = 100$ in Algorithm 1). Then, the generated perturbations are optimized to suppress their power. In the end, we obtain 148 AEs and perturbations from $\sim$44k queries ($Q = 30$ in Eq. (5.2)), i.e., 301 queries per AE on average.

To evaluate the perturbation amplitude, we randomly pick two examples from the generated perturbation as shown in Figure 5.6. We can see that the crafted perturbations have a negligible



(a) One-word commands



(b) Command phrases

Figure 5.6: Comparison of input and perturbation amplitudes.

power profile compared with the input regardless of the length of commands. Moreover, the duration of perturbation is shorter than the input, which makes it possible to conceal the presence of perturbation. Table 5.3 summarizes the results of the untargeted attacks toward 4 types of commercial APIs. We observe that every command can be altered into at least two false commands. While some of the false predictions are harmless, the attack can almost certainly *invalidate* the victim's command. Moreover, in certain cases, some perturbations can lead to a contrary response from voice assistants (e.g., "right" to "wrong" in Amazon Transcribe API, "right" to "no" in Microsoft Azure API). Considering the number of queries for generating one perturbation, the Google Cloud Speech-to-Text is reported to be the most resilient API under our attack, as it requires the most queries.

To further comprehend the query effectiveness, we conduct an additional experiment to validate the sensitivity of different APIs in terms of request rejection rates. The result shows that Google API is most sensitive as it refuses to respond to an unclear input, while the Amazon transcribe always responds to any inputs. Table 5.4 records the targeted attack results towards a longer input. The results show that our phoneme-level searching method is capable of finding the specific perturbation that could mislead the APIs to return a target result. Note that the average query amount increases dramatically in the targeted attack case, which is anticipated because the target need to be achieved by multiple round perturbation searching (line 7-10 in Algorithm 1). It is also notewor-

Table 5.3: Untargeted attack results.

Cmds.	Google Cloud	MS Azure	AMZ Trans.	IBM Watson
”down”	”damn” ”done” ”does”	”town” ”one” ”south”	”done” ”dine” ”drive”	”Downer” ”Done” ”Drone”
”follow”	”fallout” ”farm” ”four”	”fallout” ”fall over” ”learn”	“no” ”for sure” ”phone”	”fallen” ”fall over” ”fall”
”forward”	”forewarn” ”for eyes” ”for work”	”work” ”for” ”ford”	”what” ”work”	”for” ”four” ”for all”
”yes”	”yeah” ”yeah!” ”yet”	”file” ”4” ”On”	”yeah” ”yes..” ”right”	”yeah” ”yet” ”hi”
”right”	”Rite Aid” ”write” ”read”	”no” ”go” ”trade”	”write” ”run” ”wrong”	”run” ”ray” ”left”
Queries	345	251	215	314

Table 5.4: Targeted attack results

Command		Query			
Input	Target	Google Cloud	MS Azure	AMZ Trans.	IBM Wat.
”turn **right**”	”turn **left**”	1,895	1,128	1,421	1,487
”kitchen lights **off**”	”kitchen lights **on**”	1,754	857	933	1,377
”call **mom**”	”call **911**”	-	1,421	1,125	-
”**read** my message”	”**delete** my message”	2,342	1,520	1,436	1,781
Average Queries		1,997	1,232	1,229	1,548

thy that our targeted attack cannot guarantee finding a successful perturbation under any arbitrary inputs (e.g., Google Cloud fails to craft AEs for “call 911”).

Table 5.5: Comparison for Untargeted Attacks

Models↓	Ours	white box [27]	score based [33]	brute force [23]
DS 1 [84]	**185**	**90**	206	∞
DS 2 [16]	**226**	**75**	197	∞

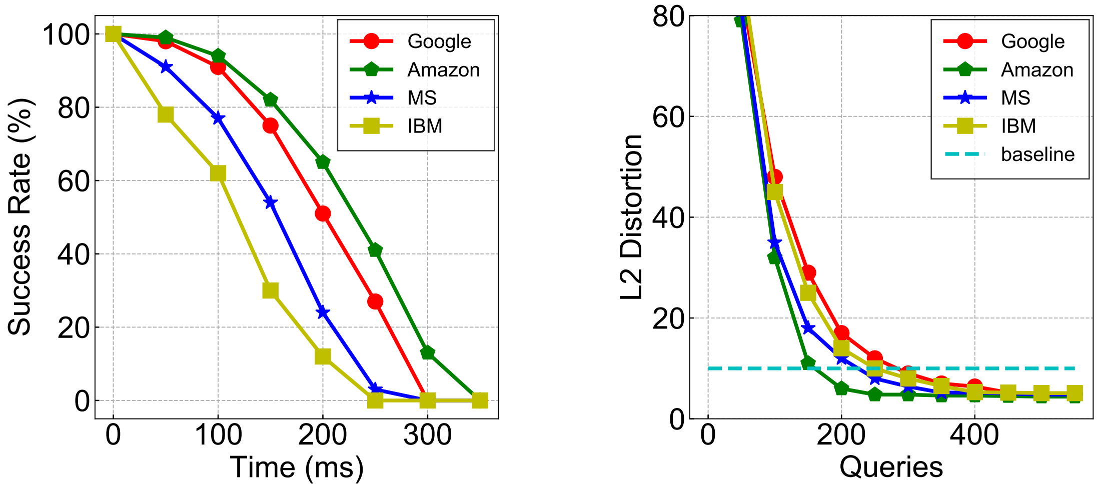
Query efficiency comparison on known models: To validate the benefits of introducing phonemes to guide the optimization direction, we implement 3 different attacks on two known models. By attacking two ASR models (DeepSpeech 1/DS 1 [84] and DeepSpeech 2/DS 2 [16]) with different prior knowledge and method, we find that PhantomSound achieves comparable query efficiency with the grey box setting, with 100% attack success rate. The result is summarized in Table 5.5. Given the same 10 benign commands, we use the 4 attacks to generate untargeted AEs with the same L_2 distortion. We record the average number of queries for different prior knowledge of the victim model. Compares to the white box attack, which can fine-tune in <90 queries, PhantomSound requests 200 queries to craft an AE, which is close to the queries of a score-based attack. This result indicates that our strategy such as 1) using phoneme to initialize perturbation 2) Query-efficient fine-tuning is working well, and performing similar results with less information (e.g., confidence score). It is noteworthy that the brute force decision boundary search method doesn't work for attacking the ASR model. Because this method initializes a random noise and retrieves model gradients by altering the noise. However, this noise can never be fine-tuned while the victim model produces an empty label to it, resulting in an infinite number of queries.

Table 5.6: Comparison for Targeted Attacks

Attacks	Knowledge	Queries	SR
Carlini [27]	Gradient	$\sim$1,000	100%
Houdini [44]	Gradient	$\sim$1,000	100%
Devil's [37]	Conf. Score	$\sim$1,500	100%
OCCAM [218]	Final decision	$\sim$30,000	100%
Ours	**Final decision**	**$\sim$1,500**	**68%**

Query efficiency comparison for targeted attacks: We compare the number of required queries with four existing attacks in Table 5.6. The white-box attacks (Wb) [26, 44] require the least amount of queries ($\sim$1,500). With the knowledge of confidence scores of API's decoding results, the Devil's Whisper [37] utilizes a surrogate model trained with around 1,500 queries to attack the APIs. In the scenario when an attacker can only access the final decision of the query API, PhantomSound needs $\sim$1,500 queries (comparable with the white-box setting) to craft a targeted perturbation. Compared with a recent black-box attack OCCAM [218], we reduce the number of queries by 95%. However, due to the limitation of phoneme length and diversity, we sacrifice the success rate to achieve high query efficiency.

Weak synchronization: Before evaluating the physical attacks, we investigate the effectiveness of the proposed weak synchronization design. In this experiment, we manually add mismatch delays between input x_0 and the generated perturbation to craft mismatched AEs. We then use the mismatched AEs to query the APIs and measure the attack success rate. Figure 5.7a displays the result, from which we can see that, after using the average loss, although we expect the weak-synchronization works within $400ms$ (detailed in Section 5.3.3), this design is only partially effective, because the success rate drops steadily with the increasing mismatch time. Moreover, we show the tendency of L2 distortion w.r.t. the number of queries in Figure 5.7b. The baseline denotes an L2 distortion of 10, which is proven unnoticeable by two volunteers when AEs are played using an LG monitor with a medium volume.



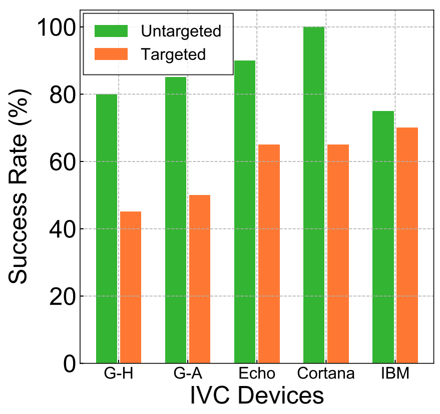
(a) Weak synchronization

(b) L_2 Distortion vs. No. of queries

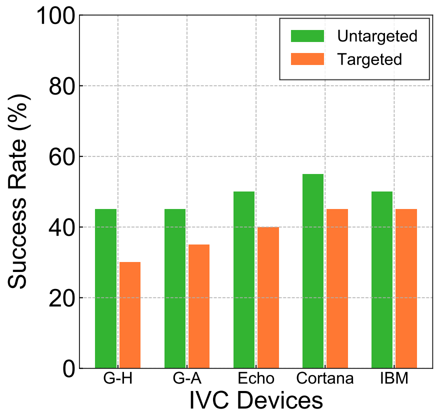
Figure 5.7: Evaluation of AE generation process.

Attack over-the-air: The over-the-air attack evaluation aims to prove the robustness of PhantomSound.

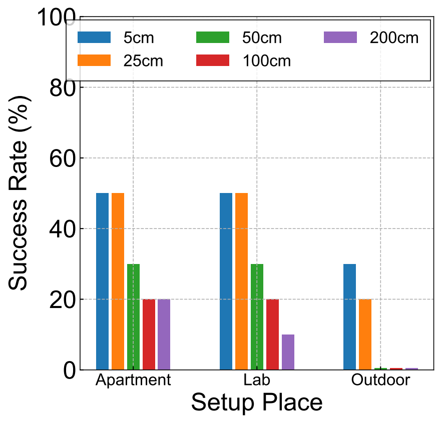
To attack commercial APIs, we play the valid AEs and perturbations (which attack successfully in over-the-line scenarios) via a SADA D6 speaker, and record it by iPhone 12 Pro, the recordings are sent to the commercial API for evaluation. The attack distance is set to 50cm. For each attack, we choose 5 AEs to play 5 times and get the average success rate. We report the result in Table 5.7. From our observations, it is apparent that in the context of a targeted attack, our method attains approximately a $\sim$ 80% success rate in attacking over-the-air commercial ASR APIs by directly playing the audio adversarial example (AE). When the attack is synchronized with the victim's



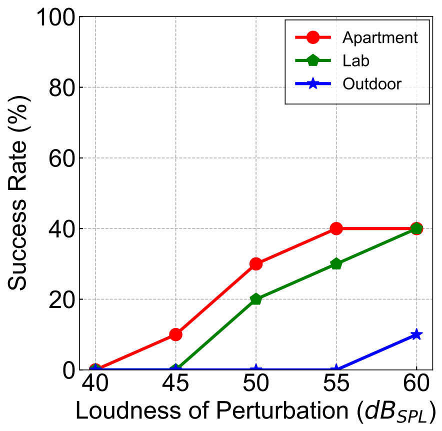
(a) AE attack



(b) Perturbation attack



(c) Distance



(d) Loudness

Figure 5.8: AE generation results.

Table 5.7: Over-the-air attack API baseline

APIs		Google Cloud	MS Azure	AMZ Trans	IBM Wat.
Targeted	AE	76%	80%	80%	84%
	Pert.	68%	72%	72%	76%
Untargeted	AE	100%	100%	100%	100%
	Pert.	72%	80%	80%	92%

speech, the perturbation attack exhibits around a $\sim 72\%$ success rate. On the other hand, when it comes to untargeted attacks, our adversarial examples (AE) and perturbation methods achieve impressively high success rates. They misdirect the victim's input with a 100% and approximately 81% success rate, respectively. Next, we follow the same setting to attack commercial IVC devices.

The result in Figure 5.8a uncovers the success rate of playing AEs directly. Among all the tested IVC devices, Microsoft Cortana is most vulnerable against the AE attack, while the Google series products (e.g., Google Home, Google Assistant) show the most resilience against the targeted AE attack. Overall, the success rate of an untargeted attack is higher than that of a targeted one, i.e., the former reaches ∼80% success rate and the latter stays around ∼50%. With the perturbation attack, Figure 5.8b reveals a relatively low success rate. Similarly, compared to the targeted perturbation attack, the untargeted attack has a higher success probability, achieving around 45% success rate on average. Nevertheless, the success rate can be further improved via multiple repeated attempts. We also summarize the success rate compared to prior black-box attacks in Table 5.8.

Table 5.8: Comparison with other real-world attacks

Target	Google Cloud	MS Azure	AMZ Trans.	IBM Wat.	Google Home	Google Assit.	MS Cortana	AMZ Echo
Devil's [37]	10/10	10/10	4/10	10/10	9/10	10/10	10/10	10/10
Danger [215]	-	-	-	-	15/100	-	-	69/100
Ours	19/25	20/25	20/25	21/25	11/25	12/25	16/25	16/25

Upon comparison with the Devil's attack [37], it is evident that our attack method yields a marginally lower success rate against the APIs, with the exception of the Amazon Transcribe API. Considering the IVC devices, the Devil's attack tends to be more effective at similar SNR levels. For the Danger attack [215], we have displayed their success rate derived from their "voice squatting" attack, where the victim's command is misinterpreted to initiate the attack skill. A comparison reveals that our attack technique yields comparable success rates when targeting Amazon Echo, and even demonstrates superior performance when used to attack Google Home.

Time cost: Different from the prior works that require a substantial amount of time to craft AEs offline, PhantomSound enables much faster AE generation. Such a fast generation feature is essential in practice, when the attackers only have a limited time budget to instantiate the attack.

In the experiment, we record the latency for querying 4 different commercial APIs to get the results. The results are presented in the first row of Table 5.9, which show that 3/4 of APIs could return a result in seconds, except Amazon Transcribe API. The Amazon API has to interact with Amazon Web Service and Storage bucket, which spends a longer period for the results to return.

We then compute the total time needed for perturbation generation, by multiplying latency with the number of queries (shown in Table 5.3, 5.4). Our result shows that PhantomSound can generate

Table 5.9: Latency for perturbation generation

Time Consumption	Google Cloud	MS Azure	AMZ Trans.	IBM Wat.
Latency (s)	0.29	0.58	26.31	1.35
Untargeted (min)	1.67	2.43	94.3	7.1
Targeted (min)	9.65	11.9	539	34.8

a perturbation for both the targeted and untargeted attacks in minutes with the exception of Amazon API, while the targeted one takes longer. Note that we take the L_2 distortion into consideration during the time cost computation, however, if the attacker ignores the impact of the perturbation loudness and uses the intermediate perturbation, the generation time can be further reduced.

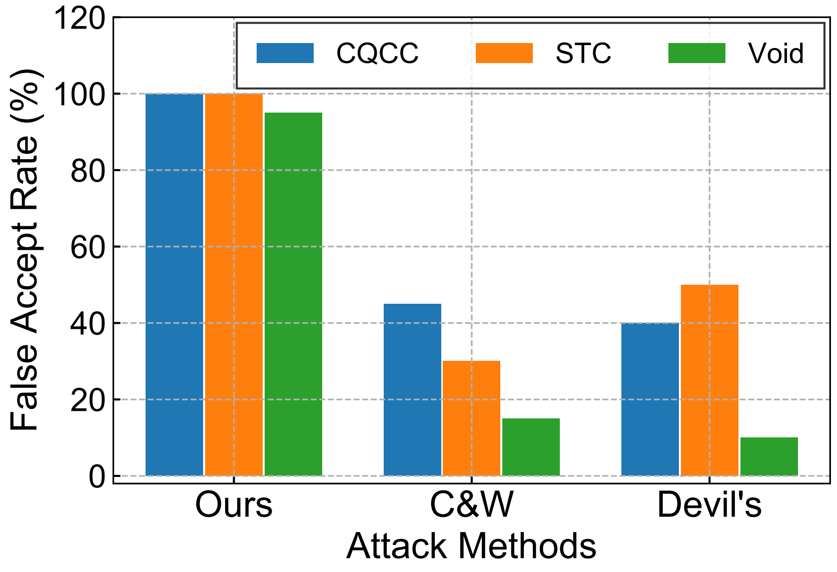


Figure 5.9: Attacks vs. liveness detection defenses

5.4.6: Ability to Bypass Liveness Detection

Compares to the existing physical adversarial attacks [37,204,218], PhantomSound relies on the benign commands spoken by the user. Although this attack setting requires extra effort to synchronize the perturbation and the user's benign speech, it brings potential benefits to bypassing the defense mechanism. For example, recent works [7, 73, 110, 112, 118, 123] proposed liveness detection approaches can differentiate the source of sound (human or machine) with high accuracy. Therefore, the conventional adversarial attacks that are launched solely by loudspeaker [37, 204, 218] have a higher probability to be defended by those liveness detection methods. In contrast, our attack is designed to launch with the user's speech, leading to a more dangerous threat to the liveness detection defenses. To validate the performance of PhantomSound over different defense mechanisms, we reproduce three liveness detection algorithms, CQCC [110], STC [112], and Void [7]; For comparison, we implement C&W attack [27] and Devil's [37]to attack with liveness detection algorithms. To conduct this experiment, we follow the settings described as follows:

Ours: We play our perturbation when the user gives the command, and record it with a smartphone. Then, we run three liveness detection algorithms to detect the sound source.

C&W [27]: We play the AEs that are generated by this attack, and then record with the same smartphone and run liveness detection algorithms to defend it.

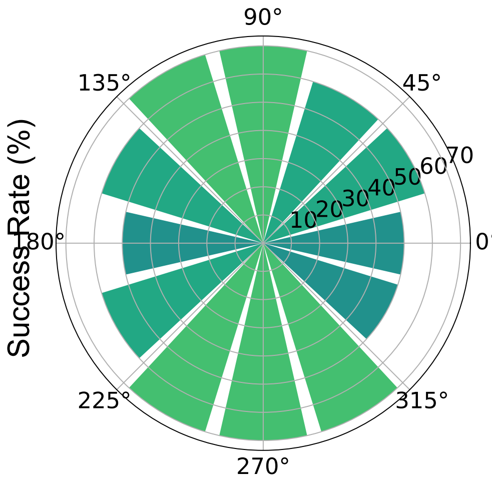
Devil's [37]: We play the AEs provided from the chapter's demonstration website, and then record it with the same smartphone, followed by the same liveness detection procedure.

