Learning Normality is Enough: A Software-based Mitigation against Inaudible Voice Attacks

Xinfeng Li[1] Xiaoyu Ji[1†] Chen Yan[1] Chaohao Li[1,2] Yichen Li[3‡] Zhenning Zhang[4‡] Weyuan Xu[1†]
[1]*USSLAB, Zhejiang University,* [2]*Hangzhou Hikvision Digital Technology Co., Ltd.,*
[3]*Hong Kong university of Science and Technology,* [4]*University of Illinois at Urbana-Champaign*

Abstract

Inaudible voice attacks silently inject malicious voice commands into voice assistants to manipulate voice-controlled devices such as smart speakers. To alleviate such threats for both existing and future devices, this paper proposes `NormDetect`, a software-based mitigation that can be instantly applied to a wide range of devices without requiring any hardware modification. To overcome the challenge that the attack patterns vary between devices, we design a universal detection model that does not rely on audio features or samples derived from specific devices. Unlike existing studies' supervised learning approach, we adopt unsupervised learning inspired by anomaly detection. Though the patterns of inaudible voice attacks are diverse, we find that benign audios share similar patterns in the time-frequency domain. Therefore, we can detect the attacks (the anomaly) by learning the patterns of benign audios (the normality). `NormDetect` maps spectrum features to a low-dimensional space, performs similarity queries, and replaces them with the standard feature embeddings for spectrum reconstruction. This results in a more significant reconstruction error for attacks than normality. Evaluation based on the 383,320 test samples we collected from 24 smart devices shows an average AUC of 99.48% and EER of 2.23%, suggesting the effectiveness of `NormDetect` in detecting inaudible voice attacks.

1 Introduction

The prevalence of voice assistants magnifies the threat of inaudible voice attacks [1–5] that can secretly control smart devices without the user's authorization. For example, an attacker can send a physically inaudible voice command to a smart speaker and make it open the home door without being heard by the user. These attacks exploit the hardware vulnerabilities of microphones to convert the inaudible ultrasonic attack signals into malicious voice commands inside the device circuitry. To mitigate such a threat, existing studies have suggested several hardware-based [6, 7] and software-based [1, 2, 5] countermeasures. Though hardware redesign can fix the vulnerability for future devices, our talk with smartphone companies indicates that software mitigations that can instantly protect miscellaneous existing devices without any hardware modification are still in desperate need.

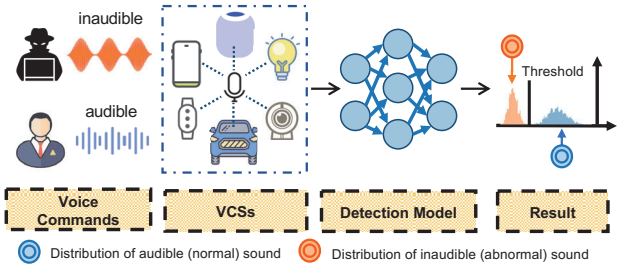


Figure 1: An illustration of `NormDetect`. Based on a universal detection model, it can distinguish inaudible voice attacks and benign audible audios on various voice-controlled systems (VCSs) that may exhibit different attack patterns.

However, a defense that is compatible with miscellaneous existing devices is non-trivial. First, existing devices, e.g., smartphones, wearables, and smart speakers, are designed with diverse microphone hardware, voice assistant software, and computational resources, which require the defense to be transferable and lightweight. Second, existing software countermeasures are supervised learning methods based on audio features of the attack signals. Collecting attack samples requires building dedicated attack equipment and is highly time-consuming. Last and most importantly, our experiment on attacking 24 devices reveals a previously unreported fact—*the inaudible voice attacks behave differently on various devices*, i.e., the characteristics of the injected voice vary from device to device for the same command, which is confirmed by audio spectral analysis. Our further investigation finds that such differences are caused by the inherent distinction of frequency response between the microphones in these devices. This observation essentially suggests that the audio features proposed by previous works for attack detection may

†Corresponding author.
‡This research is conducted when the authors were at Zhejiang University.
*Demo: https://sites.google.com/view/normdetect

not apply to every device. Customizing features, datasets, and models for each device to protect can be costly.

Our goal is to detect inaudible voice attacks with a universal and lightweight detection model that can be instantly implemented on various types of existing devices, as shown in Fig. 1. In the experiment of attacking 24 devices, we find that despite the patterns of inaudible voice attacks are diverse, benign audios share similar patterns in the time-frequency domain. This observation motivates us to detect the attacks (the anomaly) by learning the benign audios (the normality). Inspired by anomaly detection, where the abnormal samples are also generally sparse, difficult to obtain, and variable in pattern, we transform the detection of inaudible voice attacks into an *unsupervised* anomaly detection problem.

Based on this idea, we design `NormDetect`, a universal and lightweight software-based mitigation against inaudible voice attacks. The basic principle of `NormDetect` is to reconstruct the input audio signal with a variational autoencoder [8] and calculate the error between the reconstructed and original signals. We find that there exists a clear separation between the error distributions of benign audio and attack audios, making it easy to distinguish anomaly from normality. To increase transferability, `NormDetect` does not rely on attack samples or features derived from specific devices. The model is trained only based on an open-source speech dataset [9], and no attack samples are involved. To eliminate the impact of factors that increase the diversity of benign audios, such as speakers, speech content, etc., we combine the variational autoencoder with memory network [10], which can map the diverse and sparse benign audios into a compact latent embedding space. We also proposed audio preprocessing and spectrum augmentation to make the standard features more distinguishable from the attacks. To evaluate the performance, we implemented an inaudible voice attack dataset that contains attack audio samples collected from 24 mainstream smart devices and a testbed of 10 microphone modules with the setup of various attack distances, speech content, etc. Results show an average AUC of 99.48% and EER of 2.23%, indicating that `NormDetect` is effective and robust in various conditions. It is also lightweight (model parameters < 1.2M) and can be potentially deployed on various devices. The evaluation against adaptive adversary indicates the robustness and effectiveness of `NormDetect` as well. Our main contributions include:

- We are the first to identify and solve the challenges of defending inaudible voice attacks on miscellaneous existing devices. Our investigation explains the causes for the difference of attack audios on various devices.
- To the best of our knowledge, we built the largest inaudible voice attack dataset collected from 24 smartphones and 10 microphones. The dataset includes 28 speakers and 383,320 audio samples in English and Chinese.
- We design `NormDetect`, a universal and lightweight software-based mitigation that detects the attacks only by learning benign audios. Evaluation on our collected dataset shows its effectiveness and robustness.