For our attack and the C&W attack, we use 20 different perturbations/AEs to attack the liveness detection model; As for the Devil's attack, since we can only collect 10 AEs from the demonstration website, we use 10 AEs to attack the liveness detection model. We present our result in Figure 5.9. It is evident to show that our attack can bypass the three liveness detection models, resulting 95% to 100% false accept rate. In contrast, the other two attacks have a very low chance to counter the Void [7] detection with less than 15% FAR. Even for conventional liveness detection methods (e.g., CQCC and STC), the existing attacks that use complete AEs also have a low probability (40%) to attack successfully.

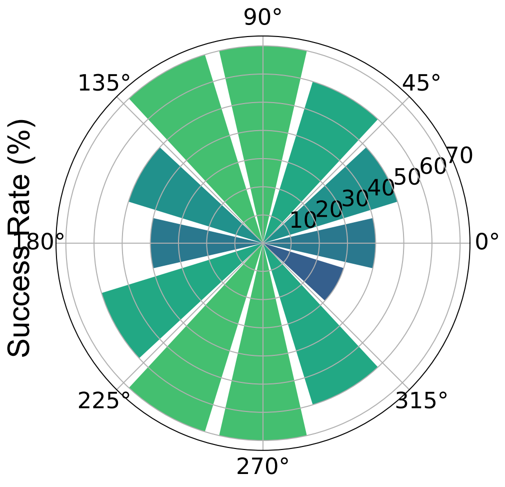
5.4.7: Impact of Practical Factors

To investigate the critical factors that may affect the success rate of PhantomSound, we evaluate the perturbation attack under different environments (e.g., apartment, lab, outdoor). The ambient noise level for the aforementioned places are 39.8 dB_{SPL} (apartment), 41.2 dB_{SPL} (lab) and 58 dB_{SPL} (outdoor), respectively.

In this experiment, we play a crafted perturbation of "turn right" 10 times, attempting to transform the prediction into "turn left", and the volume of perturbation is 60 db_{SPL}. We then record the success rate under different circumstances. Figure 5.8c demonstrates the impact of attack distances, i.e., the closer the adversary is, the higher success rate he/she achieves, which is unsurprising given that our attack relies on the successful delivery of the perturbation. The relatively short attack distance is in fact a common limitation reported by the existing work [37, 119, 204]. However, the attacker can further extend the attack distance by increasing the speaker's volume (though it could make the perturbation more noticeable) or utilizing a speaker array [141]. Next, we provide the results on how the loudness factor could affect the attack performance in Figure 5.8d. We can see that the success rate improves with the increasing perturbation loudness. This result also coincides with the prior work [119]. In an outdoor environment, it is suggested that the adversary enhance the attack robustness by amplifying the perturbations. Due to the higher noise level outdoors, the phoneme-like perturbation can still be hard to perceive.



(a) Attack Angle iPhone 12Pro



(b) Attack Angle Mi 8 Lite

Figure 5.10: Attack with different angles.

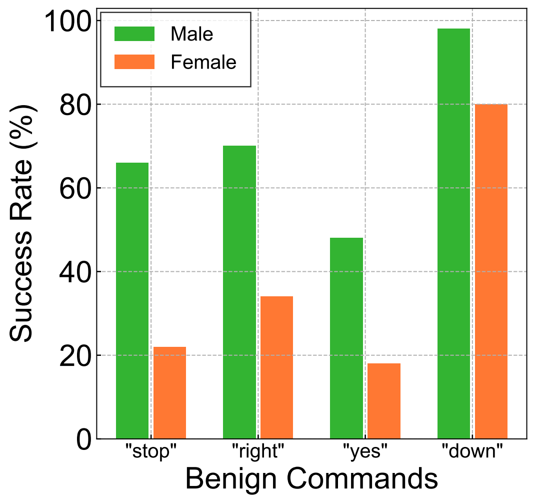
Impact of attack angles: Besides the attack environments and the distance, the attack angle can also alter the attack performance. We evaluate our attack by playing AEs to two smartphones in 12 different directions (from 0 degrees to 360 degrees, with 30-degree intervals). This experiment is conducted in Lab environment and attacks the google assistant on the smartphone. We play 10 AEs in every direction with $60dB_{SPL}$, and record the success rate of the untargeted attack. We report our result in Figure 5.10. We find that our attack has the best performance when the adversary is facing or back to the smartphone. While attacking through the side direction (e.g., 0 degrees when the adversary is parallel to the victim), the success rate is impaired. We observe the same trend on two smartphones. This result indicates that the microphone arrangement and its direction will lead to audio information loss. Unfortunately, the low power of our perturbation is hard to be sensed with the audio loss, therefore causing a low success rate in the side direction.

Impact of different victims: In the attack preparation period, every perturbation is crafted based on a specific command from a specific speaker. However, the adversary may use the crafted perturbation on the previous victim to attack the current victim. Here, we evaluate the capability of PhantomSound to attack different speakers. First, we obtain 4 perturbations from speaker #1 (male), which convert the benign commands ”stop”, ”right”, ”yes”, and ”down” into 4 target commands ”backward”, ”left”, ”no” and ”song” respectively. Next, we randomly select 100 speakers (50 males and 50 females) who are not speaker #1 from the speech commands v0.02 dataset, and inject the perturbations into their benign audio samples. For the targeted attack, if the benign commands are successfully interpreted as the target, we classify it as successful. For the untargeted attack, any case where the benign commands are misinterpreted is considered successful. The result is present in Figure 5.11.

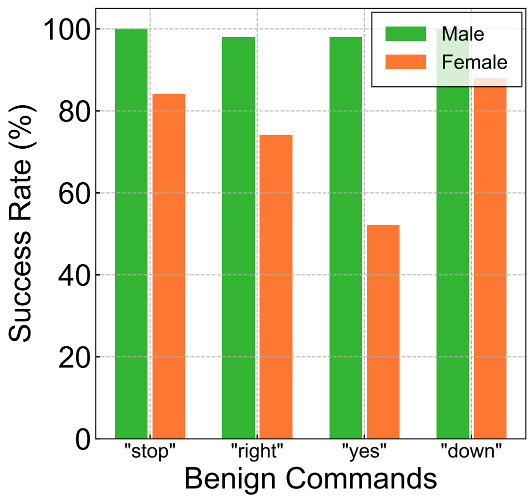
The result indicates that, for targeted attacks, the attack success rate is dependent on the benign samples. The success rate exceeds 50% when the target is of the same gender, but it falls below 40% when targeting different genders. Regarding the untargeted attacks, the perturbations demonstrate robust transferability for attacking various speakers. The average success rate is notably high, reaching 98% for males and 74% for females.

5.4.8: User Study

To evaluate the stealthiness of perturbation in a real-world attack, we conduct an online/in-person user study to investigate the users' perception level of PhantomSound. In our study, 20 volunteers are involved, and they are requested to hear 6 crafted perturbations across 4 different distances.



(a) Targeted attack

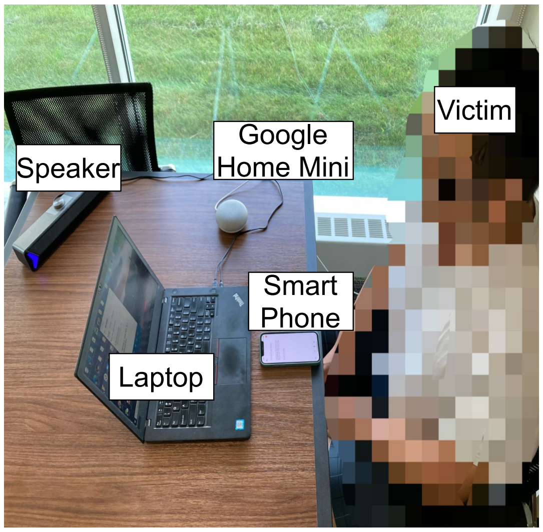


(b) Untargeted attack

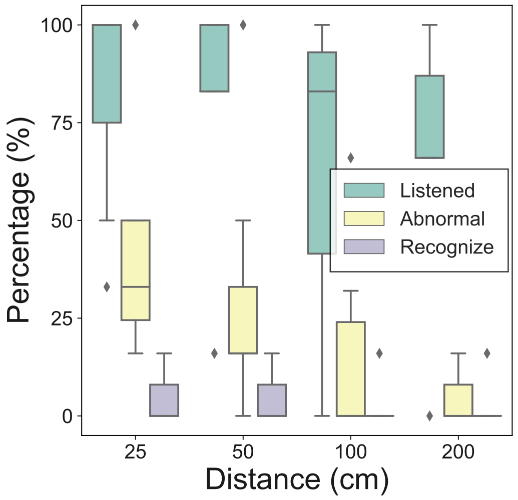
Figure 5.11: Attack cross different victims.

Two volunteers attend the in-person experiment (see Figure 5.12a) and the rest of them carry out the experiment at their homes. We recruit 13 volunteers from Amazon Mechanical Turk with complete experimental instructions.

The volunteers are asked to *pretend speaking to their voice assistants* while hearing the perturbation, after which they will answer questions to depict their comprehension of the heard perturbations. The options for perception levels include: *Listened*, *Abnormal*, and *Recognize*. *Listened* indicates that the volunteer can hear a perturbation but regard it as a normal noise; *Abnormal* implies that they hear some strange sounds; and *Recognize* stipulates that they can understand the meaning of the heard sound. We report the experiment result in Figure 5.12b. It shows that most of the participants can hear the perturbation within a short distance, but less than 50% of them regard the perturbation as an abnormal sound. Such "abnormality" feeling will gradually disappear with the increasing attack range, which ends with 10% in 2 meters. Moreover, even though all the perturbations are "meaningless" phonemes, some participants claim to understand their meanings (though the understanding is incorrect). To summarize, PhantomSound can be noticed by victims, but would not vastly raise their attentions. Notably, the victims are generally unaware of the meaning of perturbations.



(a) Experiment setup



(b) Users' perception level of PhantomSound

Figure 5.12: Real-world user study of PhantomSound.

5.5: Discussion

5.5.1: Low-cost Attack

Table 5.10 lists the cost comparison between PhantomSound and the existing work [37]. The first row records the pricing information of the commercial APIs, which is measured by the duration of given audios (in minutes). The recent black-box attack [37] is reported to incur the cost of 1,500 queries for building the substitute models, and every query uses an audio with 25 seconds long. In total, such an attack requires $1500 * 25/60 = 625$ minutes to train a surrogate model, and can only generate 10 pre-selected commands. To generate extra commands, the attacker needs to submit additional queries (∼100) for the candidate AEs. Suppose the length of candidate AEs is 6 seconds, the total time cost for generating extra AEs is $6 * 100/60 = 10$ minutes. All together, the duration of queried audio is 72.5 minutes for producing one single AE. In contrast, PhantomSound does not require a substitute model, and as such, it only takes ∼300 queries and ∼2,000 queries of one-second audios to craft an untargeted AE (Ours-U) and a targeted AE (Ours-T) respectively. We then present the cost to generate one AE based on the pricing and the query audio length (shown in row 4 and 5). In the end, PhantomSound saves 93.1% and 65.5% of the cost for crafting an AE, a drastic improvement.

5.5.2: Limitations

The limitations of PhantomSound include that: 1) the attack is sensitive to ambient noise; 2) there is no guarantee to generate an AE for any input and any target; 3) this attack could not substantially modify very long sentences; 4) the attack distance is relatively short as presented in Section 5.4.7. To address the first and the fourth limitation, the adversary can either amplify the perturbation power or attack the victim in a relatively quiet place. The second and third limitations are possibly addressed using multiple repeated attempts of phoneme injections, which will increase the likelihood of generating a successful perturbation with a potential caveat of growing costs.

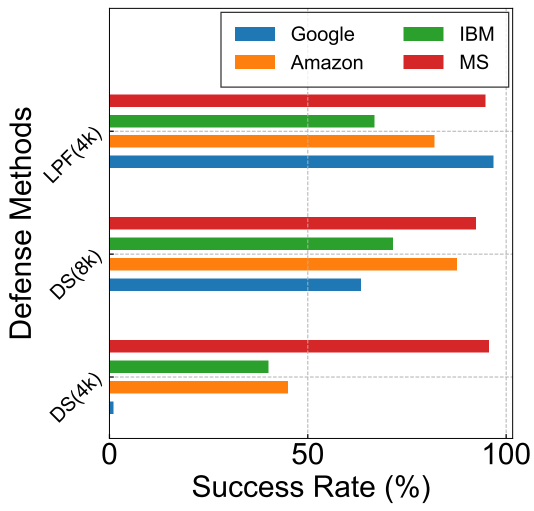
Table 5.10: Cost comparison

	Google	MS	AMZ	IBM
Pricing/min	\$0.024	\$0.016	\$0.024	\$0.01
Build model [37]	625 min			
Craft AE [37]	10 min			
Total time/AE [37]	72.5 min			
Total time/AE (**Ours**)	5 min - 25 min			
Cost/AE [37]	\$1.74	\$1.16	\$1.74	\$0.725
Cost/AE (**Ours-U**)	\$0.12	\$0.08	\$0.12	\$0.05
Cost/AE (**Ours-T**)	\$0.6	\$0.4	\$0.6	\$0.25
Saving/AE (**Ours**)	**93.1%/65.5%**			

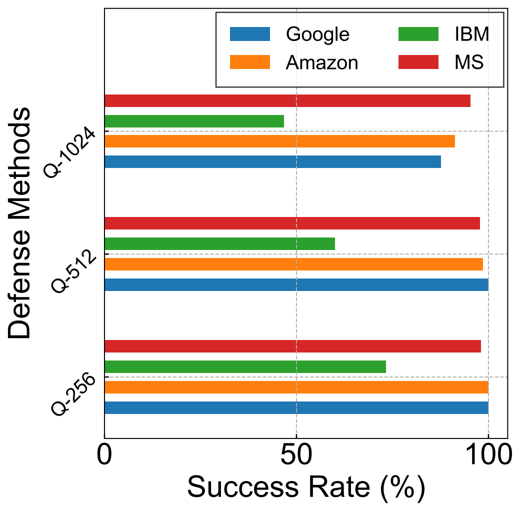
5.5.3: Defense

Prior studies [119,202,204] reveal that the audio adversarial attack can be defended by signal processing techniques, since the adversarial perturbations are delicately crafted and hence are deemed fragile. The signal processing techniques, however, can reduce the fidelity of perturbations and hence protecting the ASR models. Typical signal processing defense methods include 1) *Down sampling (DS)*: decreasing the sampling rate of AEs to disrupt the quality of AEs [119,202,204]; 2) *Quantization*: as the original AEs are encoded by 16-bit values, the quantization technique rounds the 16-bit precise value to its nearest integer multiple of Q, where Q represents the quantization level. A higher Q results in a lower precision of AEs, which has been adopted to defend against the attacks [119, 202]. 3) *Low pass filtering (LFP)*: the defense can use a Butterworth low-pass filter with different cutoff frequencies to remove the high-frequency components of the perturbations [119].

We reproduce the aforementioned three defense methods to test their effectiveness against PhantomSound. Specifically, for *DS* approach, we modify the sampling rate of AEs from 16k to 8k and 4k. In the *quantization* setting, we follow the existing work [119] to set Q as 256, 512, and 1,024. Then, we build a Butterworth low-pass filter with a cutoff frequency of 4kHz, and set the order of the filter as 6. To validate the defense performance comprehensively, we generate **1,190** AEs from 20 clean audio samples and process them with 6 different defense settings.



(a) Defense performance of DS and LPF



(b) Defense performance of quantization

Figure 5.13: Performance of PhantomSound against different defenses.

We use the processed AEs to attack 4 commercial ASR APIs. Figure 5.13a shows that *LPF* can barely impact the attack success rate of AEs and APIs. For comparison, the *DS* technique slightly changes the attack success rate from 100% to 92.4% (Microsoft), 71.4% (IBM), 87.5% (Amazon), and 63.3% (Google). This method can further reduce the success rate by applying a lower sampling rate (e.g., with 4k sampling rate, the IBM and Amazon API can defend against ∼60% attacks, while the Google API is not supported for the audio input with such a low sampling rate. Different from the findings from previous work [119, 202] that quantization is effective in defending against the adversarial attack, our results show a converse performance. From Figure 5.13b, we observe that only the IBM API can be affected by the quantization, which reduces the success rate to 73%, 61%, and 47% for q=256, 512, and 1,024, respectively. To summarize, our results demonstrate that the existing signal processing-based defense approaches cannot protect the commercial APIs from PhantomSound. Future research on defense mechanisms are needed to provide more secure speech-to-text and voice assistance services.

5.5.4: Ethical Issues

The intention behind publishing this work is to enlighten the academic and tech community about the vulnerabilities of commercial ASR APIs and smart speakers, it may also provide malicious actors with the knowledge and tools to exploit these vulnerabilities for harmful purposes, such as privacy invasion, identity theft, or unauthorized control of connected devices. If the findings of this chapter are misused, malicious actors could potentially manipulate smart speakers into sharing sensitive information or performing unauthorized actions, there may be potential financial and reputational harm to individuals and corporations. To address these ethical concerns, it would be advisable to collaborate with manufacturers of smart speakers to design effective countermeasures to defend against this attack.

5.6: Related Work

The study of adversarial attacks starts from the discovery of intriguing properties of the neural networks around 2014 [60, 160]. Researchers manually or automatically add small perturbations to the input and thereby misleading the neural network models.

Adversarial attacks against ASR systems: Existing work [12,27,44,136] has proposed different optimization algorithms to craft effective AEs towards ASR models with some knowledge of the victim's ASR model (e.g., prediction scores or logits output). However, the robustness of their attack approaches in a real-world over-the-air scenario is usually unverified. The recent physical attacks such as CommanderSong [204], Devil's Whisper [37], and AdvPulse [119] require a substantial cost (in time and money) for the attackers to succeed in attacking the black-box voice assistants.

Signal processing attacks: Rather than exploiting the vulnerabilities of neural networks in ASR systems, the signal processing attacks aim at attacking the signal pre-processing or feature extraction modules. They usually exploit the discrepancies between the human auditory system and the perceptual hearing system of microphones to fool the ASR system. These attacks analyze the input and output of the feature extraction procedure, and then they modify the input of feature extraction and preserve the shape of output to either hide their attack [4] or mislead the ASR system in producing incorrect transcriptions [5]. Even though the existing signal processing attacks demonstrate the efficiency and effectiveness against the black-box models, it is relatively straightforward to defend against using frequency filters.

Audio backdoor attacks: Different from adversarial attacks which attack a trained model, backdoor attacks [68,79,120] inject backdoor triggers during the training process. Recently, researchers demonstrated that the backdoor attack [152,207] can also be implemented in the ASR model and Speaker Verification models. To defend against the backdoor attacks in the image domain, several countermeasures are proposed [75,77].

Other related works: Some attackers exploit the imperfection of hardware (e.g., microphone) to deliver inaudible attacks through different media [115, 158, 200, 210]. Besides, Danger [215] uses homophones (i.e., different words with similar sounds) to attack ASR skills. Researchers also develop side-channel attack [184] by injecting voice commands through a power line. Speech synthesis attack produces victim's fake speech by generative models [188]. To protect the victim's original speech, researchers add perturbations([95, 183]) to prevent the generating of deep fake speech.

5.7: Summary

In this chapter, we proposed PhantomSound, a practical, black-box, and query-efficient audio attack against commercial ASR systems and IVC devices in a real-world scenario. As opposed to the existing attacks that require prior knowledge of the target model, we propose a phoneme-level searching method to generate AEs and perturbations rapidly and effectively in a black-box setting. In the real-world experiments, PhantomSound is shown to be practical and robust in attacking 5 popular commercial voice controllable devices over the air, which could potentially cause hazards to the smart home. In the next chapter, we will introduce a privacy protection design to defend against voice privacy leakage.

CHAPTER 6: SPEAKER SELECTIVE CANCELLATION VIA NEURAL ENHANCED ULTRASOUND SHADOWING[7]

6.1: Introduction

The widespread adoption of smartphones and Internet-of-Things (IoT) devices, equipped with built-in microphones, enables people to easily record audio anywhere and at any time. However, the increasing prevalence of unauthorized microphones has given rise to many instances of privacy breaches. Commercially available microphones, which are easily accessible, pose a risk of capturing users' biometric data, such as voiceprints or eavesdropping on confidential conversations. Consequently, the issue of illicit voice recording has emerged as a significant concern in society.

Recent studies [116, 169] attempt to disrupt unauthorized voice recording by emitting an ultrasonic scrambling noise wave (i.e., a jamming signal) to obfuscate the superposed voice. However, the scrambling noise wave is generated using low-level acoustic signal features that are irrelevant to the speaker's identity. Consequently, other benign microphones in the reception range will also be jammed, most of the time undesirably. In fact, the use of such voice jammers in public spaces is prohibited and unlawful (violation of *47 U.S.C. § 333*), since it poses serious risks to critical public safety communication. Moreover, if the attacker learns the frequency pattern of the scrambling noise wave, the attacker can deploy an additional microphone to nullify the noises and record them illegally. To allow users to secure their voices lawfully without intervening in others' microphones/recorders usage, we propose *NEC (Neural Enhanced Cancellation)*, which only jams *a*

[7]This chapter is based on previously published work by Hanqing Guo, Chenning Li, Lingkun Li, Zhichao Cao, Qiben Yan, and Li Xiao titled "NEC: Speaker Selective Cancellation via Neural Enhanced Ultrasound Shadowing" published at the Proceedings of the 2022 IEEE International Conference on Dependable Systems and Networks(DSN). DOI: International Conference on Dependable Systems and Networks [70].

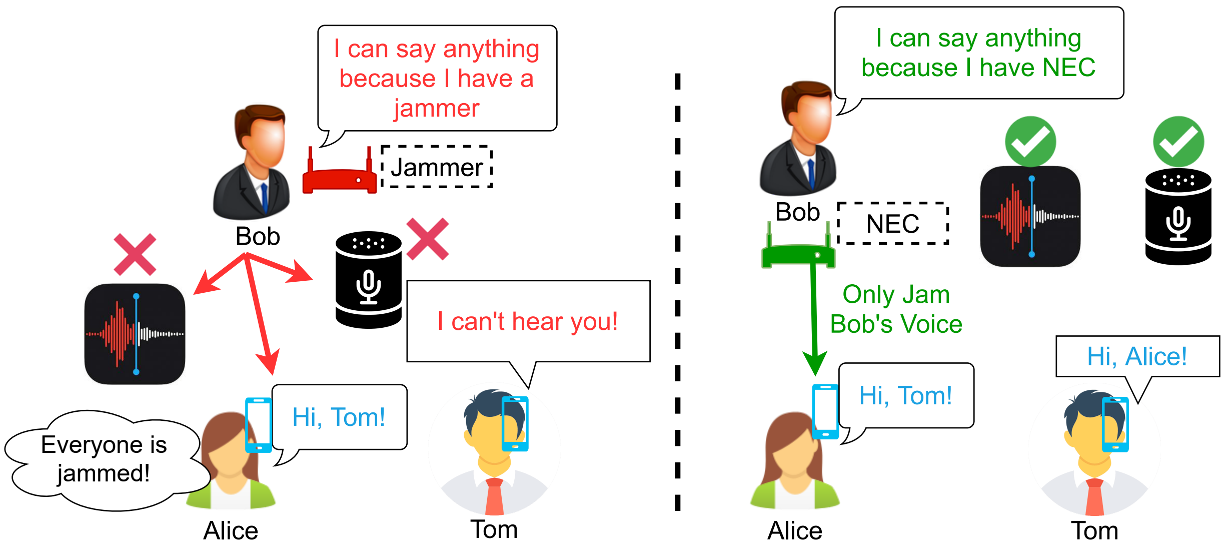


Figure 6.1: NEC cancels the target speaker's (e.g. Bob's) voice without intervening in other communications.

specific target speaker's voice from the recording of any microphones nearby.

Figure 6.1 illustrates the necessity of deploying NEC instead of the commercial audio jammer. Consider that Bob is initiating a private conversation in a public area (e.g., a cafe or work office), in order to prevent his speech from being leaked, he turns on a commercial jammer to obfuscate all the surrounding input devices. The left sub-figure shows that during the attack, other applications such as voice reminders, voice assistants, and phone calls are all effectively disabled by Bob's jammer, which is not only unlawful but also annoying to other users. In contrast, if Bob deploys NEC, only his speech is imperceptible by the others' microphones, while other users can still safely use their voice applications as usual.

Generally, NEC is composed of a microphone, a neural network model, and an ultrasonic speaker. Figure 6.2 entails the components of NEC, the red lines demonstrate the target speaker's voice (e.g., Bob's voice), while the green lines represent Bob's irrelevant voice (e.g., Alice's voice, background noise, and model processed voice). Our goal is to make Bob's voice unrecognized/unrealized on Alice's phone/recorder. At the very beginning, the microphone perceives both Bob's and Alice's voice. Then, we feed the mixed audio to our proposed deep neural network (DNN) model. Note that, compared to the existing systems that utilize low-level acoustic signals (such as Gaussian noise or scrambling noise), we use the DNN model to extract the high-level speaker-

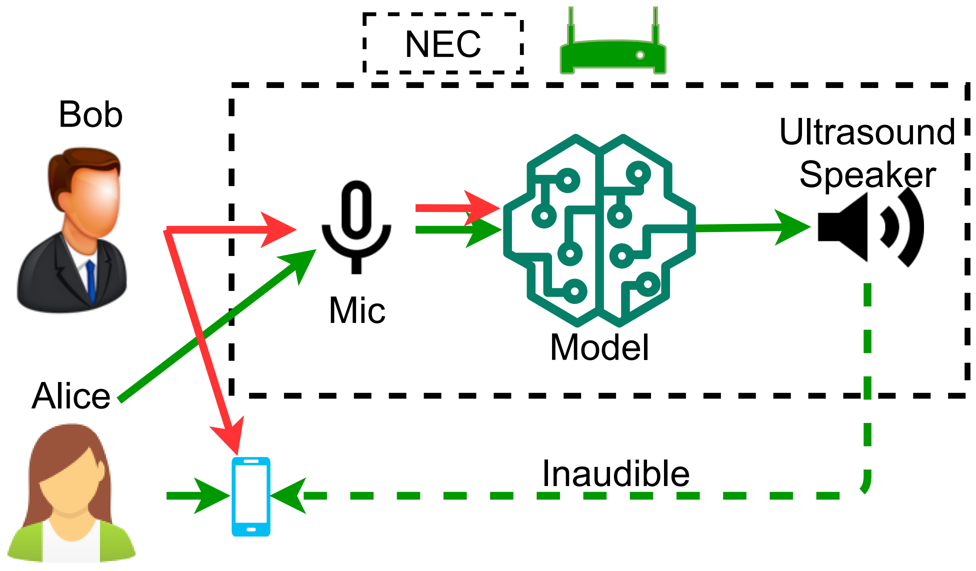


Figure 6.2: The voice stream flow of NEC

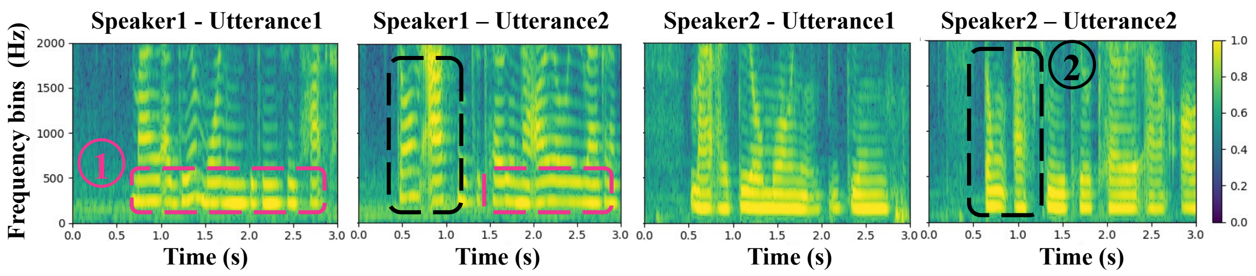


Figure 6.3: Distribution of formants across spectrograms, representing the speaker-specific but utterance-independent timber pattern. Utterance 1: "My ideal morning begins with hot coffee." Utterance 2: "Don't ask me to carry an oily rag like that."

specific vocal features for differentiating Bob's voice from the mixed recordings. The output signal of the DNN model is marked as *shadow sound*, which is then modulated to ultrasonic frequency to make it inaudible to other users. Subsequently, Alice's phone will receive a combination of Bob's voice, Alice's voice, and the inaudible shadow signal generated by NEC. The signal combo will yield a mostly undisturbed sound for Alice.

We have four main design goals as follows:

- *Utterance-independent Vocal Feature Extraction.* For a target speaker, we need to train our DNN model with the speaker's reference audio before the deployment. To alleviate the training overhead across different scenarios, the speaker's vocal features should be independent of his/her utterances. As such, we can deliver a one-fits-all DNN model, which is trained once and easily transferred.

- *Microphone-aware End-to-end DNN Training.* The shadow sound is superposed on the speaker's voice at the microphone. To make the superposition more effective, we need to design an end-to-end training pipeline that aims to maximize the effectiveness of the superposed shadow sound.
- *Low-latency Shadow Sound Generation.* We will modulate a shadow sound onto the ultrasonic frequency to make it inaudible. However, the processing delay may degrade the shadowing efficiency due to the feature mismatch between the speaker's voice and the generated shadow sound. Thus, we need a DNN model that is computationally efficient.
- *Synchronization-free.* To cancel Bob's voice on other devices, it typically requires the synchronization of the arrival time of the shadow sound, Bob's sound, and Alice's sound. However, it is challenging to synchronize them (without modifying Alice's devices). Therefore, we need a synchronization-free approach for voice cancellation.

To achieve all four goals, we first explore the human vocal principle and observe the speaker-specific but utterance-independent formants of the audio spectrogram from ten speakers using various speech contents. Then, we design a DNN model to generate a shadow sound by imitating the superposition of multiple waves at the microphone. The DNN includes the speaker encoder and selector for feature reference and extraction. Moreover, we analyze the delay bound and compress the DNN layers to guarantee that the processing delay can meet the requirement on various devices (e.g., mobile, Raspberry-Pi).

We implement NEC using commercial off-the-shelf (COTS) ultrasound transceivers and evaluate their performance in different real-world scenarios. In the experiment, we run a benchmark testing using a public speech corpus dataset and two real-world case studies. The evaluation results demonstrate that NEC effectively mutes the target speaker at a microphone by causing a 200% word error rate under Google's voice-to-text service without interfering with others' conversations. Our contributions are summarized as follows:

- NEC is the first practical speaker selective cancellation system, which aims to protect the target speaker's voice without interfering with other microphones in presence.
- We explore the human vocal principles and design a DNN model to imitate the superposition of waves at the microphone, which produces the speaker-specific but utterance-independent shadow audio in real-time.
- We implement NEC and extensively evaluate its performance with the benchmark and user studies. The results show its superior performance in comparison with state-of-the-arts systems. The demos can be found on our project website: https://nec-app.github.io/.

6.2: Background of Vocal System

Observations: To illustratively show the harmonic components of a sound induced by the physical vocal system, we first collect four audios from two volunteers. Each volunteer records two audios of their utterances of two sentences: "*my ideal morning begins with hot coffee*" and "*don't ask me to carry an oily rag like that*". For each audio, we derive the corresponding formants [166] via FFT for each frame with a duration of 20 ms. The rationale is that the duration of a typical phoneme is longer than 20 ms, representing the maximal frame length [199]. Thus, each frame is dominated by the harmonic components of sustained tones, i.e., the number and relative intensity of the upper harmonics in the sound.

The results are presented in Figure 6.3. We can observe the consistent formants of each speaker with various spoken contents. For example, the similarity of the resonant frequency and the relative intensity of formants of different utterances from the same speaker can be observed in area ①, shown in red boxes. Hence, these characteristics are utterance-independent. Conversely, area ② in black boxes implies the distinct distribution of speaker-specific formants, which can also be observed across multiple spectra of various frames.

Validation: Based on the observations above, the remaining challenge is to quantify the utterance-independent but speaker-specific feature in audio spectrograms (i.e., area ① and ②), namely timbre pattern [190]. To guarantee the phonetically balanced state in the timbre, we first average the dynamic influence of individual phonemes by computing the averaged spectrum for all frames, namely **Long-time Average Spectrum (LAS)** [121, 199]. LAS can average out the dynamic characteristics associated with various phonemes such as the motion of the articulators [105]. Suppose the spoken content for each person is divided into M frames with the duration T in time, the LAS $F(w)_{LAS}$ can be formulated by averaging the spectrum of each frame:

$$F(w)_{LAS} = \frac{1}{M} \sum_{m=1}^{M} \mathcal{F}(f_m(t)), \tag{6.1}$$

where $\mathcal{F}$ denotes the FFT, and $f_m(t)$ is the frame waveform signal with the duration T.

To visualize the distinctive LAS features for different speakers, we compute the LAS of four speakers (e.g., A, B, C, D), with every speaker requested to read the same sentence (e.g., "don't ask me to carry an oily rag like that"). The results in Figure 6.4 show that every speaker's LAS feature is unique even when their speech contents are the same. The distinctiveness of LAS features demonstrates the potential of differentiating voices from multiple speakers. To further verify

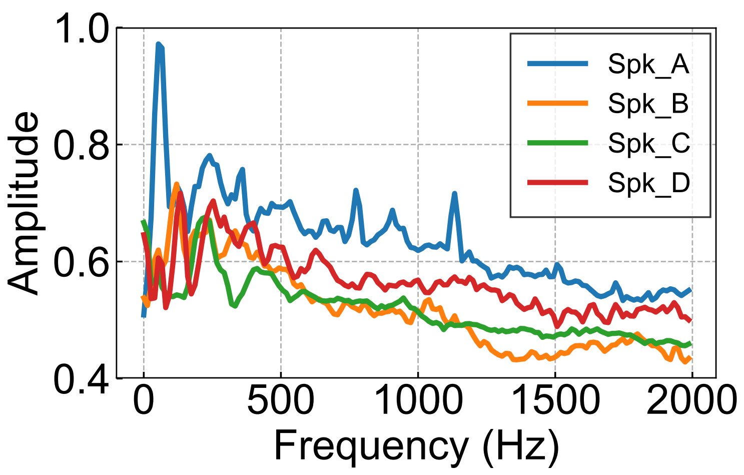


Figure 6.4: LAS results from four speakers.

the utterance-independent but speaker-specific timbre pattern in our computed LAS, we compute the Pearson correlation and deliver the correlation matrix across different speakers and spoken contents. Specifically, we first collect ten different utterances from four speakers (e.g., A, B, C, D) and compute the Pearson correlation across $F(w)_{LAS}$ [199]. As shown in Figure 6.5, the correlation coefficients for the same speaker with different utterances can reach up to 0.96 on average, whereas

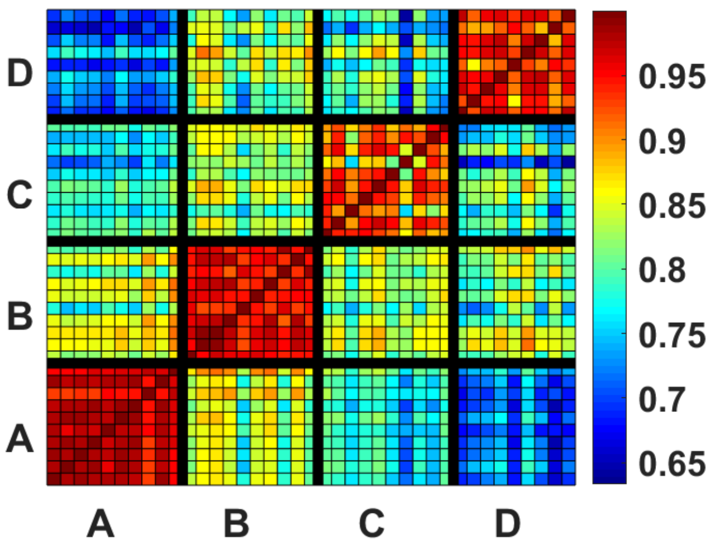


Figure 6.5: Pearson correlation matrix of the long-time average spectrum of 10 different utterances from 4 speakers.

they are generally below 0.75 across speakers, even with the same utterances. The former implies the consistency of spectrum across various spoken contents for the same speaker, while the latter indicates the distinct timbre patterns of different speakers, which demonstrates the feasibility of using LAS to quantify the timbre patterns from audio spectrograms of different speakers.

6.3: NEC System Design

As shown before, the voice signals from different human speakers present different spectrum features. Meanwhile, for the same speaker, the spectrum features are consistent across different spoken contents. The remaining challenge is to generate a speaker-specific shadow sound from these spectrum features.