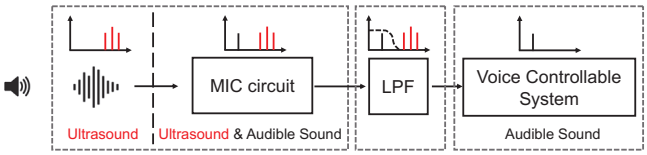


Figure 2: Inaudible voice commands transformation process. The sound can be inaudible by modulating audible voice commands on the ultrasonic wave (e.g., Amplitude Modulation). Due to the nonlinearity loophole of microphones, inaudible voice commands would be demodulated from the high-frequency carrier and then recognized by the speech recognition system.

2 Background

2.1 Inaudible Voice Attack

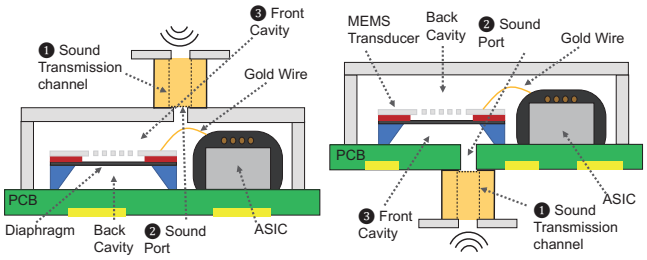
Inaudible voice attack exploits the nonlinearity effect of microphones to inject inaudible voice commands to state-of-the-art speech recognition systems [1, 2]. Fig. 2 presents the workflow of a typical inaudible voice attack. First, the malicious voice command (the baseband) is modulated on an ultrasound carrier (e.g., 25 kHz) via amplitude modulation (AM). Second, after the microphone receives the modulated ultrasound, the high-frequency input signal will be demodulated due to the nonlinearity effects of the microphone, and the baseband malicious command will appear at the microphone's output. A microphone's nonlinear transfer function is formulated as follows:

$$s_{out}(t) = \sum_{i=1}^{\infty} A_i s_{in}^i(t) = A_1 s_{in}(t) + A_2 s_{in}^2(t) + A_3 s_{in}^3(t) + \ldots \tag{1}$$

where $s_{in}(t)$ and $s_{out}(t)$ indicate the input and output of the microphone, respectively. The attacker leverages the nonlinearity loophole that microphones would inevitably recover the audible voice command from the amplitude modulated ultrasound. Third, the low-pass filter will remove the high-frequency ultrasound carrier and only leave the baseband command in the audio, which can be recognized by speech recognition and executed by voice assistants. Since the modulated ultrasounds are above 20 kHz, inaudible voice attacks are imperceptible to human users.

2.2 Microphone

With the proliferation of voice assistants, microphones have become essential for almost all smart personal devices. There are two main types of microphones, Micro-Electro-Mechanical Systems (MEMS) microphones and Electret Condenser Microphones (ECM). Notably, MEMS microphones dominate the market share due to their advantages such as miniature package sizes, low power consumption, robustness to mechanical vibration and temperature stability [11]. Fig. 3 presents the internal structure of two packages of MEMS



(a) open-top package MEMS (b) open-bottom package MEMS

Figure 3: The internal structure of two MEMS package types. (a) open-top & (b) open-bottom package MEMS microphone.

microphones: open-top and open-bottom. Specifically, the MEMS microphone leverages the MEMS diaphragm as a capacitor to transform sound waves into electrical signals. Similarly, ECM utilizes the capacity formed by a flexible electric material and a fixed pick-up plate to record sound waves.

Notably, the frequency response of a microphone is strongly correlated to its mechanical structure. For example, the frequency response of MEMS microphones is affected by three mechanical factors as tagged in Fig. 3: ❶ **radius of the sound transmission channel**, ❷ **size of the sound port** and ❸ **the cavity volume surrounded by MEMS mechanical cover** [12–14]. For example, the smaller the ❷ sound hole is, the higher the microphone's resonance frequency will be. Meanwhile, the open-bottom package microphone has a wider frequency response and a higher signal-to-noise ratio than the open-top microphone due to a smaller ❸ front cavity volume.

3 Attack Investigation

Before designing the defense, we conduct several experiments to reproduce the inaudible voice attacks and investigate their patterns on various target devices. In particular, we are concerned with three key research questions. **RQ1:** What are the differences between benign audio and inaudible voice attacks? **RQ2:** How different are the benign audio and inaudible voice attacks? **RQ3:** Why do the attack patterns vary on different target devices?

3.1 Preliminary Experiment

To answer **RQ1**, we recorded the audible benign voices and inaudible voice attacks on 3 commercial smartphones: iPhone 12, Huawei Mate40 pro, and OPPO Find X2. Specifically, the setup variation in this experiment includes distances (from 10 to 300 cm), attack powers (from 0.1 to 1 Watt), voice commands, and target devices. We summarize several remarkable observations in the following:

- The quality of the injected attack audio varies greatly with the target device. For example, the recorded attack commands sound clear on Find X2, relatively clear on iPhone 12, and slightly clear on Mate40 pro due to obvious noise.
- The quality of the recorded benign voices is clear and similar across all devices. There is only a negligible difference between their volume levels.
- We find that the attack power directly affects the audio quality. For example, the attack audio on Mate40 pro sounds like scream if the attack power exceeds a certain threshold.
- Speech commands have little impact on the attack effectiveness. For example, we used 5 TTS commands as the baseband, and the qualities of the injected attack audio are almost the same.

Insight 1: From the perspective of speech quality, the characteristics of audible voice commands are similar. In contrast, the inaudible ones vary according to the attack power, and most significantly, the target devices.