6.3.1: System Overview

System pipeline: The goal of NEC is to cancel Bob's voice in the wild (e.g., no one can record Bob's voice in their microphone, and no one is affected by Bob's NEC devices). However, passively canceling Bob's voice on Alice's recorder is very challenging. Prior work [151] takes a great effort to estimate the arrival of Bob's voice through a wireless channel, and compute the inverse signal of Bob's voice before the acoustic signals of Bob arrive. Next, they synchronize Bob's voice with

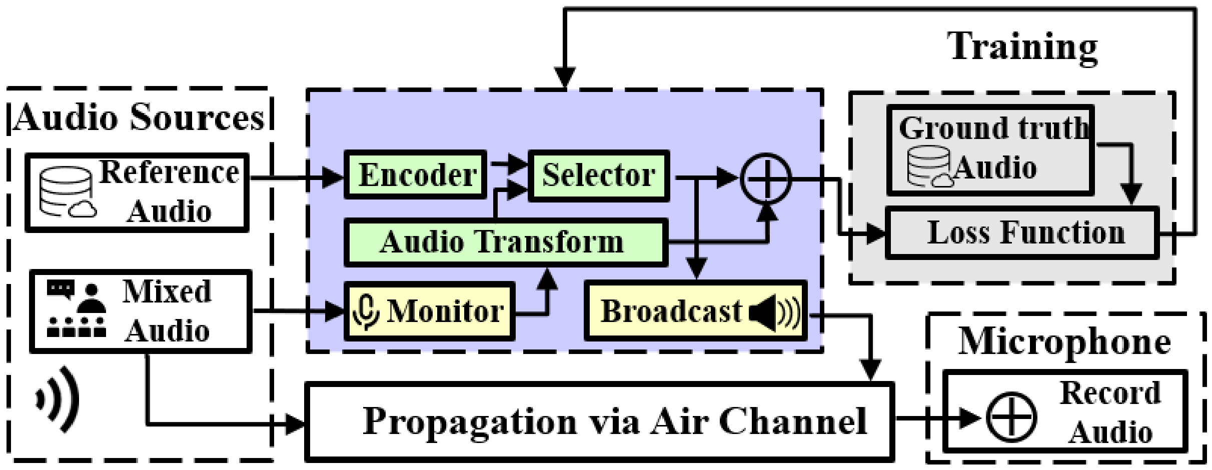


Figure 6.6: Overview of NEC, which includes the software (green) and hardware (yellow) design as well as the training stage (grey) of our system.

the crafted inverse signal to perform the voice cancellation using rigorous procedures. However, such design relies on the speed difference between wireless signal and acoustic signal. In short-range scenario (e.g., Bob is close to Alice), their work will no longer be effective since the arrival time could be very close. Instead of generating the inverse signal by the prior knowledge of Bob's speech, we propose a *superposition* method to reduce the strength of Bob's sound signals received by Alice's microphone. In other words, NEC produces a shadow signal to be mixed with Bob's voice, which will distill Alice's sound on her microphone.

Figure 6.6 shows an overview of NEC's architecture from audio sources (left block) to the (Alice's) recording microphone (bottom right block), which serve as inputs and outputs, respectively. The reference audio is the historical recordings of the user, which is prepared to assist the DNN model to separate the voice stream of the user. The mixed audio refers to the audios containing Bob's voice and others' (background) voices. The output of NEC model is a shadow signal transmitted by an ultrasound speaker. The mixed audio and the shadow signal combined together to form the recordings on Alice's microphone. We assume that Alice receives the same mixed audio as the one collected by the NEC's microphone in proximity.

To create a general neural-enhanced framework, we first train NEC at the spectrogram level (top right block) in the offline training stage, where a $\oplus$ operation in the purple block represents the audio spectrogram superposition that combines the outputs of the **Selector** and **Audio Transform** modules. The functionality of **Selector** is to generate spectrogram that exclude Bob's sound, and the **Audio Transform** serves to transform the waveforms into spectrograms. Then, we convert the shadow spectrogram induced by our selector into inaudible ultrasound wave via **Broadcast**. The shadow wave will propagate through the air channel along with the mixed wave (§6.3.3). $\bigoplus$ inside the Microphone block indicates the wave superposition of the mixed audio and broadcasting shadow sound at the microphone. Due to the equivalence of audio superposition for wave and spectrogram, the effectiveness of wave superposition is guaranteed for testing scenarios, as mixed audio and shadow sound arrive simultaneously at the microphone. The superposed wave corresponds to the recorded audio which effectively hides the target's (e.g. Bob's) voice.

Training stage: The purpose of model training is to generate spectrogram that not caused by Bob's voice for any speech context with Bob. To achieve that, we manually craft mixed audios which contain Bob's voice and other speakers' voice, and use our selector to generate Bob's irrelevant spectrogram. To train NEC, we first provide a pre-trained **Encoder**, which generates the speaker-specific d-vector [181, 182] from the reference audio (e.g., 3 audio instances lasting 3 seconds) as reference input for the selector. Meanwhile, the mixed audio is processed by the audio transform, which generates a mixed spectrogram as another input of the selector. The rationale for using spectrogram has two folds. First, the **LAS** feature is effective in distinguishing different speakers based on our previous observation (§6.2); second, the calculation of LAS refers to the procedure of calculating the average spectrum for audio clips, which can be unfolded across multiple clips as a spectrogram. We directly feed the mixed spectrogram into our selector, along with the d-vector extracted from the reference input. This can boost the accuracy of DNN in extracting the high-level speaker-specific but utterance-independent vocal features from the mixed sound (§6.3.2).

Overshadow stage: A key property of NEC is its generalization for deployment in the wild. First, rather than the cumbersome model-retraining and data collection, only 3 audio instances lasting 3 seconds are required by our one-fits-all model for new user enrollment. Second, due to the linearity of the Fourier Transform (§6.3.2), we can transfer the spectrogram superposition into the wave superposition of audios at the microphone to guarantee the overshadowing performance. Finally, to avoid the disturbance during the overshadowing of NEC, we further convert the shadow spectrogram into inaudible ultrasound (§6.3.3).

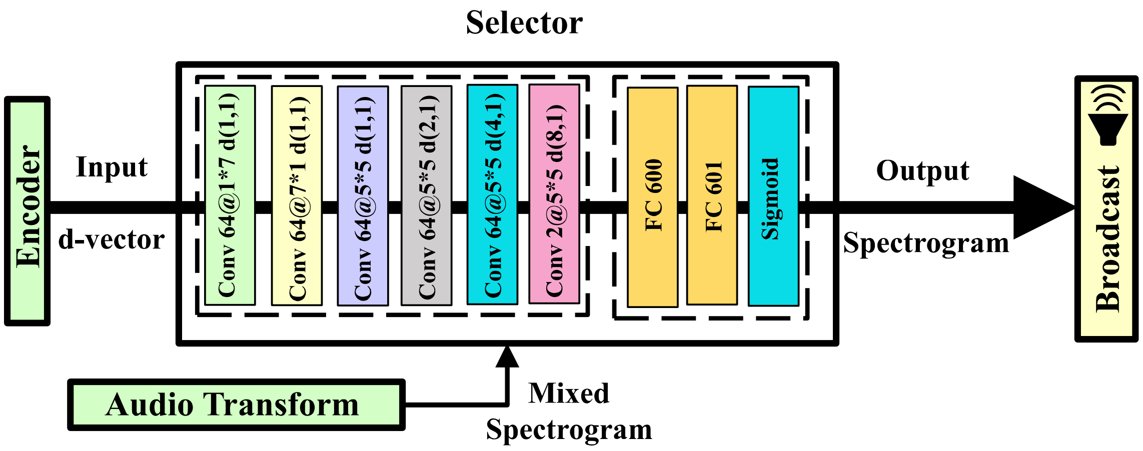


Figure 6.7: NEC's DNN Selector generates the utterance-independent but speaker-specific shadow spectrogram by imitating the superposition of waves at the microphone.

6.3.2: Neural Enhanced Selective Speaker Cancellation

In this section, we present the design of NEC's DNNs, which aim to utilize the utterance-independent but speaker-specific features to generate the shadow sound. NEC incorporates an efficient **selector** to produce a shadow spectrogram, and further add Bob's voice through overshadowing onto the mixed spectrogram.

Architecture of DNN

Encoder: The encoder module follows the design of **d-vector** in prior studies [176, 181, 182]. This module takes the reference audio of a target speaker as input and produces a speaker-specified embedding to allow the **selector** to filter out the target speaker's voice from the mixed audio spectrogram.

Selector: The purpose of the selector is to produce a shadow spectrogram and further hide Bob's voice by superposing the shadow spectrogram onto a mixed spectrogram. As shown in Figure 6.7, the selector takes the d-vector and the mixed spectrogram as input. We formulate the mixture of the spectrogram as follows:

$$S_{mixed} = |\sum_{n=-\infty}^{\infty} x_{mixed}[n]W[n-m]e^{-j\omega n}|, \tag{6.2}$$

where the n-sample mixed audio in $\mathbb{C}^n$ is converted into a spectrogram with t *sampling points* and f frequency bins in $\mathbb{R}^{t\times f}$. $W[n-m]$ is the Hann window, and m is the window size. More specifically, the mixed spectrogram is composed of Bob's voice S_{Bob} and background voice S_{bk} (e.g., Alice's voice) as follows:

$$S_{mixed} = S_{Bob} + S_{bk}. \tag{6.3}$$

In practice, the input audio lasts 3 seconds with a sampling rate of 16 kHz. The number of samples is 48,000. Also, we set the FFT size as 1,200, resulting in 601 frequency bins. The window length and hop length are 400 and 160, respectively, which generates 299 frames. Then, the shape of S_{mixed} is 601×299, denoted as (F, T), the frequency resolution and frame resolution are 13.31 Hz and 25ms with 15ms overlap. We transpose the mixed spectrogram for further processing and denote the shape of the transposed spectrogram as (T, F).

With the mixed spectrogram and d-vector in hand, we then utilize them to design a neural network based on our observation in §6.2. Revisiting Figure 6.3, the frequency distribution of formants [166] and harmonic determine the identity of a given speech (i.e., LAS sufficiently captures the speaker characteristics). Our design goal of the selector is to capture these characteristics with multiple layers of CNNs. Prior to building the neural network structure, we propose the requirement for our DNN model as: **1)** the selector should be able to capture the formants and harmonic feature; **2)** the selector should consider the consistency of the frequency distribution within the same voice source.

In our DNN design, we only focus on the first three formants since we observed that the lower orders of harmonic have more energy and are more representative for a single speaker. As the bandwidth of the first three formants ranges from 33 Hz to 79 Hz [56], we design the first convolutional layer with 64 filters, whose size is 1×7. The rationale of using this flat filter is to convolve the frequency domain information (F). In particular, each filter covers 93.17 Hz, which is enough to cover the individual formant bandwidth as mentioned previously. Another 64 filters follow, whose size is 7×1, which can cover 115ms (determined by the frame resolution) time-domain feature (T). It is worth mentioning that the length of phoneme varies from 5 ms to 670 ms based on existing vocal research [98], and the average reading speed for an adult is 184 ± 29 words per minute [168], i.e., $281\sim387$ms per word. So the second convolutional layer only serves to explore the detailed information of the phoneme level.

To further incorporate both F domain and T domain features, we apply a sequence of (5×5) convolutional layers with the dilation ranging from (1,1) to (8,1). The dilation setting on T domain extends the effective range of filters from (5×5) to (5×40), corresponding to 85ms to 610ms. This range covers a few words and meets our **R2** for considering the consistency of frequency distribution. While other studies [181, 182, 208] also add extra layers (e.g., LSTM, CNN with larger filter size and dilation shape) for speaker separation task, we consider that those layers play a less important role. For example, a larger filter will introduce irrelevant frequency information and long time span data, when the speaker merely adjusts his/her formants frequency when speaking a single word or a short sentence.

The output of CNNs has the shape of (T, $2\times$F) since we add a padding layer before the convolutional layer to maintain the shape of feature domain consistency, where $2\times$F comes from two filters in the last CNN layer. After that, the d-vector is repeatedly concatenated to the output of the last convolutional layer in every time frame. The fused feature embedding will be fed into two fully connected layers. As a result, we get a (T, F) shadow spectrogram. Figure 6.7 shows the detailed flowchart of our selector. In total, we only use 6 CNN layers and 2 Fully Connected (FL) layers for the selector model. Compared with the existing models such as [181, 182, 208], our model is computationally efficient by eliminating the redundant modules (e.g., LSTM, CNN with larger filter size and dilation shape) unrelated to our research goal.

Spectrogram-based Overshadowing

In the overshadowing process, we first feed mixed spectrogram and d-vectors into our selector. Then, we deliver the generated shadow spectrogram to be superposed with the received mixed audio at the microphone.

Shadow spectrogram generation: From the point of view of the microphone, the received mixed audio and shadow sound should be superposed to imitate the over-the-air overshadowing at the microphone, formulated as $\mathbf{x}_{record} = \mathbf{x}_{mixed} + \mathbf{x}_{shadow}$. Those vectors represent time-series samples of mixed audio, shadow sound, and recorded audio, respectively.

Through crafting the shadow sound, our goal is to make the recorded audio as close as the background audio (e.g., Alice's sound or environmental noise). A straightforward idea is to optimize the shadow sound directly with the audio-level superposition in the time domain. However, there are two drawbacks to the temporal wave superposition. First, the temporal waveform is less representative than a spectrogram. Second, since the output of our selector is the shadow spectrogram, an Inverse STFT module should be introduced to convert spectrogram to waveform ahead of the loss function, which results in the gradient vanishing issue for back-propagation based on our evaluations. Therefore, we use a shadow spectrogram from our DNN selector for the following overshadowing processing.

Superposition for audio wave and spectrogram: The linearity of the Fourier Transform guarantees the equivalence of the temporal wave and spectrogram superposition, which can be denoted as follows:

$$\mathcal{F}[\sum_{i=1}^{n} a_i x_i(t)] = \sum_{i=1}^{n} a_i X_i(w), \textit{for } \mathcal{F}[x_i(t)] = X_i(w), \tag{6.4}$$

where $\mathcal{F}$ denotes the Fourier Transform and $x(t)$ is the temporal waveform signal. Given the linearity of Fourier transform with a coefficient a_i, we can convert temporal wave superposition into a linear combination of spectrograms as follows:

$$S_{record} = S_{mixed} + S_{shadow}. \tag{6.5}$$

The S_{record}, S_{mixed} and S_{shadow} correspond to the spectrogram of recorded audio, mixed audio and shadow audio, respectively. To avoid the gradient propagation issues and expedite the convergence of DNN, the shadow spectrogram from the speaker selector is first normalized before being superposed with the mixed spectrogram. To allow the recorded magnitude to eliminate Bob's voice while retaining other's (e.g., Alice's) voice components, we design the loss function:

$$Selector^*_{opt} = \underset{Selector*}{\text{argmin}} \, ||S_{record} - S_{bk}||_2^2, \tag{6.6}$$

where the $Selector*$ denotes the model parameters of our DNN selector, and the S_{record} is the sum of mixed spectrogram and shadow spectrogram. Using the back-propagation with the L_2 norm loss, we can derive an optimal parameter $Selector^*_{opt}$ for our DNN selector, which will output an optimal shadow spectrogram S_{shadow}. This optimization ensures the resulting S_{record} to be as close to S_{bk} as possible.

6.3.3: Overshadowing Over the Air

Inaudible Shadow Sound Generation

Given the shadow spectrogram generated by the trained DNN selector, we can apply the inverse STFT on the shadow spectrogram to derive the shadow sound wave for further broadcasting. To make the shadow sound inaudible for privacy concerns and deployment convenience, we resort to the non-linear property of microphones [140, 201] to modulate the emitted shadow wave, via the **Broadcast** module in Figure 6.6.

Non-linearity of hardware: The non-linearity property of microphone hardware represents the physical limitations of the diaphragm and the pre-amplifier, which amplify the signals in a non-linear manner. Mathematically, given an input signal V_{in} to the microphone, the output signal V_{out} of the commercial amplifier within the microphone is not amplified linearly, i.e., $V_{out} \neq A_1 V_{in}$, where A_1 is the gain for input. Instead, the output signal is $V_{out} = A_1 V_{in} + A_2 V_{in}^2 + A_3 V_{in}^3 + \cdots$. We focus on $A_2V_{in}^2$ of the non-linear V_{out} by ignoring (relatively small) higher-order components [140, 211]. Without loss of generality, let $m(t)$ be a simple tone, e.g., $m(t) = cos(2\pi f_m t)$. We then up-convert the baseband signal m_t onto a carrier with central frequency $f_c > 20kHz$. The modulated signal can be written as follows with the power coefficient α:

$$V_{in} = (cos(2\pi f_m t) + \alpha)cos 2\pi f_c t. \tag{6.7}$$

Since f_c is in the inaudible frequency range, the modulated signal V_{in} cannot be heard by humans. Given the non-linearity effect, the recorded signal V_{out} will not only contain the linear component A_1V_{in}, but also the non-linear component $A_2V_{in}^2$ representing the inaudible but recorded component, denoted as follows:

$$\begin{aligned} V_{in}^2 &= (cos^2(2\pi f_m t) + \alpha^2 + 2\alpha cos(2\pi f_m t))cos^2(2\pi f_c t) \\ &= \sum_i \lambda_i cos(2\pi f_i t) + \mu, \end{aligned} \tag{6.8}$$

where f_i denotes frequency components at f_m, $2f_c$, $2f_m$, $2(f_m \pm f_c)$, $f_m \pm 2f_c$ and μ is a consequent constant. Given the low-pass filter in the COTS microphone, we can eliminate the high frequency components while retaining the f_m components, where f_m is the baseband frequency of $m(t)$ perceived by a microphone.

Shadow sound broadcast: Then, we can encode our shadow wave $\mathbf{x}_{shadow}$ into an inaudible frequency range by modulating it with a carrier whose central frequency is f_c. The broadcast shadow wave can be computed as follows:

$$\mathbf{b}_{shadow} = \mathbf{x}_{shadow} \times cos 2\pi f_c t, \tag{6.9}$$

where $\mathbf{b}_{shadow}$ refers to the inaudible shadow wave, and $\mathbf{x}_{shadow}$ is induced by S_{shadow} from inverse Fourier transform process. More discussion about the over-the-air lantency of shadow sound can be found in [70].

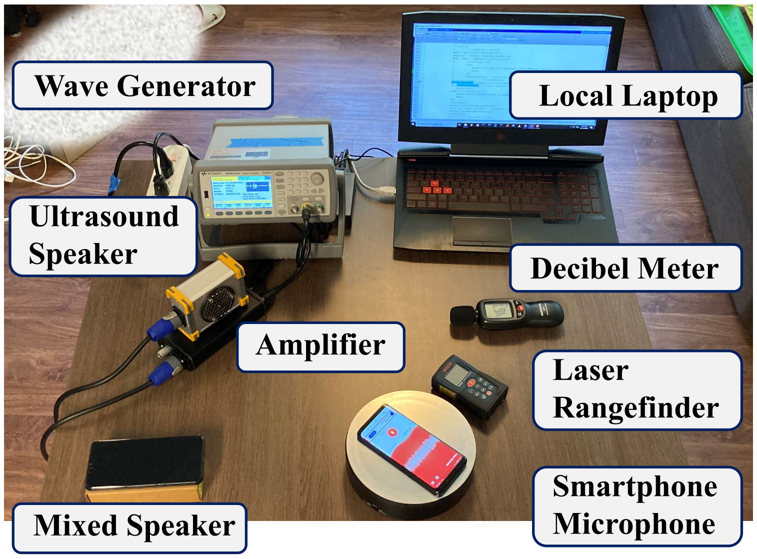


Figure 6.8: Implementation and experimental settings.

6.4: Implementation

6.4.1: Experimental Setup

Figure 6.8 presents the implementation and experimental settings of NEC. In NEC, the input mixed audio is first collected and processed by our trained encoder and selector DNNs, which produces the corresponding shadow spectrogram in Figure 6.7. Then, we transform it into audios and up-convert it into the ultrasound carrier frequency, making it inaudible during broadcasting (§6.3.1). We run NEC on a local laptop (§6.5.3) to generate the shadow spectrogram, which is sent to a Keysight 33500B waveform generator, followed by an ultrasonic power amplifier [21] to amplify the inaudible shadow wave. Being transmitted through the air by a wide-band dynamic ultrasonic speaker, Vifa [19], the shadow wave is superposed with the mixed audio at a COTS smartphone's microphone. We use a loudspeaker to play mixed audios, i.e., the ”Mixed Speaker” in Figure 6.8 emulates a mixed conversation from Alice, Bob, and others. The target's voice will be effectively muted in the final recorded audio.

Table 6.1: Testing dataset for benchmark and user cases

Scenario	Source	Freq.	Type	Instance
Joint[a]	LibriSpeech	0-8k	40/18	560/-
Conv.	/ Volunteers	0-8k	40/-	560/40
Babble[b]		0-4k	-	690 / 40
Factory[c]	NOISEX-92	0-2k	-	690/40
Vehicle[d]		0-500	-	690/40

[a]Two speakers talk jointly. [b]100 people whispering.
[c]a production hall. [d]a vehicle running at 120 km/h.

6.4.2: Dataset Compilation

Table 6.1 summarizes our testing dataset. First, we conduct the System Benchmark by testing the target speaker with the public datasets in controlled environments to verify whether our target speaker's voice can be hidden in the presence of real-world noises. Then, we deploy our system in the wild for a real attack scenario: the target volunteer wants to avoid being recorded while talking in public scenarios, but the COTS microphone can record others' voices normally.

Model training: Prior to the evaluation of NEC, we train a one-fits-all DNN model for all the defensive scenarios in public. The training dataset is constructed by mixing audios of two different speakers from LibriSpeech [134], and mixing target speaker audios with different noises from NOISEX-92 [172]. We provide the background audio that excludes the target speaker and train our model to hide the target speaker's voice, given the mixed audio and reference audios of the target speaker.

System benchmark: Using the public dataset LibriSpeech [134] as the corpus source, we first select **10 target speakers**, we collect 3 audios for each target speaker as their reference audio, and the rest audios of the speaker are treated as normal speech in a real scenario (e.g., Bob's speech). To measure the robustness of NEC, we simulate different environments with different types of noises. In order to cover different frequencies of noises (e.g., high-frequency speech and low-frequency ambient noises), the noises from 5 application scenarios are then mixed with the target speakers' voices, which results in 3,190 mixed audios in total. Then, we randomly mix the 10 target speakers' voices with the ones from the other 40 speakers, which generates 560 total instances for the joint conversation.

User case studies-1: We further collect the user study dataset from **10 target volunteers**, covering 3 females and 7 males. All volunteers are required to speak 25 sentences, respectively. Analogous to the dataset for the benchmark, we select the reference and test audios randomly, then mix test audios with 4 sources of noise. In total, 160 mixed audios are produced. Then, we randomly mix the audios of 10 target volunteers with the ones from another 18 volunteers to derive the joint conversation dataset.

User case studies-2: We conduct another user case study to justify the feasibility of NEC in the real world. As shown in Figure 6.9, Bob carries the NEC device to hide his sound in the wild. We ask Bob and Alice to speak normally, with volume at $77dB_{SPL}$ from our decibel meter placed at $5cm$ away from their lips. Then, we record the loudness, SONR, and the proportion of Bob's sound on Alice's recorder (a Moto Z4 phone) at different distances for different cases (with or without NEC).

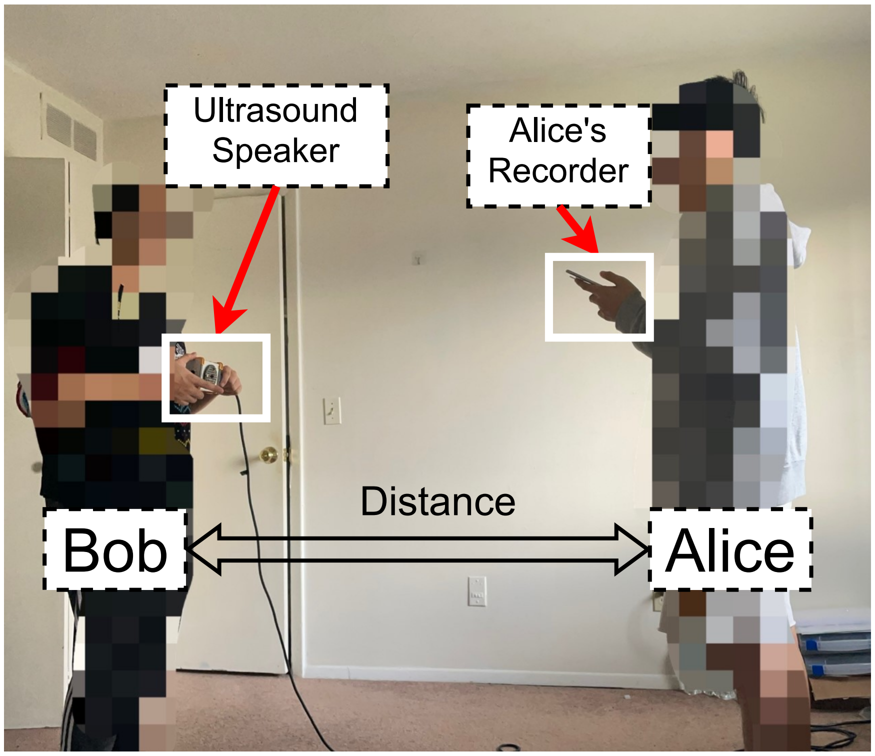


Figure 6.9: Hiding Bob's voice from Alice's recording in a real-world scenario.

Note that our testing dataset is disjoint from the training one and reference audios. Thus, the two trained models can be deployed directly with only three arbitrary reference audios from the new target speaker volunteers, avoiding the cumbersome deployment costs (e.g., model re-training and data re-collection) [212, 219].

6.4.3: Quantitative Metrics

To measure the quality of NEC, we consider four main metrics:

Source to Distortion Ratio (SDR) [175, 182] measures the ratio of energy (in dB) between the energy of the target signal and the errors (induced by the interfering speakers and artifacts) in the mixed signal. It should be low for Bob's voice and high for Alice's voice.

Word Error Rate (WER) is adopted broadly to evaluate the machine translation systems [203]. We compute the WER by employing Google's speech-to-text service to transform the acoustic signals into texts. NEC aims to enlarge the WER for the target speaker and minimize it for other speakers (e.g., Alice).

User Rating Score (URS) is the rating for recordings, in which **10 reviewers** rank the raw mixed and recorded audios of NEC with score 1-5, along with Bob's clean voice as the ground truth. Specifically, score 5 denotes the best performance, in which reviewers cannot recognize any words of the target speaker (e.g., Bob).

Sound Noise Ratio (SONR) is used to evaluate the proportion of Bob's sound in the recorded sound. We regard the mixed audio as useful sound and treat Bob's voice as noise. By computing the power ratio between the mixed audio and Bob's sound at different distances, we validate the efficacy.

6.5: Evaluation

In this section, we comprehensively evaluate NEC in different environments with different settings and devices.

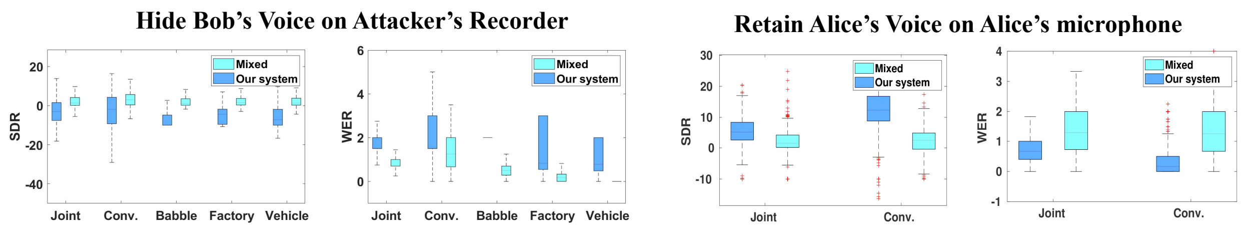


Figure 6.10: Overall system performance of our system on three setups across multiple sources of noises.

6.5.1: Overall Performance

System benchmark: We first evaluate NEC on the public dataset and provide SDR and WER across multiple scenarios in Figure 6.10. When the target speaker, i.e., Bob, expects his voice to be hidden in the recordings, the recorded audios achieve a lower SDR and higher WER compared with the mixed audios. This shows that our shadow audios can hide Bob's voice reliably, making it unrecognizable by the Google service. Specifically, the median WER increases from 0.894 to 1.798, while the SDR reaches -4.918 dB from 0.997 dB. Note that the WER of the mixed audio is too high to be recognized by the Google service due to the background speeches from other people. Yet, it can still be recognized by humans. Conversely, NEC achieves a higher WER by hiding Bob's voice using the shadow wave, making it even unrecognizable for humans. We further verify its efficacy in the user studies below.

Also, we evaluate the effectiveness of NEC to retain others' voice (e.g., Alice) in Figure 6.10(right). We set the ground truth as Alice's clear voice, and calculate the SDR and WER for the recorded audio and the ground truth audio. The result shows that, compared to the mixed audio which contains Bob's voice, we can achieve higher SDR and lower WER for capturing Alice's sound when Bob deploys NEC.

User case study-1: Figure 6.11 shows the performance of SDR and URS for hiding target volunteers' voices in the wild. We observe a consistent declination in SDR of the recorded audios compared with raw mixed ones. We can hardly recognize the target volunteer's voice in the recorded audios, as the median SDR reaches -4.374 dB, much lower than the SDR of mixed audios at 2.798 dB. To evaluate the recorded audios comprehensively, we ask 10 reviewers to score the recorded

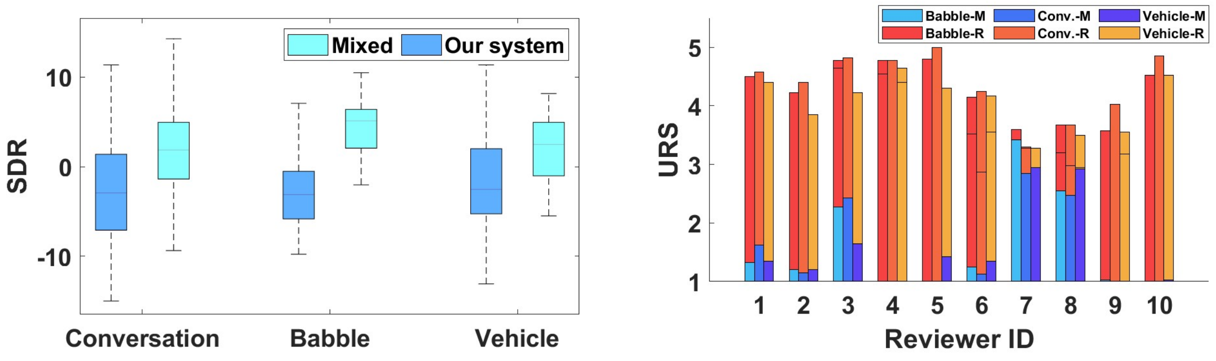


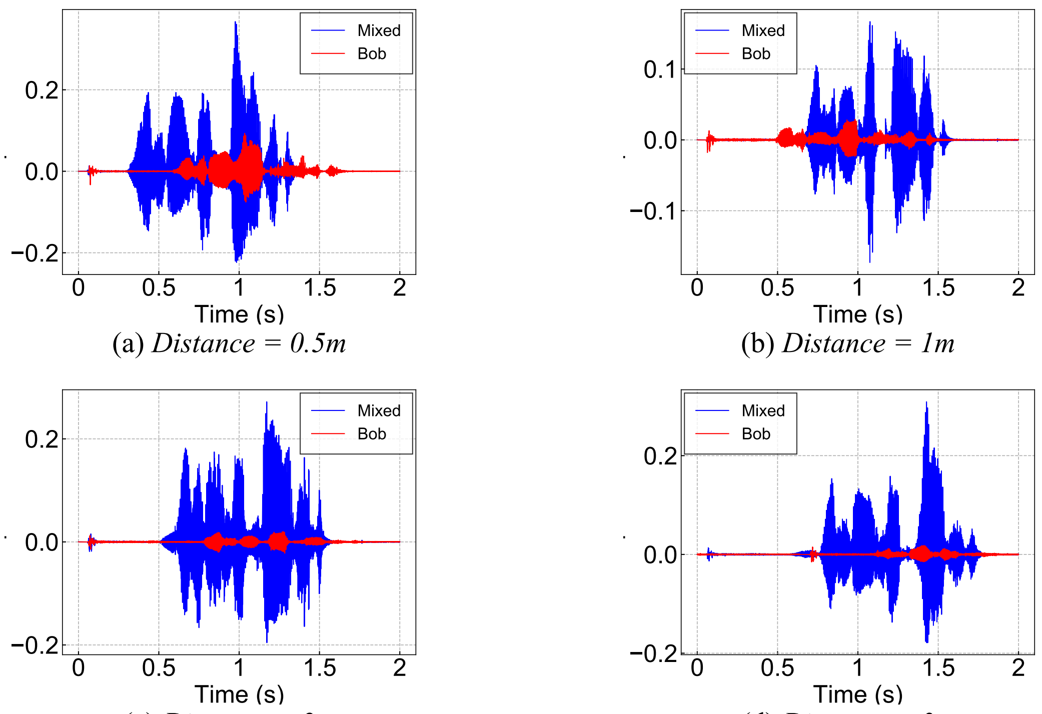
Figure 6.11: User study results.

audios and the mixed ones with ranking scores from 1-5. We expect a higher score when fewer utterances of the target volunteer can be recognized. It shows that the average score of the recorded audios can reach 4.034 for different reviewers. All 10 reviewers give 4 for most of the recorded audios, while most scores of 1 are given to the mixed audios, except the reviewers 7 and 8.

User case study-2: As depicted in Figure 6.9, in this user study, we evaluate how much of Bob's voice will be leaked to Alice's recorder with/without deploying NEC. We ask Bob and Alice to speak simultaneously, and also record Bob's sole speech with the same speech content. The mixed audio and Bob's individual speech audio are recorded by Alice's phone (Moto Z4), with varying distances between Alice and Bob (from 0.5 to 3 meters).

Figure 6.12 visualizes the waveforms of Bob's audio and the mixed audio. We can see that with the increasing distance, Bob's audio contributes less to the mixed one. We further record Bob's sound pressure level (SPL) at Alice's position and present the result in Figure 6.13a. The result shows that the SPL of Bob's audio attenuates with the increasing distance, and its loudness reaches $43dB_{SPL}$ at the $5m$ distance (between Alice and Bob) with an environmental noise level of $39.8dB_{SPL}$. In comparison, the SPL of Alice's voice recorded by her own recorder remains at $77dB_{SPL}$. Given the large gap between the SPL of Alice and Bob's voices across different distances, and the attenuation of Bob's voice with the increasing distance, we can see that Bob only needs to cancel his voice over a short range (e.g., $2m$). Next, we further justify whether NEC can effectively overshadow Bob's sound across the distance.

Figure 6.13b presents the SONR results with/without NEC. When NEC is not deployed, the



(a) *Distance = 0.5m* (b) *Distance = 1m*

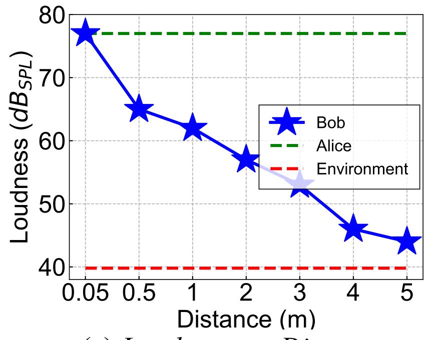
(c) *Distance = 2m* (d) *Distance = 3m*

Figure 6.12: Waveform of mixed audio and Bob's sole speech audio

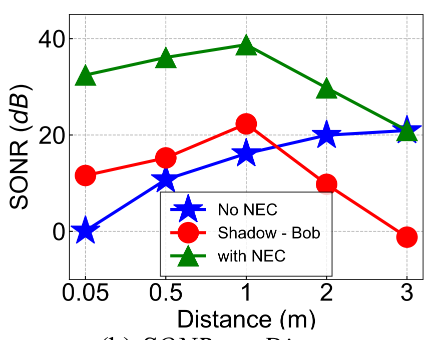
SONR between the recorded mixed audio and Bob's voice stays below 20dB, which implies that Bob's voice can be effectively captured by Alice's recorder. However, when Bob deploys NEC, even with a close distance ($< 2m$), Bob's voice can be mostly overshadowed, with SONR reaching 30dB. As mentioned above, the strength of Bob's voice signals drops significantly beyond $2m$. Therefore, although the recorded shadow audio strength also degrades dramatically beyond $2m$, the effectiveness of NEC within $2m$ makes it a viable solution for target voice cancellation.

6.5.2: Comparison Study

Next, we perform a comparison experiment between NEC and two systems. The first one uses white noise to jam unauthorized recordings, which is commonly applied to commercial ultrasonic jammers. To simplify the jamming process, we manually add 10dB white noise over the recording sound to simulate this type of jamming system. Notice that the volume of white noise is usually determined by different jammers, we use 10dB based on our previous observation of the shadow sound volume on the same phone. The second one is a scrambling-based voice hiding system called



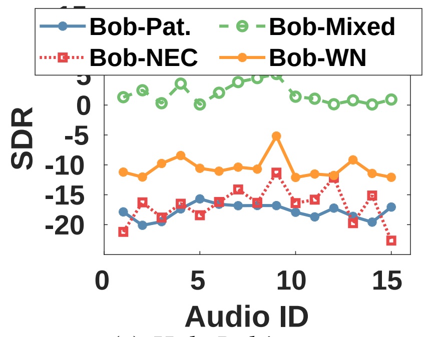
(a) *Loudness vs. Distance*



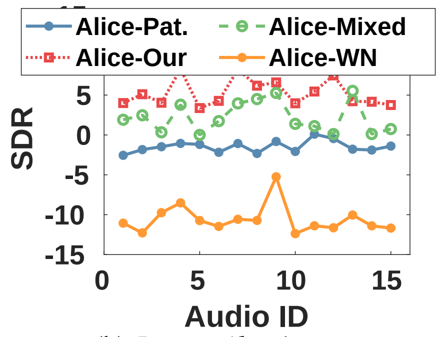
(b) *SONR vs. Distance*

Figure 6.13: Effectiveness of NEC across different distances

Patronus [116], which can hide the target recordings by scrambling with specially designed white noises and recover the target recordings at an authorized device. Given the mixed (joint) audios (e.g., two volunteers, one of which is target), we reproduce the scrambling algorithms of Patronus to hide a speaker's voice.



(a) *Hide Bob's voice.*



(b) *Retain Alice's voice.*

Figure 6.14: Comparison study.