3.2 Quantitative Experiment

The phenomena of Sec. 3.1 inspire us to explore the experiment further with **RQ2**. We conducted an experiment to quantify the difference between normal and inaudible voice commands received by 24 smart devices. Due to space limits, we show the spectrums of 9 representative smartphones of different manufacturers and models in Fig. 4. Subfigures (a,b,c) demonstrate an audible command "OK Google" recorded by 3 smartphones. Subfigures (d-l) indicate the inaudible voice attack modulating the same command "OK Google" and recorded by 9 different smartphones. We observe that the spectrums of the 3 audible cases are quite similar. However, the patterns of (d-l) are different from each other. Notably, the command recorded by Find X2 sounds similar to the audible case, and its spectrum depicted in (l) is also similar to (a-c).

To precisely quantify the similarity between (a) to (l), we apply the DPAM [15], a deep perceptual audio metric. The results are shown in Fig. 5. The closer the score is to 1, the more similar the pair of spectrums is. We can observe that three audible commands (a-c) marked in the red dash box show high similarity up to 0.85, while the attack cases (d-l) are quite different from each other. Although the similarity between Find X2 and the normal voice commands reaches 0.64, there is still a significant gap compared to the aggregated degree within the normal ones.

Insight 2: Previous audio feature-based detection methods may not perform equally well on all devices due to the low similarity of attack patterns between different devices.

3.3 Root Cause Exploration

We are motivated to explore **RQ3**, the reasons why the attack patterns are different on various devices. According to Sec. 2.1, inaudible voice attacks leverage the nonlinearity of microphones. Thus, we assume microphones may be a dominant factor for the diverse attack patterns among other potential factors such as the operating systems and recording apps. We try to correlate the experimental phenomenon with

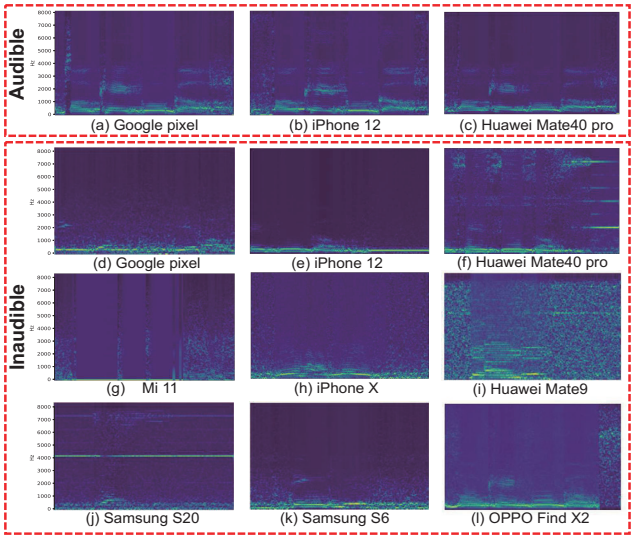


Figure 4: STFT spectrums of voice commands "OK Google". **(a-c).** The audible command "OK Google" is recorded by Google Pixel, iPhone 12, Huawei Mate40 pro; **(d-l).** The same command "OK Google" is modulated on the ultrasonic wave, which is imperceptible to human-being but can be recorded by smartphones.

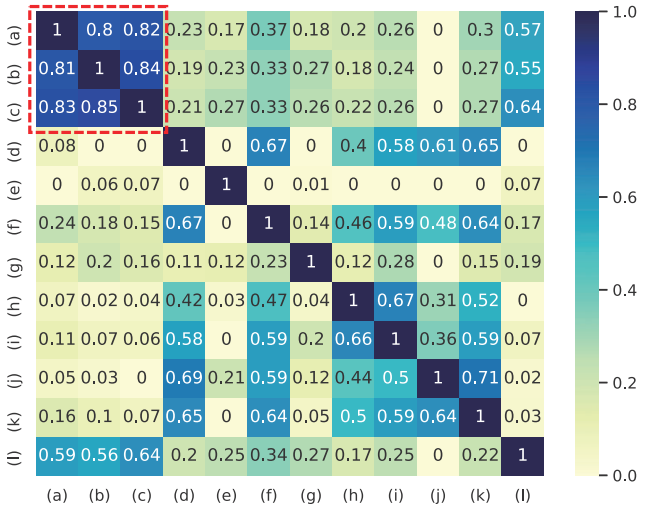


Figure 5: Similarity calculation of each two audio in Fig. 4, take pair in pairs and calculate their similarity, respectively. The red box represents the similarity among (a), (b), and (c), which are very similar, while each inaudible audio is distinct from the audible as well as other inaudible audio.

the microphone model of each smartphone. Unfortunately, we were not able to identify all the microphones' manufacturers and models through their silkscreen. As an alternative, we survey 10 commercial microphones with their datasheets and show the frequency responses of 6 microphones in Fig. 6. We find that they share a quite similar frequency response characteristic within the audible frequency bands (250 Hz to 6 kHz), because these frequency bands contain rich information of human speech from 300 to 3400 Hz [16]. It has also been designed as a standard bandwidth in PSTN (Public Switched Telephone Network) [17], and vendors would calibrate their MEMS chips to comply with this standard. The frequency response of different microphones in this band is calibrated to be almost identical, which explains why the spectrums of normal voices are so similar.

In contrast, in the bands over 10 kHz, the response of different microphones [18–23] is distinct. Notably, we find the microphone models of the OPPO Find X2 and Samsung S20 are SPH0644 (in purple) and SD18OB371 (in yellow), shown in Fig. 6, respectively. We can observe that the purple curve gets a relatively good response sensitivity at 6 dB over 20 kHz. However, there is a declining trend of the yellow curve in that frequency band. Note in Sec. 3.2, we have mentioned that such an attack has a good effect on Find X2 while bad on Samsung S20. Thus, we assume those with a good attack effect have an increasing microphone frequency response curve at ultrasonic frequency bands and vice versa.

To validate our assumption, we replace these two devices' microphones by carefully welding the S20's microphone on another Find X2, as shown in Fig. 7. After switching the microphones, we performed an inaudible voice command "Hi Cortana" to the S20, modified Find X2 (with the S20's microphone), and unaltered Find X2, respectively. Fig. 8 shows the respective 3 raw STFT spectrums. The modified Find X2 in (b) is highly similar to the Samsung S20 (a) while different from the unaltered Find X2 (c). This experiment supports our previous assumption.

Insight 3: The diverse frequency response of microphones in the upper 10 kHz band is the root cause for the different attack patterns on various devices. Normal voices are more similar to each other because the microphone's frequency response from 300 to 3400 Hz is close due to the manufacturers' calibration to comply with the PSTN standard.