We first compare the performance of voice hiding by computing the SDR of the target voice. Figure 6.14a shows all of the three systems: NEC (Bob-NEC), White Noise (Bob-WN), and Patronus (Bob-Pat.) achieve a low SDR by effectively hiding the target voice in the mixed audio (Bob-Mixed). We find that, compared to NEC and Patronus, the white noise solution results in higher SDR, which means it retains more target voice than the other systems. Patronus and NEC can reduce the SDR of the mixed audios from 3 dB to nearly -20 dB. Therefore, the voice hiding performance of NEC is on par with that of the specially designed scrambling-based Patronus, and better than the white noise scrambling approach. Next, we evaluate the reception quality of Alice's voice in the presence of the three systems. As shown in Figure6.14b, among the three systems, the White Noise approach cannot recover the disrupted voice, and therefore results in the lowest SDR for Alice's voice. For comparison, Patronus can recover a limited portion of scrambled sound by its recovery algorithm, and achieve low SDR for Alice's voice (i.e., -2.5 dB). The quality of Alice's voice after recovery is even lower than that of the raw mixed audios due to the influence of the scrambling noise. In comparison, NEC achieves a 5 dB gain compared with the mixed audios in recovering Alice's voice, since NEC carefully nullifies Bob's voice in the mixed audio. This experiment result demonstrates that NEC could selectively hide a target speaker's voice without interfering with other speakers. Surprisingly, NEC can even improve the reception quality of others' recording.

6.5.3: Running Time Analysis

To demonstrate the efficiency of our system, we measure the time consumption of each system module in Table 6.2. Given 100 1s mixed audios, we evaluate the latency in two different hardware platforms: 1) desktop with a single NVIDIA 1080Ti GPU; 2) Raspberry Pi 4. The total processing time of the DNN module in NEC is around 1.51 ms, and the ultrasound modulation consumes 11.96 ms on average, well below the lasting period of the 1s chunks. In comparison, it takes 2.4$\times$ more time for VoiceFilter to process the same mixed audio. On the Raspberry Pi 4, the overhead of the selector is 293.7 ms, which is faster than 446.2 ms of VoiceFilter. The achieved latency (< 300) ms on the edge deployment using Pi 4 is less than the time offset tolerance of overshadowing, which further corroborates the feasibility of NEC.

Table 6.2: Time consumption of NEC with an audio sample lasting 1s

Platform	System	Encoder	Selector	Broadcast
PC (1080Ti)	NEC	0.467ms	1.51ms	11.96ms
	VoiceFilter [182]	0.467ms	3.65ms	11.96ms
Rasp	NEC	12.7ms	293.7ms	11.96ms
	VoiceFilter [182]	12.7ms	446.2ms	11.96ms

6.5.4: Parameter Study

Diversity of hardware dependence: The variance of the non-linearity for the hardware (e.g., microphones, amplifiers, filters) on smartphones can influence the optimal selection of the modulation parameters [201], which in turn impacts the performance of our system. Here, we evaluate our system using 7 different mobile devices listed in Table 6.3. Specifically, the carrier frequency f_c is the dominant factor that affects the effectiveness of the non-linearity effect. All the tested smartphones have a range of acceptable frequency settings, and the best carrier frequency is listed in the brackets.

Table 6.3: Smartphones used for two user studies.

Model	**Brand**	**Carrier f_c (kHz)**	**Max Dis. (m)**
Moto Z4	Motorola	24-28 (28.0)	3.2
iPhone 7 P	Apple	21-29 (27.8)	0.49
iPhone SE2	Apple	23-28 (25.2)	1.77
iPhone X	Apple	27-32 (25.3)	0.43
iPad Air 3	Apple	22-31 (28.0)	3.72
Mi 8 Lite	Xiaomi	24-32 (27.4)	1.65
Pocophone	Xiaomi	22-29 (26.3)	0.7
Galaxy S9	Samsung	25-31 (27.2)	3.64

Diversity of effective distance: Our system can be deployed with various maximum effective distances with different smartphone recorders, ranging from 49 cm to 3.72 m, as shown in Table 6.3. The result also shows a great variance across recorders. We attribute this diversity to the difference in frequency response of these recorders, and the non-linearity of audio processing circuits.

Multiple recorders: Since the performance of NEC can be affected by the variance of hardware, we investigate whether NEC system can be used to support multiple recorders simultaneously. To conduct this experiment, we use Moto Z4, Mi 8 Lite, POCOPHONE, and Galaxy S9 as recorders to eavesdrop on Bob's voice. With the collected recorded audios, we compute the SDR for recorded audios. For comparison, the SDR of mixed audio is also calculated to reveal the effect of NEC. We define that, if the SDR of recorded audio is less than the mixed audio, NEC is successfully performed. Our experiment result is presented in Table 6.4. For three different carrier center frequency settings, we played 20 crafted mixed audios and run NEC to superpose shadow audio to affect three recorders' recording. The column named **1+**, **2+**, **3** means at least **1**, **2**, or **3** devices are affected simultaneously by NEC. And the reported values such as 20/20 denote that all the 20 recorded audios are unable to recognize Bob's voice. This result provides the supportive evidence that NEC is capable of operating in public and affecting multiple recorders by carefully tuning the system parameters.

Table 6.4: NEC's performance with multiple recorders.

Number of Recorder		**1+**	**2+**	**3**
f_c **(kHz)**	26.3	20/20	9/20	4/20
	27.2	20/20	15/20	11/20
	27.4	20/20	14/20	8/20

6.6: Discussion

Limitation of non-linear effect: The success of NEC relies on the imperfection of the receivers' (e.g., Alice's) microphone. However, when the non-linear effect is not present due to two reasons: 1) the great precision of Alice's microphone or 2) the improper modulation parameter settings, our selective voice protection will no longer be effective.

Limitation of protecting conversation: Although prior benchmark and user case experiments demonstrate that NEC can protect the target speaker's voice in the wild, it is a challenge to protect a conversation that involves multiple speakers while not disrupting other users (e.g., Alice). We failed to train a Selector model that is applicable to multiple target speakers with the current system architecture. In future work, we will figure out how to integrate the multiple speakers' embeddings and re-design the Selector model to avoid removing Alice's voice in the private conversation.

Directional of ultrasonic speaker: In our prototype shown in Figure 6.8, we assume the ultrasound speaker has the shadow audio ready before playing it. However, when we integrate the monitor, DNN models, and ultrasound speaker into one device and run it in a real-time manner, the shadow audio is dependent on the incoming mixed audio. In this case, one critical concern of NEC is whether the current mixed audio will be affected by the current shadow audio, therefore impacting the quality of future shadow audio. Fortunately, we can avoid it by putting the monitor and ultrasound speaker in *opposite direction*. We find that by exploiting the directional property of the ultrasound speaker, the shadow audio is barely sensed by the NEC's monitor as it produces limited amplitude in its back direction.

6.7: Related Work

Microphone jamming: Microphone jamming has been proposed [36, 116, 169] to protect private conversations. To avoid the recording of private conversations, a pre-configured audio jammer is deployed to emit the scrambling noise waves to disrupt the speech recording. Specifically, Chen et al. [36] adopt the white noise to distort the microphone recordings, while Tung et al. [169] explore the sound masking with the specially designed scramble noise to obfuscate the spoken sensitive information. Patronus [116] emits ultrasound to generate the scrambling waves at the recorder without introducing human-sensitive noise. In contrast, rather than canceling and jamming by the low-level signal features (e.g., frequency, phase), we use high-level human vocal features to generate a shadow sound for speaker-selective jamming.

AI-augmented speaker diarization: AI plays important role of processing signal [74, 113, 114, 220]. Recent studies [41, 181, 182] propose AI-based speaker diarization, a process to partition multi-speaker audio into homogeneous single speaker segments based on the speaker identity. It effectively solves "who spoke when" in a multi-speaker scenario. Several audio embedding models have been proposed for speaker-specific feature extraction, including speaker factor [28], i-vector [147, 221], and d-vector [181, 182, 208]. Based on these features, a number of classification models have been designed to extract the speaker-specific embedded audios, such as clustering algorithms [147, 181, 221], DNN model [182, 208], and even an integrated model with visual information (e.g., lip movement and face recognition) [6, 41, 52]. However, these methods cannot be adopted in our scenario. First, all existing speaker diarization models are used for post-processing after the audio is recorded, but we need to deal with voice cancellation in an end-to-end fashion. Additionally, the processing delay is an important factor to guarantee an effective shadow sound generation, which has been ignored by these post-processing models. In this work, we design the adaptive features, DNN structures, and training methods to realize an end-to-end voice cancellation system to protect a target speaker's voice.

6.8: Summary

We present NEC, a lightweight AI-augmented voice protection system to protect the target speech without interfering with others' audio conversations. As an end-to-end processing system, NEC first actively emits specially designed ultrasound signals to a recorder. Due to the non-linearity effect, a shadow sound is generated and superposed onto the received mixed sound at the recorder, which effectively cancels the target speaker's voice in the recordings. To determine the frequency composition of the shadow sound, NEC leverages a tailored Deep Neural Network (DNN) to extract high-level speaker-specific but utterance-independent vocal features from the mixed sound. By imitating the overshadowing in the air, we superpose the shadow audio with the mixed audio in the training stage of the DNN model and deliver a one-fits-all model, which can be trained only once and deployed directly for new users. Our experimental evaluations demonstrate NEC's efficacy in a wide variety of real-world scenarios. The results show that NEC effectively disables the microphones from recording the target speaker's voice.

CHAPTER 7: CONCLUSION AND FUTURE WORK

Voice serves as a key medium for human interaction, not only with each other but also with computers and intelligent devices. It facilitates information exchange between individuals and allows users to command AI agents or smart devices through speech. Additionally, voice characteristics are employed by cloud services for user authentication. However, with the widespread use of voice in contemporary communication, concerns regarding its security in various contexts have emerged within the research community. Potential risks include adversaries circumventing voice-based authentication systems to access personal data or manipulate devices such as mobile phones, customer accounts, or smart speakers. Furthermore, speech recognition models could be compromised, leading to deceptive outputs from smart speakers or speech-to-text services, such as unauthorized door opening or the generation of inappropriate responses. Privacy issues also arise, with attackers potentially eavesdropping and using voice cloning techniques to execute speech synthesis attacks. This dissertation focuses on identifying vulnerabilities in AI-powered voice systems and developing defensive strategies against these threats. It concentrates on three primary areas: speaker authentication, speech recognition, and privacy protection. The core methodology of our present and future research is to meticulously address the three areas, thereby fostering the adoption of voice-enabled AI systems in the modern world.

In summary, this dissertation not only presents a series of significant research findings and developments in the field of voice technology but also provides a comprehensive and detailed overview of the current state of speaker authentication, speech recognition, and privacy protection. The work represents a substantial contribution to the field and sets the stage for future research and development in these crucial areas. Our work's importance reaches far beyond academic circles, offering substantial and diverse benefits to different areas within the security sector and society in general. First, for the security community, our study serves as a groundbreaking resource. It sheds light on the less explored aspects of voice-enabled AI systems, an area experiencing rapid growth. By identifying these vulnerabilities, we empower cybersecurity professionals to build stronger and more effective defenses against potential attacks. This forward-thinking strategy is vital in an era where technological progress often surpasses security protocols. Second, our research has a significant impact on the well-being of society. With voice-enabled devices increasingly common in homes and public spaces, securing these devices is crucial. Our findings play a key role in protecting the privacy and safety of the broader public, who are frequently the unintended victims of security lapses. To conclude, our in-depth analysis and the proactive approaches we propose to the issues in AI-driven voice interaction systems mark a considerable advancement. We introduce a crucial perspective on security in a field that is essential to our society's technological progress. Our work establishes a foundation for more secure and reliable AI interactions, benefiting both the security community and the wider society.

7.1: Summary of Contributions

This dissertation presents a comprehensive collection of five of my scholarly publications, delving into the critical areas of speaker authentication, speech recognition, and privacy issues.

In speaker authentication, we introduce SuperVoice, a speaker verification system that utilizes ultrasound features in human voice to verify speaker identities. This system demonstrates robustness and accuracy across various environments. Furthermore, we identify new vulnerabilities in existing speaker verification models. Specifically, we introduce a novel attack named MasterKey and showcase the potential risks and impacts of such attacks on speaker verification systems. For speech recognition, this dissertation presents two attack methodologies. The first, named SpecPatch, is designed for scenarios that involve direct human interaction, and the false command injection attack could succeed even when users are actively engaging with speech-to-text systems. The second methodology, PhantomSound, targets black-box systems, where direct interaction or insight into the system's inner workings is limited. Both approaches represent significant advancements in understanding the vulnerabilities of speech recognition systems. Lastly, the dissertation addresses the important issue of data privacy in voice technology. In response to the growing concerns about unauthorized voice recordings and eavesdropping, we introduce NEC, an innovative jamming device engineered to prevent unauthorized voice recordings without interfering with authorized ones. Specifically, this dissertation makes the following contributions:

Robust speaker authentication system: In this dissertation, we introduce SuperVoice, an advanced speaker verification system that enhances secure speaker authentication by utilizing ultrasound characteristics present in human speech. This system represents a departure from conventional speaker verification methods that rely on spectrographic features derived from the audible frequency range of voice commands. Instead, we venture into a novel area of human voice research by examining the distinct traits of human speech in the ultrasound frequency band. Our findings reveal that the high-frequency components of ultrasound in human speech, particularly those in the 20 to 48 kHz range such as speech fricatives, can markedly improve the security and precision of speaker verification systems. This approach opens new possibilities for leveraging ultrasound features to bolster the effectiveness of speaker authentication technologies.

Attacking speaker authentication models: This dissertation introduces a novel threat to speaker authentication systems, termed MasterKey. This backdoor attack is designed to compromise various speaker verification (SV) models, targeting a real-world scenario where the attacker lacks knowledge about the specific victim. The development of MasterKey involved a thorough examination of the limitations inherent in existing poisoning attacks aimed at unseen targets. Our approach led to the optimization of a universal backdoor capable of attacking any target. We further refined the backdoor by embedding subtle characteristics of the speaker's voice and semantic information, rendering it virtually undetectable. Additionally, we accounted for channel distortion, incorporating this element into the backdoor's design. We successfully attacked 53 speaker verification models, involved 16,430 enrolled speakers. Remarkably, our attack achieved a 100% success rate with a 15% poisoning rate. We conducted validation of our attack in three real-world settings, successfully executing the attack both over the air and via telephony lines. This comprehensive testing underscores the effectiveness of MasterKey and highlights the critical need for enhanced security measures in speaker authentication systems.

Attacking speech recognition system when Human-in-the-loop: This dissertation introduces SpecPatch, a new adversarial audio attack targeting automated speech recognition (ASR) systems, uniquely involving human interaction in the process. Traditional audio adversarial attacks operate under the assumption that users will not detect the adversarial audio, thereby ensuring the effective delivery of manipulated examples or perturbations. However, this overlooks a critical aspect of real-world scenarios: users of intelligent voice-controlled devices are often alert to any unusual sounds, particularly when issuing voice commands. If users perceive any suspicious audio, they tend to counteract it by interrupting the adversarial audio and overpowering the malicious voice with stronger, corrective commands. This user vigilance renders most existing attacks ineffective when user interaction and adversarial audio delivery happen simultaneously. To address this challenge and enable a truly imperceptible and robust adversarial attack that can withstand potential user interruptions, we developed SpecPatch. This practical voice attack employs a sub-second audio patch signal to initiate an attack command, coupled with periodic noises designed to disrupt communication between the user and the ASR system. In comparison to existing methods, SpecPatch significantly extends the attack impact length (by 287%), effectively lengthening the duration of the target command. Moreover, we demonstrate that our attack maintains a 100% success rate in both over-the-line and over-the-air scenarios, even in the face of user intervention.

Attacking commercial speech recognition services: This dissertation introduces PhantomSound, a black-box attack designed for commercial voice assistants. Traditional black-box adversarial attacks on voice assistants often involve a substantial number of queries and an extensive training phase. PhantomSound, on the other hand, employs a decision-based attack strategy to efficiently produce effective adversarial audios, significantly reducing the number of required queries by optimizing gradient estimation. Our experiments involved testing PhantomSound against four different speech-to-text APIs in three real-world scenarios to assess its real-time impact. The results confirm that PhantomSound is both practical and robust, capable of successfully attacking five popular commercial voice-controlled devices over the air. It also demonstrates the ability to circumvent three liveness detection mechanisms with a success rate exceeding 95%. Compared to leading black-box attacks, PhantomSound achieves a reduction in required queries by 93.1% for untargeted attacks (approximately 300 queries or around 5 minutes) and 65.5% for targeted attacks (roughly 1,500 queries or about 25 minutes). This efficiency makes PhantomSound a significant advancement in the realm of black-box adversarial attacks on voice assistants.

Protecting unauthorized recording: This dissertation introduces NEC (Neural Enhanced Cancellation), a novel defense mechanism designed to protect the privacy of everyday conversations by preventing unauthorized microphones from recording a target speaker's voice. NEC offers a significant advancement over existing audio cancellation techniques, which typically rely on scrambling methods. Unlike these methods, NEC is capable of selectively eliminating a target speaker's voice from a mixed speech environment without disrupting others. The core of NEC's functionality lies in a specially designed Deep Neural Network (DNN) model. This model is trained to isolate high-level vocal features that are specific to the target speaker but independent of their particular utterances. These features are extracted from the speaker's reference audio samples. When a microphone is in operation, the DNN actively generates a 'shadow sound' that effectively cancels out the target voice in real-time. NEC has been thoroughly implemented and evaluated using 8 different smartphone microphones across various settings. The results from these evaluations demonstrate that NEC is highly effective in muting the target speaker at a microphone, all while ensuring that other users can carry on their conversations without any interference. This makes NEC a highly promising solution for enhancing privacy in daily communications.

7.2: Limitations and Discussion

Although the research described in this dissertation makes a significant contribution to the field of Voice AI system security, our work has limitations and room for further improvement.

Robust speaker authentication system: In this dissertation, we have proposed a secured speaker authentication system called SuperVoice. Although this system show robustness and efficient in distinguishing speakers and defend against fake speech, it has some limitations. First, we noted that certain phonemes, particularly fricatives and stops, have high energy above 20 kHz. However, sentences without fricatives might lack this energy spike. Despite this, SuperVoice often detects High-Frequency Energy (HFE) in non-fricative commands due to airflow alterations by the speaker. This HFE acts as an additional feature for speaker verification. Even for sentences with predominantly low-frequency energy (below 8 kHz), SuperVoice's dual-frequency stream architecture ensures consistent performance, effectively utilizing high-frequency features from non-fricative commands. Second, we found that in SuperVoice, the distance affects both low and high-frequency components' power, particularly for fricatives and plosives, therefore affecting the performance to verify speakers in long distance. A potential solution like a power amplifier could address this attenuation, and its effectiveness in long-distance verification will be explored in future work.

Attacking speaker authentication models: In this dissertation, we introduce an innovative attack named MasterKey against the large-scale speaker authentication models, however, it has some limitations that need to be addressed. First, this attack relies on the speaker authentication model's maintainer to use the attacker-prepared dataset to fine-tune the commercial model. This is a strong assumption because companies intend to use the local/private dataset to enhance their model. Second, the attack is possibly not robust to re-training defense. In this case, the model maintainer can re-train and fine-tune the backdoored model on the benign dataset, which will lead to the invalidate of the backdoor attack. To enhance the robustness of our attack, one possible solution is to craft an out-of-domain backdoor sample, so that the re-training will not affect the backdoor-target mapping because the benign dataset does not include the out-of-domain backdoor distribution.