To summarize, we are motivated by three research questions and derived insights, which directly or indirectly reveal why previous software-based methods may not be applicable to unseen devices. The key is to find a detection method independent of device variation while getting rid of the pre-training of specific anomalies.

4 System Design

4.1 Design Objectives and Challenges

We aim to design a holistic unsupervised learning algorithm that can detect inaudible voice attack for many smart devices only based on accessible normal audios. Specifically, our design needs to meet the following goals:

- **Lightweight.** Considering resource limitation in IoT devices is common, our algorithm needs to be lightweight and requires minimal computational overhead and storage resources.

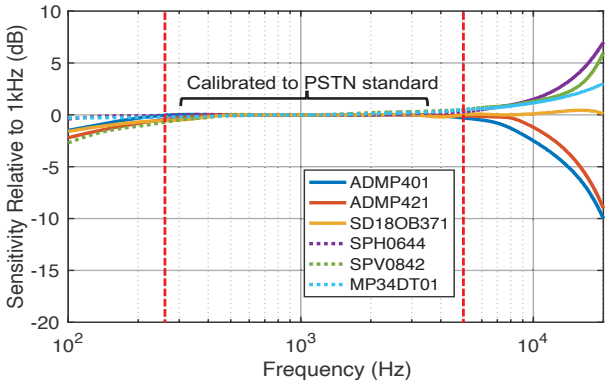


Figure 6: Frequency response of six different microphones.

Figure 7: The replacement experiment. **Left:** an original OPPO Find X2 with SPH0644 microphone; **Right:** a modified OPPO Find X2 replaced with Samsung S20's SD18OB371 microphone.

- **Universal.** The defense is expected to be instantly applied to a wide range of devices without requiring any hardware modification. In addition, it can be applicable to devices with only a single microphone (iWatch [24]), without relying on multi-microphones.
- **Unsupervised.** As we proposed an unsupervised system, our algorithm does not require inaudible data, so as to let researchers and manufacturers free from expensive equipment and costly data acquisition.

Challenges. To design such a holistic and robust system, we have to address the following challenges:

- In order to make the algorithm work for most devices, we need to consider the case where there is only one microphone. Such channel information is just not enough, making noise reduction difficult.
- For audible (normal) sound, normal audio patterns may also be very diverse due to the influence of the speaker, language content, age, etc.
- For inaudible (anomaly) sound, `NormDetect` cannot learn the patterns of abnormal samples. It needs to distinguish various anomaly samples recorded by distinct devices effectively.

4.2 System Overview

To meet the above goals as well as solve the challenges, we introduce the design of `NormDetect` in Fig. 9 to detect inaudible voice commands via learning the similar pattern of

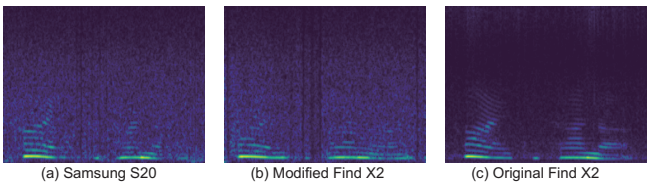


Figure 8: Raw STFT spectrums of the same inaudible voice command "Hi Cortana" recorded by three devices.

normal audio without any attack samples. The `NormDetect` consists of three key stages. 1) Speech Preprocessing: aims to transform raw audios into clean and normalized audios. 2) Spectrum Augmentation: aims to convert a preprocessed audio into an augmented spectrum by removing disturbing factors from normal audio. 3) `NormDetect` Model: in the inference stage, it maps the augmented spectrums into a latent space ($X \rightarrow Z$). Then it queries the memory module that only stores normal embeddings to make each input similar to normal patterns ($Z \rightarrow Z'$). Finally, it reconstructs the embedding to a spectrum ($Z' \rightarrow X'$).

4.3 Preprocessing

First, we need to eliminate some interference factors of voice commands, such as ambient noise and speaking speed, so that the subsequent spectral features can represent the core information.

Noise Removal. Because the ultrasonic wave is a high-frequency (over 20 kHz) mechanical wave, while common microphones are designed for telecommunication, which complies with the PSTN. They are ill-response to such a high-frequency attack. Specifically, their recorded sounds usually have varying degrees of distortion and noise, due to the abnormal response of their internal diaphragm and cavity. The ambient noise is somewhat similar to the abnormal microphone response caused by ultrasonic waves. Therefore, it is essential to remove environmental noise while retaining the abnormal pattern of microphone response caused by the inaudible attack.

We utilize a simple but effective method that makes the devices constantly "sense" their environments. This process is also named as "*Noise Level Perception*" in Fig. 9. We constructed a noise queue of the last five samples, averaged it to represent the ambient noise, and updated it with a timer thread so that the queue can reflect the real-time noise situation. When a voice is detected, we utilize spectral subtraction [25] for fast denoising it.

Silence Removal. It is vital to eliminate the influence of different unvoiced segments, which are caused by speakers' habits, such as speaking speed, and semantic pauses. For example, when we say "OK Google", the pause duration and speech speed are varying among young and old people. We want to eliminate pauses as much as possible to ensure that the model can focus on learning voiced features. Similarly, there is also a difference in the pause duration between the

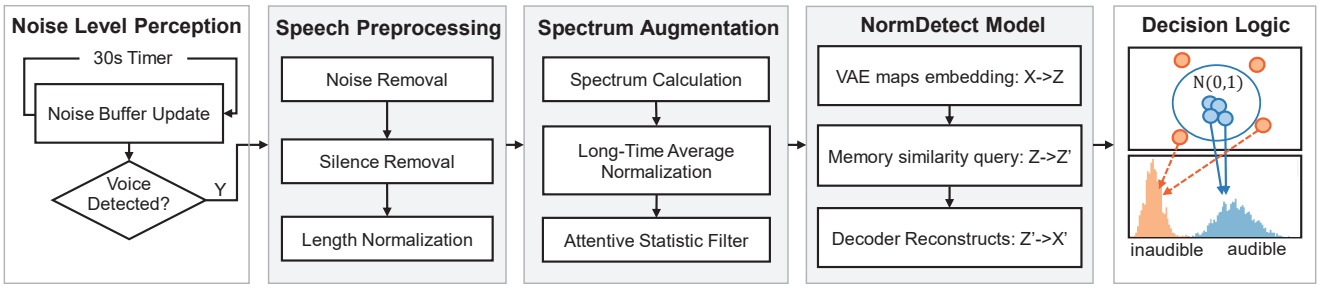


Figure 9: The diagram of NormDetect. The system will timely capture the ambient sound for noise level perception. While the voice activity is detected, it will go through speech preprocessing, spectrum augmentation, and NormDetect model inference (details are depicted in Fig. 11). Finally, we can judge it as attack or benign by its anomaly score.