Attacking speech recognition system: In this dissertation, we present two approaches to attack the speech recognition system. The first attack, SpecPatch, successfully attacks the speech recognition model while human-in-the-loop. However, SpecPatch presents several constraints: 1) its dependency on specific models; 2) failure to inject very long target sentences; and 3) a limited effective attack range. Addressing the first constraint, the attack is tailored to target recurrent neural networks, exploiting vulnerabilities in the connections between individual cells. The second issue could potentially be resolved by using a longer patch, although this might increase the likelihood of alerting the target. As for the third constraint, enhancing the power of the patch could extend its attack range. However, this approach requires managing distortions caused by the amplifier and attenuation due to increased distance.

Our second attack, PhantomSound, is designed to attack commercial speech-to-text API and voice assistants. Although PhantomSound demonstrated the ability to craft voice adversarial examples promptly. It has the following limitations. First, the PhantomSound shows vulnerability to the presence of ambient noise; Second, the PhantomSound is constrained to consistently create adversarial examples (AEs) for every input and target; Third, this attack is struggling to significantly alter lengthy sentences; Last, the capability of launching a long-range attack is limited. To mitigate the first and fourth limitations, an attacker could either boost the strength of the perturbation or choose a quieter environment for the attack. The second and third limitations might be overcome by employing multiple iterations of phoneme injections. While this increases the chances of producing a successful perturbation, it also potentially escalates the cost and effort required.

Protecting unauthorized recording: This dissertation introduces a smart microphone jammer that is capable of jamming a specific user's voice on the attacker's microphone. Even though the idea and prototype are promising, there are some limitations. First, the effectiveness of NEC is contingent on the imperfections in the receiver's microphone. Its selective voice protection fails when the non-linear effect is absent, either due to the high precision of the microphone or incorrect modulation parameter settings. Second, NEC faces challenges in safeguarding conversations involving multiple speakers without disrupting others. The current system architecture does not support a Selector model capable of handling multiple target speakers. Future work will explore integrating multiple speaker embeddings and redesigning the Selector model to prevent the inadvertent removal of voices from private conversations. Third, the jamming performance is affected by the direction of the ultrasound signal. The prototype assumes the ultrasound speaker is pre-loaded with shadow audio. When integrating the monitor, DNN models, and ultrasound speaker into a single device for real-time operation, the shadow audio depends on the incoming mixed audio. A key concern is whether the current shadow audio might affect the quality of future shadow audio. This issue can be mitigated by positioning the monitor and ultrasound speaker in opposite directions, utilizing the directional nature of the ultrasound speaker to ensure that the shadow audio has minimal impact on the NEC's monitor.

7.3: Future Work

I will continue advancing the frontier of the adoption of AI-enabled systems by addressing their security, privacy, and usability challenges. Some future works are listed as follows.

Defend against telecommunications fraud: Current telecommunications fraud creates deepfake sound of the victim through speech synthesis techniques (generative networks), thereby gaining the trust of the victim's relatives, to achieve the purpose of fraud. My observations on voice adversarial attacks show that the victim can protect their sound misuse by adding perturbations before they upload to the Internet. However, this approach's performance has suffered because of the diversity of the generative models and multiple signal-processing tools. In my future study, I will borrow the black-box attack idea to train a generalized model to simulate the adversary's generative model and craft robust universal perturbations to mislead the identity of the generated fake audio. I firmly believe that this approach has the potential to be developed into a generalized defense framework to safeguard the deepfake audio.

Advancing secure interaction for large language models in smart speakers: The prospective integration of Large Language Models (LLMs) like ChatGPT with smart speakers such as Alexa is poised to significantly enhance daily life assistance through intelligent and responsive interactions. However, this amalgamation brings forth critical concerns and challenges related to security, efficiency, and accuracy that need meticulous investigation and resolution. My future work is committed to pioneering advancements in this domain, focusing on developing secure, efficient, and high-performing smart speakers embedded with sophisticated LLM logic. We aim to delve deep into the security vulnerabilities inherent in these integrations, emphasizing the creation of robust authentication mechanisms to mitigate unauthorized access and potential malicious activities. A pivotal area of our research will be to refine the accuracy of speech-to-text conversion processes and the model's logical comprehension of commands (chain of thought), ensuring precise interpretation and execution of user instructions. We will explore optimizations to meet the computational demands of LLMs and investigate energy-efficient solutions to address increased power consumption. Additionally, we aspire to implement mechanisms enabling smart speakers to justify proposed actions and seek user confirmations, enhancing user control and satisfaction. Through our endeavors, we anticipate contributing to the development and adoption of secure, user-friendly, and intelligent smart speakers, paving the way for the next era of smart assistance technologies.

Voice watermarking for IP protection: Watermarking is usually used for protecting the copyright. In my future work, I will aim to fortify the integrity of speech datasets against unauthorized usage by adding voice watermarks. The watermark can be a different style of speech and will be used to verify the ownership of the dataset. Once a suspicious model uses our dataset without permission, the fine-tuning of our dataset will leave a watermark on the model. Compared to the existing watermark approach which injects backdoors into the trained model and potentially harms legitimate dataset users (e.g., leaving a backdoor to their model). We aim to make a harmless watermark by injecting different styles of speech (with correct labels) to serve as a watermark. The watermarked audio samples are intricately designed to be challenging for speech recognition systems to decode correctly and will be incorporated seamlessly into the dataset, thus acting as a protective shield without compromising the audio quality. This approach probably offers a reliable protection mechanism for speech datasets and encourages the responsible and ethical use of data resources.

BIBLIOGRAPHY

[1] TIMIT Acoustic-Phonetic Continuous Speech Corpus. https://catalog.ldc.upenn.edu/LDC93S1, 1993. Accessed: 2021-11-04.

[2] Persian Vowel recognition with MFCC and ANN on PCVC speech dataset. https://github.com/smalekz/PCVC, 2018. Accessed: 2021-11-04.

[3] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. TensorFlow: A system for Large-Scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[4] Hadi Abdullah, Washington Garcia, Christian Peeters, Patrick Traynor, Kevin RB Butler, and Joseph Wilson. Practical hidden voice attacks against speech and speaker recognition systems. In *Network and Distributed Systems Security (NDSS) Symposium*, 2019.

[5] Hadi Abdullah, Muhammad Sajidur Rahman, Washington Garcia, Kevin Warren, Anurag Swarnim Yadav, Tom Shrimpton, and Patrick Traynor. Hear" no evil", see" kenansville"*: Efficient and transferable black-box attacks on speech recognition and voice identification systems. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 712–729. IEEE, 2021.

[6] T. Afouras, J. S. Chung, and A. Zisserman. The conversation: Deep audio-visual speech enhancement. In *INTERSPEECH*, 2018.

[7] Muhammad Ejaz Ahmed, Il-Youp Kwak, Jun Ho Huh, Iljoo Kim, Taekkyung Oh, and Hyoungshick Kim. Void: A fast and light voice liveness detection system. In *USENIX Security*, 2020.

[8] Muhammad Ejaz Ahmed, Il-Youp Kwak, Jun Ho Huh, Iljoo Kim, Taekkyung Oh, and Hyoungshick Kim. Void: A fast and light voice liveness detection system. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 2685–2702, 2020.

[9] Federico Alegre, Artur Janicki, and Nicholas Evans. Re-assessing the threat of replay spoofing attacks against automatic speaker verification. In *2014 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–6. IEEE, 2014.

[10] Federico Alegre, Ravichander Vipperla, Nicholas Evans, and Benoït Fauve. On the vulnerability of automatic speaker recognition to spoofing attacks with artificial signals. In *2012 Proceedings of the 20th european signal processing conference (EUSIPCO)*, pages 36–40. IEEE, 2012.

[11] Raziel Alvarez and Hyun-Jin Park. End-to-end streaming keyword spotting. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6336–6340. IEEE, 2019.

[12] Moustafa Alzantot, Bharathan Balaji, and Mani Srivastava. Did you hear that? adversarial examples against automatic speech recognition. In *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.

[13] Amazon. Amazon Echo. https://www.amazon.com/All-New-Echo-4th-Gen/dp/B07XKF5RM3, 2021.

[14] Amazon. Amazon Transcribe. https://aws.amazon.com/transcribe/, 2021.

[15] Amazon. Amazon Transcribe. https://aws.amazon.com/transcribe/, 2021.

[16] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR, 2016.

[17] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018.

[18] Avisoft. http://www.avisoft.com/ultrasound-microphones/cm16-cmpa/.

[19] Avisoft. http://www.avisoft.com/playback/vifa/.

[20] AWS. *AWS Voice ID*. AWS, September 2022. https://aws.amazon.com/connect/voice-id/.

[21] Avisoft Bioacoustics. http://www.avisoft.com/playback/power-amplifier/.

[22] Bose. https://www.bose.com/en_us/support/products/bose_speakers_support/bose_smarthome_speakers_support/soundtouch-10-wireless-system.html.

[23] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[24] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

[25] William M Campbell, Douglas E Sturim, and Douglas A Reynolds. Support vector machines using gmm supervectors for speaker verification. *IEEE signal processing letters*, 13(5):308–311, 2006.

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

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

[28] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair. Stream-based speaker segmentation using speaker factors and eigenvoices. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008.

[29] Chase. *Chase Voice ID*. Chase, September 2022. https://www.chase.com/personal/voice-biometrics.

[30] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

[31] Guangke Chen, Sen Chenb, Lingling Fan, Xiaoning Du, Zhe Zhao, Fu Song, and Yang Liu. Who is real bob? adversarial attacks on speaker recognition systems. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 694–711. IEEE, 2021.

[32] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1277–1294. IEEE, 2020.

[33] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 15–26, 2017.

[34] Tao Chen, Longfei Shangguan, Zhenjiang Li, and Kyle Jamieson. Metamorph: Injecting inaudible commands into over-the-air voice controlled systems. In *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

[35] Yu-hsin Chen, Ignacio Lopez-Moreno, Tara N Sainath, Mirkó Visontai, Raziel Alvarez, and Carolina Parada. Locally-connected and convolutional neural networks for small footprint speaker recognition. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[36] Yuxin Chen, Huiying Li, Shan-Yuan Teng, Steven Nagels, Zhijing Li, Pedro Lopes, Ben Y Zhao, and Haitao Zheng 0001. Wearable Microphone Jamming. *CHI*, 2020.

[37] Yuxuan Chen, Xuejing Yuan, Jiangshan Zhang, Yue Zhao, Shengzhi Zhang, Kai Chen, and XiaoFeng Wang. {Devil's} whisper: A general approach for physical adversarial attacks against commercial black-box speech recognition devices. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 2667–2684, 2020.

[38] Minhao Cheng, Simranjit Singh, Patrick Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh. Sign-opt: A query-efficient hard-label adversarial attack. *arXiv preprint arXiv:1909.10773*, 2019.

[39] Noam Chomsky and Morris Halle. The sound pattern of english. 1968.

[40] Anurag Chowdhury and Arun Ross. Fusing mfcc and lpc features using 1d triplet cnn for speaker recognition in severely degraded audio signals. *IEEE Transactions on Information Forensics and Security*, 2019.

[41] Joon Son Chung, Jaesung Huh, Arsha Nagrani, Triantafyllos Afouras, and Andrew Zisserman. Spot the conversation: speaker diarisation in the wild. *ArXiv*, 2020.

[42] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. *arXiv preprint arXiv:1806.05622*, 2018.

[43] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. *arXiv preprint arXiv:1806.05622*, 2018.

[44] Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*, 2017.

[45] Donghui Dai, Zhenlin An, and Lei Yang. Inducing wireless chargers to voice out for inaudible command attacks. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1789–1806. IEEE, 2023.

[46] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2010.

[47] Jiangyi Deng, Yanjiao Chen, and Wenyuan Xu. Fencesitter: Black-box, content-agnostic, and synchronization-free enrollment-phase attacks on speaker recognition systems. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 755–767, 2022.

[48] Srinivas Desai, E Veera Raghavendra, B Yegnanarayana, Alan W Black, and Kishore Prahallad. Voice conversion using artificial neural networks. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3893–3896. IEEE, 2009.

[49] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification. *arXiv preprint arXiv:2005.07143*, 2020.

[50] Wei Du, Peixuan Li, Boqun Li, Haodong Zhao, and Gongshen Liu. Uor: Universal backdoor attacks on pre-trained language models. *arXiv preprint arXiv:2305.09574*, 2023.

[51] Eastern. *Easternbank Voice ID*. Eastern, September 2022. https://www.easternbank.com/personal-banking/mobile-online/voice-id.

[52] Ariel Ephrat, Inbar Mosseri, Oran Lang, Tali Dekel, Kevin Wilson, Avinatan Hassidim, William T. Freeman, and Michael Rubinstein. Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. *ACM Trans. Graph.*, 2018.

[53] Navy Federal. *Navy Federal Credit Union Voice ID*. Navy Federal, September 2022. https://www.navyfederal.org/services/security/voice-id.html.

[54] Huan Feng, Kassem Fawaz, and Kang G Shin. Continuous authentication for voice assistants. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, pages 343–355, 2017.

[55] FirstHorizon. *FirstHorizon Voice ID*. FirstHorizon, September 2022. https://www.firsthorizon.com/Personal/Support/Security-and-Fraud-Protection/Voice-Biometrics.

[56] Mario Fleischer, Silke Pinkert, Willy Mattheus, Alexander Mainka, and Dirk Mürbe. Formant frequencies and bandwidths of the vocal tract transfer function are affected by the mechanical impedance of the vocal tract wall. *Biomechanics and modeling in mechanobiology*, 14(4):719–733, 2015.

[57] Daniel Galvez, Greg Diamos, Juan Ciro, Juan Felipe Cerón, Keith Achorn, Anjali Gopi, David Kanter, Maximilian Lam, Mark Mazumder, and Vijay Janapa Reddi. The people's speech: A large-scale diverse english speech recognition dataset for commercial usage. *arXiv preprint arXiv:2111.09344*, 2021.

[58] John S Garofolo. Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium, 1993*, 1993.

[59] Yuan Gong and Christian Poellabauer. Crafting adversarial examples for speech paralinguistics applications. *arXiv preprint arXiv:1711.03280*, 2017.

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

[61] Google. Google Assistant. https://assistant.google.com/, 2021.

[62] Google. Google Home/Nest. https://store.google.com/product, 2021.

[63] Google. Google Speech. https://cloud.google.com/speech-to-text, 2021.

[64] Google. Google Speech. https://cloud.google.com/speech-to-text, 2021.

[65] Google. Google Vision. https://cloud.google.com/vision, 2021.

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

[67] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[68] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[69] Hanqing Guo, Xun Chen, Junfeng Guo, Li Xiao, and Qiben Yan. Masterkey: Practical backdoor attack against speaker verification systems. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–15, 2023.

[70] Hanqing Guo, Chenning Li, Lingkun Li, Zhichao Cao, Qiben Yan, and Li Xiao. Nec: Speaker selective cancellation via neural enhanced ultrasound shadowing. In *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 355–366. IEEE, 2022.

[71] Hanqing Guo, Guangjing Wang, Yuanda Wang, Bocheng Chen, Qiben Yan, and Li Xiao. Phantomsound: Black-box, query-efficient audio adversarial attack via split-second phoneme injection. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 366–380, 2023.

[72] Hanqing Guo, Yuanda Wang, Nikolay Ivanov, Li Xiao, and Qiben Yan. Specpatch: Human-in-the-loop adversarial audio spectrogram patch attack on speech recognition. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 1353–1366, 2022.

[73] Hanqing Guo, Qiben Yan, Nikolay Ivanov, Ying Zhu, Li Xiao, and Eric J Hunter. Supervoice: Text-independent speaker verification using ultrasound energy in human speech. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, pages 1019–1033, 2022.

[74] Hanqing Guo, Nan Zhang, Shaoen Wu, and Qing Yang. Deep learning driven wireless real-time human activity recognition. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[75] Junfeng Guo, Ang Li, and Cong Liu. AEVA: Black-box backdoor detection using adversarial extreme value analysis. In *International Conference on Learning Representations*, 2022.

[76] Junfeng Guo, Ang Li, and Cong Liu. Backdoor detection and mitigation in competitive reinforcement learning, 2023.

[77] Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. Scale-up: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. *arXiv preprint arXiv:2302.03251*, 2023.

[78] Junfeng Guo and Cong Liu. Practical poisoning attacks on neural networks. In *ECCV*, 2020.

[79] Junfeng Guo and Cong Liu. Practical poisoning attacks on neural networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, pages 142–158. Springer, 2020.

[80] Wei Guo, Benedetta Tondi, and Mauro Barni. A master key backdoor for universal impersonation attack against dnn-based face verification. *Pattern Recognition Letters*, 144:61–67, 2021.

[81] Nawar Halabi. Arabic Speech Corpus. http://en.arabicspeechcorpus.com/, 2016. Accessed: 2021-11-04.

[82] Hasan Abed Al Kader Hammoud, Shuming Liu, Mohammad Alkhrasi, Fahad AlBalawi, and Bernard Ghanem. Look, listen, and attack: Backdoor attacks against video action recognition. *arXiv preprint arXiv:2301.00986*, 2023.

[83] A. Hannun. Sequence modeling with ctc. https://distill.pub/2017/ctc, 2017.

[84] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[85] HarryVolek. HarryVolek GE2E. https://github.com/HarryVolek, 2018.

[86] Shoji Hayakawa and Fumitada Itakura. Text-dependent speaker recognition using the information in the higher frequency band. In *Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages I–137. IEEE, 1994.

[87] Shoji Hayakawa and Fumitada Itakura. The influence of noise on the speaker recognition performance using the higher frequency band. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 321–324. IEEE, 1995.

[88] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[89] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[90] Yitao He, Junyu Bian, Xinyu Tong, Zihui Qian, Wei Zhu, Xiaohua Tian, and Xinbing Wang. Canceling inaudible voice commands against voice control systems. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–15, 2019.

[91] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer. End-to-end text-dependent speaker verification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5115–5119. IEEE, 2016.

[92] Yedid Hoshen, Ron J Weiss, and Kevin W Wilson. Speech acoustic modeling from raw multichannel waveforms. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4624–4628. IEEE, 2015.

[93] HOTENDA. https://www.hotenda.com/datasheet-html/2493/1/SPU0410LR5H-QB.html, 2013.

[94] HSBC. *HSBC Voice ID*. HSBC, September 2022. https://www.us.hsbc.com/customer-service/voice/x.

[95] Chien-yu Huang, Yist Y Lin, Hung-yi Lee, and Lin-shan Lee. Defending your voice: Adversarial attack on voice conversion. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 552–559. IEEE, 2021.

[96] Shehzeen Hussain, Paarth Neekhara, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. {WaveGuard}: Understanding and mitigating audio adversarial examples. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2273–2290, 2021.

[97] IBM. IBM Speeche. https://www.ibm.com/cloud/watson-speech-to-text, 2021.

[98] Magdalena Igras, Bartosz Ziółko, and Mariusz Ziółko. Length of phonemes in a context of their positions in polish sentences. In *2013 International Conference on Signal Processing and Multimedia Applications (SIGMAP)*, pages 59–64. IEEE, 2013.

[99] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018.