same person saying "OK Google" and saying "Turn off the Light". Our method of removing silence is based on a normalized threshold. Firstly, we normalize an audio signal's amplitude to $[-1,1]$ via a min-max scale. Secondly, the max energy frame is noted to be 0 dB. Thus we regard frames with energy lower than a specific value as silence. Specifically, we conducted experiments based on thresholds ranging from -45 to -15 dB. Empirically, we found that the setting of -35 dB got the best overall performance.

Length Normalization. Since we apply convolutional / deconvolutional blocks for voice representation learning (encoder) as well as reconstruction (decoder), we need to carefully design a fixed input utterance length to represent various voice commands. In addition, we aim to design our system lightweight and quick-time. After counting the open-source Fluent Speech Commands dataset, we found that the average duration of typical voice commands within three words is around 1.9 seconds. If we remove the unvoiced segments of commands, the average duration is about 1.5 seconds, hence, improving real-time performance. We also apply the repeat pad method to fill for some voice commands whose duration is less than 1.5 seconds.

4.4 Spectrum Augmentation

After the above stages have preprocessed an audio, short and long-term interference factors still exist. Short-term factors are mainly caused by speech content, while long-time factors include speaker characteristics, microphone frequency response, etc. Note that in VCSs, we cannot avoid these challenging factors. In addition, since our system is unsupervised, spectrum augmentation is crucial for NormDetect to discern whether the sound is an attack and address the above interference. In this section, we explore methods to transform normalized audio into an augmented spectrum.

Spectrum Generation. We conducted preliminary feature selection experiments based on short-time Fourier transform (STFT), Fbank, and MFCC. The STFT outperforms others. Our setting parameters: sampling rate is 16 kHz; each frame contains 256 sample points; the number of sample points between adjacent frames named hop length is 64; applying Hanning window to each frame. Finally, we obtained an STFT spectrum of shape 128×376 according to a 1.5s voice command.

Long-Time Average Normalization. Although in the preprocessing stage, we mentioned eliminating interference factors such as different semantic pauses and speaking speed by removing the unvoiced segment. While it cannot address the problems caused by speech content and speakers mismatch. We use the long-time average normalization [26] method to suppress these factors. Specifically, the method is to average along the time axis of log STFT spectrums, therefore obtaining the long-time average spectrum (LTAS):

$$\mathrm{LTAS}_{\mathrm{x}}(k) = \frac{1}{L}\sum_{l=1}^{L}\log(|X(k,l)|) \tag{2}$$

where $X(k,l)$ is the spectrum of signal $x(n)$, k is the frequency index, l is the frame index, and L is the total number of frames.

Attentive Statistical Filter. The *Fratio* [27] was first used to improve the performance of speaker recognition, and it has shown effectiveness in emphasizing individual information (inter-speaker) and restraining linguistic information (intra-speaker). For detecting inaudible voice attacks, our goal is to enhance high discriminative information between audible and inaudible classes. Therefore, suppressing speakers' individual and linguistic information is helpful for this goal.

Specifically, *Fratio* is effective in our scenario because it performs statistical calculations on various LTASs of normal audio in the training stage. We can obtain the significance of each sub-band of the normal voice commands. The significance is exactly a weight coefficient vector of 128-dimension related to the shape of LTAS. It can be regarded as a filter in the test stage, because we apply it to weight different sub-bands of each input LTAS, making some sub-bands significant while masking others. We name it the "Attentive Statistic Filter" out of its idea of attention and computing statistical information of normal audio. We also leverage the previous knowledge [2] to improve this filter, which indicates the energy inaudible voice attack is concentrated in low-frequency bands. Empirically, we reduce the weights of the attentive statistic filter in sub-100 Hz. We depicted Fig. 10 to understand the effect of such a filter better. The differences between two audible and inaudible voice command spectrums have

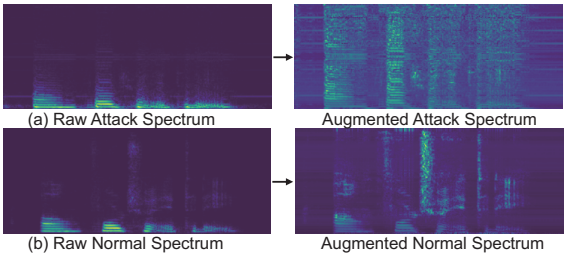


Figure 10: The raw attack and normal voice command spectrums of "Unfortunately" augmented by long-time average normalization and attentive statistical filter.

been obviously enlarged.

4.5 `NormDetect` Model

As shown in Fig. 11, after preprocessing and spectrum augmentation stages, our model takes the augmented spectrum as input to map it, reconstruct it, and judge it based on the anomaly score. While the same content and audible audio of speakers captured by various devices are very similar, different speech content or speaker factors are challenging. We design to effectively eliminate distracting factors from the neural network perspective to achieve effective learning from the normal audio. Autoencoder (AE), a vital branch in anomaly detection, inspires our approach because it does not depend on the label information of data. It is generally assumed that it reconstructs the normal well based on standard training data while reconstructs poor for abnormal data. Due to mismatch of model parameters only apply to standard samples. However, this assumption does not always hold.

VAE maps embedding $X \rightarrow Z$. Variational Autoencoder (VAE) is a well-designed variant version of AE, in Fig. 11 **Stage 1**. It is based on bayesian variational inference to describe data with a multivariate Gaussian distribution $(\mu + \delta)$ in latent space, shown in Equ. 3, where $\theta_e, \theta_\mu, \theta_\delta$ indicates network parameters of encoder components, respectively. In the training stage, we use VAE to successfully reconstructs the spectrum X of audible voice commands, combining with the KL divergence loss to make the encoder constrain the embedding of audible voice commands approximating a standard Gaussian distribution. Notably, these two goals of successful reconstruction and constraint embedding are contradictory to each other, with the idea of adversarial training. It maps the input spectrum X into a latent embedding Z following Gaussian distribution. As Equ. 4 indicates, we drive VAE to simulate various normal audio being disturbed by noise, therefore, making the modelling of normal audio patterns more robust. At the inference stage, compared with the random distributed hidden vectors obtained by AE, the distribution $P(Z)$ of the VAE encoder's output is a squeezed multivariate Gaussian distribution. Each normal embedding Z_i is very close to the other. In contrast, the embeddings of inaudible voice commands are relatively cast out of the distribution.

$$\begin{aligned} \mu &= F_e(X; \theta_e + \theta_\mu) \\ \delta &= F_e(X; \theta_e + \theta_\delta) \end{aligned} \tag{3}$$

$$Z = \mu + \sigma \times \delta \tag{4}$$

Memory Similarity Query $Z \rightarrow Z'$. We would like to constrain the generalization capability of VAE to keep the inaudible voice commands from being falsely accepted by VCSs, especially for those similar to normal audios. How to make our model capable of memorizing the normal patterns? We introduced the memory network [10], which is equivalent to using a global memory that can be read and written to perform the memory task better than the classical method. As the Fig. 11 **Stage 2** indicates, our memory module contains N items recording various prototypical patterns of audible voice commands. The mapped embedding Z from the previous steps calculates similarity with each memory item M_i. Therefore, a memory similarity weights vector w is acquired in Equ. 5, which is defined more specifically in Appendix. B. After removing low similarity items in w, it transformed into w'. Furthermore, according to Equ. 6, it acquires Z', which indicates replacing the original Z with queried Z'. This process is critical for detecting inaudible samples because the anomaly data falls outside the normal distribution. Subsequently, the memory module is used to force to replace the anomalous embedding with the pattern of the normal training data so that the difference between the Z' reconstructed X' and X is further amplified.

$$w = ZM = [Zm_1, Zm_2, ..., Zm_n] \tag{5}$$

$$Z' = w'M = \sum_{i=1}^{N} \omega_i' m_i \tag{6}$$

Decoder Reconstructs $Z' \rightarrow X'$. After obtaining the memory queried embedding Z', we reconstruct $Z' \rightarrow X'$ using the decoder, referring to the style of DCGAN's generator that leverages some topological constraints. It is beneficial to decode more stably even trained with an unsupervised framework. We also evaluated MSE and CE as loss functions. Although both are derived from the maximum likelihood theory, CE outperforms MSE in our task. In the inference stage, the decoder in **Stage 3** will reconstruct the hidden vector Z' to intact spectrum X'. Therefore, the anomaly score can be obtained by computing the negative log-likelihood between X' and X. Details are in Appendix. D.

5 Implementation

This section introduces our self-made large-scale voice command corpus, which includes both audible and inaudible audio. To our best knowledge, this is the first large-scale inaudible voice commands speech corpus based on various

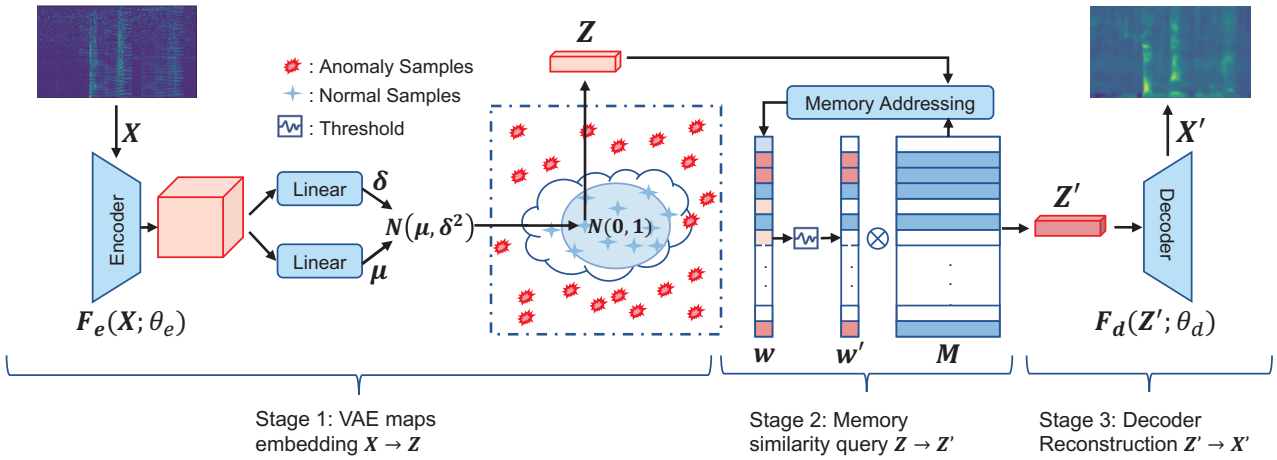


Figure 11: Diagram of the proposed NormDetect model. **Stage 1:** The VAE maps the input spectrum X to tensor and transformed it into μ, δ by two linear full-connected layer. Then it resamples a vector Z to represent X, from the normal distribution $N(\mu, \delta)$. **Stage 2:** The memory module items record normal spectrum patterns, and Z calculates similarity with each memory slot, respectively, named Memory Addressing. After softmax, the similarity weights sum up to 1. Then the threshold remove those slots with low similarity, leaving high similarity weights to multiply with the corresponding memory items, getting memory queried Z'. **Stage 3:** Decoder reconstructs Z' into X', hence acquiring anomaly score between original X and generated X'.

Table 1: The details of two datasets in our experiment

	Speakers	Data Source	Number of Utterance	Language	Average Duration
Training Set	97	Fluent Speech Commands Dataset[9]	30,043	English	3.16s
Test Set	28	Audible & Inaudible Voice Commands Corpus	383,320	English & Chinese	1.31s

smart devices, collected with our experiment setup shown in Fig. 12. We also designed an integrated microphone testbed for the subsequent microphone-only evaluation.

5.1 Audible & Inaudible Voice Commands Corpus

Motivated by the following reasons, we design and collect a large-scale audible & inaudible voice commands corpus: (a) **For manufacturers**, establishing a large dataset of inaudible voice commands is over-cost, e.g., it requires extra hardware and professional personnel; (b) **For researchers**, there is no available inaudible voice command dataset, setting a barrier to conducting defensive research on inaudible voice command attacks. (c) **For validation**, we can evaluate the robustness and transferability of NormDetect on such a large-scale dataset.