[100] Acoustics — Normal equal-loudness-level contours. Standard, August 2003.

[101] Roman Jakobson, C Gunnar Fant, and Morris Halle. Preliminaries to speech analysis: The distinctive features and their correlates. 1951.

[102] Artur Janicki, Federico Alegre, and Nicholas Evans. An assessment of automatic speaker verification vulnerabilities to replay spoofing attacks. *Security and Communication Networks*, 9(15):3030–3044, 2016.

[103] N. S. Jayant and Peter Noll. *Digital Coding of Waveforms, Principles and Applications to Speech and Video*, page 688. Prentice-Hall, Englewood Cliffs NJ, USA, 1984. N. S. Jayant: Bell Laboratories; ISBN 0-13-211913-7.

[104] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. In *Advances in neural information processing systems*, pages 4480–4490, 2018.

[105] Allard Jongman. Acoustics of american english speech: A dynamic approach. *Language and Speech*, 1995.

[106] Allard Jongman, Ratree Wayland, and Serena Wong. Acoustic characteristics of english fricatives. *The Journal of the Acoustical Society of America*, 108(3):1252–1263, 2000.

[107] Patrick Kenny. Joint factor analysis of speaker and session variability: Theory and algorithms. *CRIM, Montreal,(Report) CRIM-06/08-13*, 14:28–29, 2005.

[108] Patrick Kenny, Themos Stafylakis, Pierre Ouellet, Md Jahangir Alam, and Pierre Dumouchel. Plda for speaker verification with utterances of arbitrary duration. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7649–7653. IEEE, 2013.

[109] Tomi Kinnunen and Haizhou Li. An overview of text-independent speaker recognition: From features to supervectors. *Speech communication*, 52(1):12–40, 2010.

[110] Tomi Kinnunen, Md Sahidullah, Héctor Delgado, Massimiliano Todisco, Nicholas Evans, Junichi Yamagishi, and Kong Aik Lee. The asvspoof 2017 challenge: Assessing the limits of replay spoofing attack detection. 2017.

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

[112] Galina Lavrentyeva, Sergey Novoselov, Egor Malykh, Alexander Kozlov, Oleg Kudashev, and Vadim Shchemelinin. Audio replay attack detection with deep learning frameworks. In *Interspeech*, pages 82–86, 2017.

[113] Chenning Li, Zhichao Cao, and Yunhao Liu. Deep ai enabled ubiquitous wireless sensing: A survey. *ACM Computing Surveys (CSUR)*, 54(2):1–35, 2021.

[114] Chenning Li, Manni Liu, and Zhichao Cao. Wihf: Enable user identified gesture recognition with wifi. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 586–595. IEEE, 2020.

[115] Gen Li, Zhichao Cao, and Tianxing Li. Echoattack: Practical inaudible attacks to smart earbuds. In *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services*, pages 383–396, 2023.

[116] Lingkun Li, Manni Liu, Yuguang Yao, Fan Dang, Zhichao Cao, and Yunhao Liu. Patronus: preventing unauthorized speech recordings with support for selective unscrambling. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys)*, pages 245–257, 2020.

[117] Zhuohang Li, Cong Shi, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. Practical adversarial attacks against speaker recognition systems. In *Proceedings of the 21st international workshop on mobile computing systems and applications*, pages 9–14, 2020.

[118] Zhuohang Li, Cong Shi, Tianfang Zhang, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. Robust detection of machine-induced audio attacks in intelligent audio systems with microphone array. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1884–1899, 2021.

[119] Zhuohang Li, Yi Wu, Jian Liu, Yingying Chen, and Bo Yuan. 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*, pages 1121–1134, 2020.

[120] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Soc, 2018.

[121] A Löfqvist and B Mandersson. Long-time average spectrum of speech and voice analysis. *Folia phoniatrica*, 1987.

[122] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[123] Yan Meng, Jiachun Li, Matthew Pillari, Arjun Deopujari, Liam Brennan, Hafsah Shamsie, Haojin Zhu, and Yuan Tian. Your microphone array retains your identity: A robust voice liveness detection system for smart speakers. In *31th USENIX Security Symposium (USENIX Security 21)*, 2022.

[124] Yan Meng, Zichang Wang, Wei Zhang, Peilin Wu, Haojin Zhu, Xiaohui Liang, and Yao Liu. Wivo: Enhancing the security of voice control system via wireless signal in iot environment. In *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 81–90, 2018.

[125] Microsoft. Microsoft Azure. https://azure.microsoft.com/en-us/, 2021.

[126] Brian B Monson, Eric J Hunter, Andrew J Lotto, and Brad H Story. The perceptual significance of high-frequency energy in the human voice. *Frontiers in psychology*, 5:587, 2014.

[127] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

[128] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. Voxceleb: Large-scale speaker verification in the wild. *Computer Speech & Language*, 60:101027, 2020.

[129] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*, 2017.

[130] Satoshi Nakamura, Kazuo Hiyane, Futoshi Asano, Takanobu Nishiura, and Takeshi Yamada. Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition. 2000.

[131] Mahesh Kumar Nandwana, Julien van Hout, Mitchell McLaren, Allen R Stauffer, Colleen Richey, Aaron Lawson, and Martin Graciarena. Robust speaker recognition from distant speech under real reverberant environments using speaker embeddings. In *Interspeech*, pages 1106–1110, 2018.

[132] Tao Ni, Xiaokuan Zhang, Chaoshun Zuo, Jianfeng Li, Zhenyu Yan, Wubing Wang, Weitao Xu, Xiapu Luo, and Qingchuan Zhao. Uncovering user interactions on smartphones via contactless wireless charging side channels. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 3399–3415. IEEE, 2023.

[133] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[134] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[135] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[136] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International conference on machine learning*, pages 5231–5240. PMLR, 2019.

[137] Mirco Ravanelli and Yoshua Bengio. Speaker recognition from raw waveform with sincnet. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1021–1028. IEEE, 2018.

[138] Douglas A Reynolds, Thomas F Quatieri, and Robert B Dunn. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1-3):19–41, 2000.

[139] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. Backdoor: Making microphones hear inaudible sounds. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, pages 2–14, 2017.

[140] Nirupam Roy, Haitham Hassanieh, and Romit Roy Choudhury. Backdoor: Making microphones hear inaudible sounds. In *Proceedings of ACM MobiSys*, 2017.

[141] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. Inaudible voice commands: The {Long-Range} attack and defense. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 547–560, 2018.

[142] Sada. https://www.aliexpress.com/item/4001241222763.html.

[143] Tara N Sainath, Ron J Weiss, Andrew Senior, Kevin W Wilson, and Oriol Vinyals. Learning the speech front-end with raw waveform cldnns. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[144] Lea Schönherr, Thorsten Eisenhofer, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Imperio: Robust over-the-air adversarial examples for automatic speech recognition systems. In *Annual Computer Security Applications Conference*, pages 843–855, 2020.

[145] Martin F Schwartz. Identification of speaker sex from isolated, voiceless fricatives. *The Journal of the Acoustical Society of America*, 43(5):1178–1179, 1968.

[146] scikit fda. fetch phoneme. skfda.datasets.fetch_phoneme.html, 2021.

[147] M. Senoussaoui, P. Kenny, T. Stafylakis, and P. Dumouchel. A study of the cosine distance-based mean shift for telephone speech diarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2014.

[148] Giorgio Severi, Jim Meyer, Scott Coull, and Alina Oprea. {Explanation-Guided} backdoor poisoning attacks against malware classifiers. In *30th USENIX security symposium (USENIX security 21)*, pages 1487–1504, 2021.

[149] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31, 2018.

[150] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4779–4783. IEEE, 2018.

[151] Sheng Shen, Nirupam Roy, Junfeng Guan, Haitham Hassanieh, and Romit Roy Choudhury. Mute: Bringing iot to noise cancellation. In *Proceedings of ACM SIGCOMM*, 2018.

[152] Cong Shi, Tianfang Zhang, Zhuohang Li, Huy Phan, Tianming Zhao, Yan Wang, Jian Liu, Bo Yuan, and Yingying Chen. Audio-domain position-independent backdoor attack via unnoticeable triggers. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 583–595, 2022.

[153] Stephen Shum, Najim Dehak, Reda Dehak, and James R Glass. Unsupervised speaker adaptation based on the cosine similarity for text-independent speaker verification. In *Odyssey*, page 16, 2010.

[154] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[155] Berrak Sisman, Junichi Yamagishi, Simon King, and Haizhou Li. An overview of voice conversion and its challenges: From statistical modeling to deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:132–157, 2020.

[156] SiSonic. https://www.digikey.com/product-detail/en/knowles/SPU0410LR5H-QB-7/423-1139-1-ND/2420983.

[157] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur. Deep neural network embeddings for text-independent speaker verification. In *Interspeech*, pages 999–1003, 2017.

[158] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. Light commands: laser-based audio injection attacks on voice-controllable systems. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 2631–2648, 2020.

[159] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

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

[161] Marija Tabain. Variability in fricative production and spectra: Implications for the hyper-and hypo-and quantal theories of speech production. *Language and speech*, 44(1):57–93, 2001.

[162] Raphael Tang and Jimmy Lin. Deep residual learning for small-footprint keyword spotting. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5484–5488. IEEE, 2018.

[163] Rohan Taori, Amog Kamsetty, Brenton Chu, and Nikita Vemuri. Targeted adversarial examples for black box audio systems. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 15–20. IEEE, 2019.

[164] Siri Team. Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant. *Apple Machine Learning Journal*, 1(6), 2017.

[165] Siri Team. Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant. *Apple Machine Learning Journal*, 1(6), 2017.

[166] Ingo R Titze, Ronald J Baken, Kenneth W Bozeman, Svante Granqvist, Nathalie Henrich, Christian T Herbst, David M Howard, Eric J Hunter, Dean Kaelin, Raymond D Kent, et al. Toward a consensus on symbolic notation of harmonics, resonances, and formants in vocalization. *The Journal of the Acoustical Society of America*, 2015.

[167] Francis Tom, Mohit Jain, and Prasenjit Dey. End-to-end audio replay attack detection using deep convolutional networks with attention. In *Interspeech*, pages 681–685, 2018.

[168] Susanne Trauzettel-Klosinski, Klaus Dietz, IReST Study Group, et al. Standardized assessment of reading performance: The new international reading speed texts irest. *Investigative ophthalmology & visual science*, 53(9):5452–5461, 2012.

[169] Yu-Chih Tung and Kang G. Shin. Exploiting sound masking for audio privacy in smartphones. In *Proceedings of ACM Asia Conference on Computer and Communications Security*, 2019.

[170] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. 2018.

[171] usbank. How voice-activated devices work with banks. https://www.usbank.com/financialiq/manage-your-household/personal-finance/how-voice-activated-devices-work-with-banks.html, 2020.

[172] Andrew Varga and Herman JM Steeneken. Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech communication*, 1993.

[173] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. Deep neural networks for small footprint text-dependent speaker verification. In *2014 IEEE ICASSP*, pages 4052–4056. IEEE, 2014.

[174] Verizon. *Verizon Voice ID*. Verizon, September 2022. https://www.verizon.com/support/voice-id-faqs/.

[175] E. Vincent, R. Gribonval, and C. Fevotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 2006.

[176] L. Wan, Q. Wang, A. Papir, and I. L. Moreno. Generalized end-to-end loss for speaker verification. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[177] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4879–4883. IEEE, 2018.

[178] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4879–4883. IEEE, 2018.

[179] Vincent Wan and Steve Renals. Speaker verification using sequence discriminant support vector machines. *IEEE transactions on speech and audio processing*, 13(2):203–210, 2005.

[180] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[181] Quan Wang, Carlton Downey, Li Wan, Philip Andrew Mansfield, and Ignacio Lopz Moreno. Speaker diarization with lstm. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5239–5243. IEEE, 2018.

[182] Quan Wang, Hannah Muckenhirn, Kevin Wilson, Prashant Sridhar, Zelin Wu, John R. Hershey, Rif A. Saurous, Ron J. Weiss, Ye Jia, and Ignacio Lopez Moreno. Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking. In *Proceedings of Interspeech*, 2019.

[183] Yuanda Wang, Hanqing Guo, Guangjing Wang, Bocheng Chen, and Qiben Yan. Vsmask: Defending against voice synthesis attack via real-time predictive perturbation. *arXiv preprint arXiv:2305.05736*, 2023.

[184] Yuanda Wang, Hanqing Guo, and Qiben Yan. Ghosttalk: Interactive attack on smartphone voice system through power line. *arXiv preprint arXiv:2202.02585*, 2022.

[185] Yuanda Wang, Hanqing Guo, and Qiben Yan. Ghosttalk: Interactive attack on smartphone voice system through power line. In *Network and Distributed Systems Security (NDSS) Symposium*, 2022.

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

[187] Wechat. *Wechat Voice ID*. Wechat, May 2015. https://blog.wechat.com/2015/05/21/voiceprint-the-new-wechat-password/.

[188] Emily Wenger, Max Bronckers, Christian Cianfarani, Jenna Cryan, Angela Sha, Haitao Zheng, and Ben Y Zhao. " hello, it's me": Deep learning-based speech synthesis attacks in the real world. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 235–251, 2021.

[189] WILDLIFE. https://www.wildlifeacoustics.com/products/echo-meter-touch-2-pro-ios.

[190] Fritz Winckel and Thomas Binkley. Music, sound and sensation : a modern exposition. 1967.

[191] Shutong Wu, Jiongxiao Wang, Wei Ping, Weili Nie, and Chaowei Xiao. Defending against adversarial audio via diffusion model. *arXiv preprint arXiv:2303.01507*, 2023.

[192] Zhizheng Wu, Nicholas Evans, Tomi Kinnunen, Junichi Yamagishi, Federico Alegre, and Haizhou Li. Spoofing and countermeasures for speaker verification: A survey. *speech communication*, 66:130–153, 2015.

[193] Zhizheng Wu, Sheng Gao, Eng Siong Cling, and Haizhou Li. A study on replay attack and anti-spoofing for text-dependent speaker verification. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific*, pages 1–5. IEEE, 2014.

[194] Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. Patchguard: A provably robust defense against adversarial patches via small receptive fields and masking. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2237–2254, 2021.

[195] Chong Xiang and Prateek Mittal. Detectorguard: Provably securing object detectors against localized patch hiding attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3177–3196, 2021.

[196] Wayne Xiong, Lingfeng Wu, Fil Alleva, Jasha Droppo, Xuedong Huang, and Andreas Stolcke. The microsoft 2017 conversational speech recognition system. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5934–5938. IEEE, 2018.

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

[198] Chen Yan, Yan Long, Xiaoyu Ji, and Wenyuan Xu. The catcher in the field: A fieldprint based spoofing detection for text-independent speaker verification. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1215–1229, 2019.

[199] Chen Yan, Yan Long, Xiaoyu Ji, and Wenyuan Xu. The catcher in the field: A fieldprint based spoofing detection for text-independent speaker verification. In *Proceedings of ACM CCS*, 2019.

[200] Qiben Yan, Kehai Liu, Qin Zhou, Hanqing Guo, and Ning Zhang. Surfingattack: Interactive hidden attack on voice assistants using ultrasonic guided wave. In *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

[201] Qiben Yan, Kehai Liu, Qin Zhou, Hanqing Guo, and Ning Zhang. Surfingattack: Interactive hidden attack on voice assistants using ultrasonic guided wave. In *Proceedings of Network and Distributed Systems Security (NDSS) Symposium*, 2020.

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

[203] Ye-Yi Wang, A. Acero, and C. Chelba. Is word error rate a good indicator for spoken language understanding accuracy. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, 2003.

[204] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, XiaoFeng Wang, and Carl A Gunter. Commandersong: A systematic approach for practical adversarial voice recognition. In *27th USENIX Security Symposium (USENIX Security 18)*, pages 49–64, 2018.

[205] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[206] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. Narcissus: A practical clean-label backdoor attack with limited information. *arXiv preprint arXiv:2204.05255*, 2022.

[207] Tongqing Zhai, Yiming Li, Ziqi Zhang, Baoyuan Wu, Yong Jiang, and Shu-Tao Xia. Backdoor attack against speaker verification. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2560–2564. IEEE, 2021.

[208] Aonan Zhang, Chong Wang, John Paisley, Quan Wang, and Zhenyao Zhu. Fully supervised speaker diarization. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[209] Guoming Zhang, Xiaoyu Ji, Xinfeng Li, Gang Qu, and Wenyuan Xu. Eararray: Defending against dolphinattack via acoustic attenuation. 2021.

[210] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of the 2017 ACM CCS*, pages 103–117, 2017.

[211] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of ACM CCS*, 2017.

[212] Jie Zhang, Zhanyong Tang, Meng Li, Dingyi Fang, Petteri Nurmi, and Zheng Wang. Crosssense: Towards cross-site and large-scale wifi sensing. In *Proceedings of ACM MobiCom*, 2018.

[213] Linghan Zhang, Sheng Tan, and Jie Yang. Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication. In *Proceedings of the 2017 ACM CCS*, pages 57–71, 2017.

[214] Linghan Zhang, Sheng Tan, Jie Yang, and Yingying Chen. Voicelive: A phoneme localization based liveness detection for voice authentication on smartphones. In *Proceedings of the 2016 ACM CCS*, pages 1080–1091, 2016.

[215] Nan Zhang, Xianghang Mi, Xuan Feng, XiaoFeng Wang, Yuan Tian, and Feng Qian. Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1381–1396. IEEE, 2019.

[216] Ruiteng Zhang, Jianguo Wei, Wenhuan Lu, Longbiao Wang, Meng Liu, Lin Zhang, Jiayu Jin, and Junhai Xu. Aret: Aggregated residual extended time-delay neural networks for speaker verification. In *INTERSPEECH*, pages 946–950, 2020.

[217] Shihao Zhao, Xingjun Ma, Xiang Zheng, James Bailey, Jingjing Chen, and Yu-Gang Jiang. Clean-label backdoor attacks on video recognition models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14443–14452, 2020.

[218] Baolin Zheng, Peipei Jiang, Qian Wang, Qi Li, Chao Shen, Cong Wang, Yunjie Ge, Qingyang Teng, and Shenyi Zhang. Black-box adversarial attacks on commercial speech platforms with minimal information. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 86–107, 2021.

[219] Yue Zheng, Yi Zhang, Kun Qian, Guidong Zhang, Yunhao Liu, Chenshu Wu, and Zheng Yang. Zero-effort cross-domain gesture recognition with wifi. In *Proceedings of ACM MobiSys*, 2019.

[220] Shangyue Zhu, Junhong Xu, Hanqing Guo, Qiwei Liu, Shaoen Wu, and Honggang Wang. Indoor human activity recognition based on ambient radar with signal processing and machine learning. In *2018 IEEE international conference on communications (ICC)*, pages 1–6. IEEE, 2018.

[221] W. Zhu and J. Pelecanos. Online speaker diarization using adapted i-vector transforms. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[222] Tehseen Zia and Usman Zahid. Long short-term memory recurrent neural network architectures for urdu acoustic modeling. *International Journal of Speech Technology*, 22(1):21–30, 2019.