The overview of our dataset presents in Table. 1. The entire self-made dataset comprises around 383,320 speech samples, including both audible and inaudible instances. Moreover, it involves 28 speakers, including human speakers and virtual speakers, and 33 different speech contents. The detailed information of our self-made dataset is listed as below:

- **Speaker:** A group of 26 human individuals consisting of 22 males and four females participated in our data collection, aged 22 to 29. Besides, we added two virtual speakers: a male voice and a female voice generated by Baidu Text-to-Speech API [28], respectively.
- **Content:** The speech content can be divided into four types: English wake-up words, Chinese wake-up words, common phoneme phrases, and typical speech commands. Participants are asked to say each wake-up word, phrase, or command three times. The details of 33 speech contents are given in Appendix. A.
- **Recording Devices:** Firstly, we collect the audible voices with four smartphones (Google Pixel, OPPO Find X2, Huawei Mate 40 Pro, and iPhone 12) recording simultaneously. Secondly, we employ 24 mainstream smart devices to record the modulated inaudible voice commands, as shown in Tab. 3.
- **Recording Setup:** We select the audible commands of iPhone 12 at 30 cm as baseband for ultrasound modulation and launch the inaudible voice attacks at seven distances: 10, 30, 60, 100, 150, 200, and 300 cm, respectively. To collect high-quality audio samples, the attack angle and ambient noise are set to within -30-30 ° and 40 45 dB respectively. The ultrasound carrier frequency to 25 kHz, and the ultrasonic speaker power is up to 1 Watt.
- **Recording Samples:** With the above settings, our smart devices testbed acquires 54,600 audible samples, 327,600 inaudible samples. The microphone testbed also acquires 560 audible samples as well as 560 inaudible samples.

[§]We followed the Institutional Review Board (IRB) regulations to protect the rights of human participants.

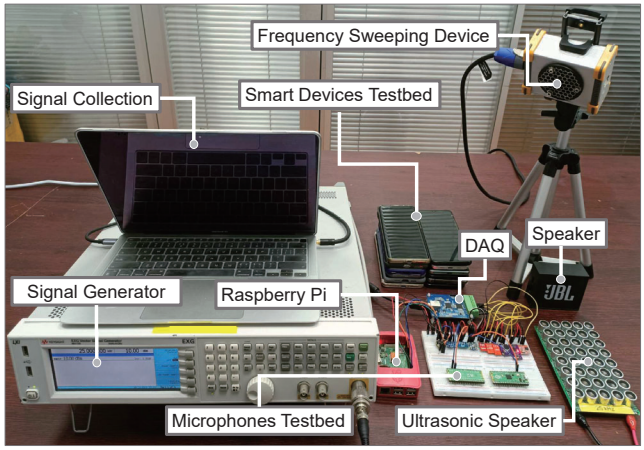


Figure 12: Signal Generator modulates audible voice commands on 25 kHz carrier signal; The whole smart devices testbed are composed of 24 commercial devices, which was used to collect the large inaudible voice commands dataset. In addition, verifying our system performance.

Effectiveness of the Corpus. We validate the effectiveness of our audio & inaudible voice commands corpus in terms of both recognition accuracy of the automatic speech recognition (ASR) systems and human speech intelligibility. To precisely evaluate the corpus, we utilize four local ASR models based on SpeechBrain [29] and three commercial ASR models [30–32]. We obtain the best ASR failure rate down to 10.7% on Microsoft Azure [32], where the failure rate means that an audio cannot be recognized for any word and the failure is due to following reasons, such as the speaker's accent and low SNR (signal-noise-rate) audios on several devices, e.g., Samsung S20. We also paid 5 crowdsourced workers to review 10,000 randomly selected audio samples, and they reported that 92.5% of the audio could be accurately recognized.

5.2 Integrated Microphones Testbed

Recall from Sec. 2.2, the microphone is the main reason for the difference in inaudible speech attack performance on various devices. To uncover the correlation between the frequency response of the microphone and the performance of inaudible speech attacks, we built an integrated microphones testbed. We investigate ten mainstream microphones, as shown in Fig. 12. It contains two I2S outputs, two PDM (Pulse Density Modulation) outputs, five analog outputs from MEMS microphones, and an ECM. Notably, at the data acquisition stage, analog microphones can easily access real-time voice data via DAQ. In contrast, the I2S and PDM microphones require external embedded systems to decode their digital coding signals. Thus, we implement the software simulated I/O drive for I2S and PDM with Raspberry Pi and Raspberry Pico, respectively, to acquire digital microphone data.

Table 2: The overall performance of `NormDetect`

Evaluation Setting.	AUC.	EER.	FRR.	FAR.
All Data Mixed	99.11%	3.81%	2.0%	5.03%
All Data Averaged	99.48%	2.23%	1.54%	2.57%

6 Evaluation against Inaudible Voice Attacks

In this section, we report how we obtained the `NormDetect` model and evaluate its performance of discriminating inaudible voice attacks by our collected corpus. In addition, we report the overall performance and also verify the robustness of `NormDetect` across multiple factors.

Experimental Setup. Our experimental settings, such as speaker, content, devices, and environment, etc., are precisely described in Sec. 5. In the training stage, we obtain the `NormDetect` model only based on Fluent Speech Commands in Table. 1. Note that no data from the devices in Tab. 3 participates in training.

Metrics. We use the following metrics throughout the evaluation. False Rejection Rate (FRR) characterizes the rate at which the system falsely rejects audible commands. False Acceptance Rate (FAR) characterizes the rate at which an inaudible voice command is wrongly accepted by the system and considered normal. AUC is widely applied for anomaly detection due to its insensitivity to data imbalance. A higher AUC score indicates that the system has more discriminative power against inaudible voice attacks. Besides, EER, Precision/Recall, and Accuracy are also used to represent the performance of our method.

6.1 Overall Performance

We utilize the same trained `NormDetect` model mentioned in the experimental setup to test all the abnormal samples recorded by 24 devices and the normal samples recorded by four smartphones. We summarize the overall performance of `NormDetect` in Tab. 2 and discuss it in the following.

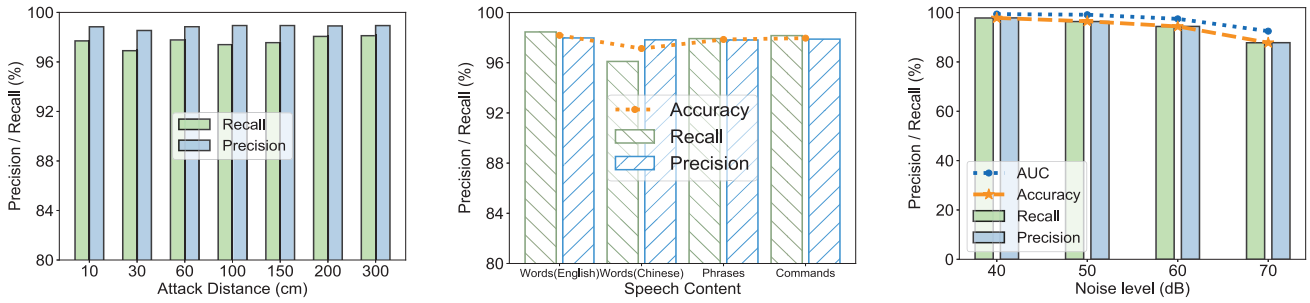
We derive an overall AUC of 99.11% and an EER of 3.81%, demonstrating that our model has a solid discriminatory ability under a large amount of data. Since in the practical scenario, users have no previous knowledge of the abnormal patterns of their devices. We set the detection threshold based on the FRR of the normal data. Empirically, we set 2% here for better usability and get a log-likelihood threshold corresponding to a FAR of 5.03%. We name this case as **All Data Mixed**, which indicates we utilize a unified threshold for all evaluated data.

In addition, the inaudible voice attack audio recorded by different devices appears to be diverse and has disorderly patterns. Therefore, we apply the model to evaluate the performance of each device. Specifically, each case includes $7(D) \times 25(C) \times 3(T) \times 26(P) = 13650$ normal samples and attack samples. Notably, for these 24 devices, we keep their

[¶]D: Distance; C: Speech Content; T: times; P: Participants

Table 3: Performances on 24 smart devices

Manufacture.	Model.	OS/Ver.	Performance.		Model.	OS/Ver.	Performance.	
			AUC.	EER.			AUC.	EER.
Google	Pixel	Andriod 10	99.44%	2.12%	Nexus 5	Android 6.0	99.12%	3.55%
Samsung	S6	Android Nougat	98.95%	4.03%	S20	Android 10	99.92%	1.15%
Xiaomi	Mix 2	MIUI 11.0.2	99.98%	0.20%	Mi 5	MIUI 10.0	99.25%	2.92%
	Redmi K30	MIUI 12.0.18	99.92%	0.46%	Mi 11	MIUI 12.5.4	99.18%	3.81%
Huawei	Nova 2	EMUI 8.0.0	99.52%	1.65%	P10	EMUI 5.1	99.41%	2.84%
	Mate 9	EMUI 9.0.1	99.61%	1.48%	Mate 40 pro	HarmonyOS 2.0.0	99.29%	3.55%
	MatePad	HarmonyOS 2.0.0	99.57%	1.92%	iWatch SE	WatchOS 7.0.1	98.76%	4.28%
Apple	iPhone 6	iOS 11.3	99.95%	0.82%	iPhone 7	iOS 14.0.1	99.84%	0.66%
	iPhone X	iOS 14.1	**99.99%**	**0.19%**	iPhone 12	iOS 15.0	99.38%	2.43%
OPPO	K3	ColorOS 11	99.81%	1.42%	Reno5 pro	ColorOS 11.1	99.20%	3.77%
	Reno3 5G	ColorOS 7.1	99.89%	1.55%	Find X2	ColorOS 11	**98.52%**	**4.59%**
Seeed	ReSpeaker	/	99.70%	1.07%	Find X5	ColorOS 11	99.30%	3.01%



(a) Impact of Distance (b) Impact of Content (c) Impact of Noise

Figure 13: (a) The performance of NormDetect on seven distances; (b) The impact of four types of content, and words is an abbreviation of wake-up words; (c) The impact of noise, ranging from quite living room around 40 dB to noisy street over 70 dB.

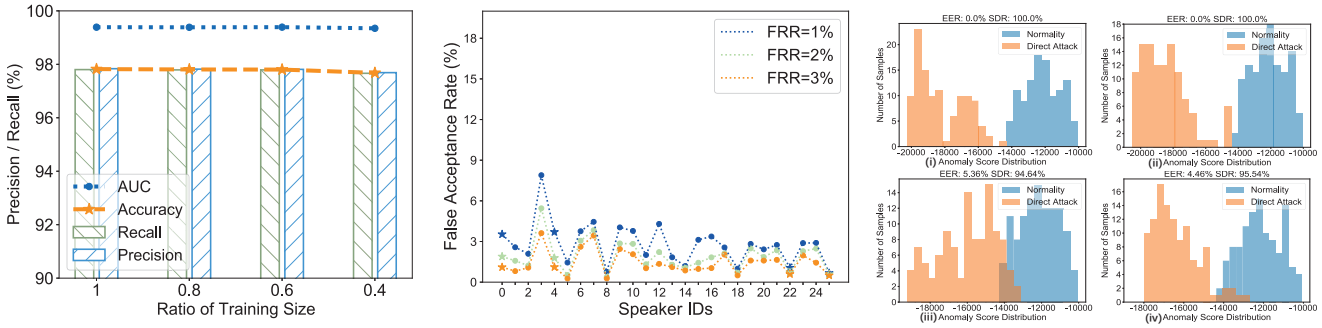
normal samples the same. We also obtain the Tab. 3, which indicates the model also has good performance on each device. The minimum EER is down to 0.19%, and the best AUC is 99.99%. In comparison, it performs relatively poorly on OPPO Find X2, the most easily attacked device among 24 devices. Finally, we average out the results to obtain the overall performance based on the whole collection of data. We get an overall AUC of 99.48%, an EER of 2.23%, a FAR of 1.54%, and an FRR of 2.57%. We name this case as **All Data Averaged**, which indicates we set a threshold for each device accroding to their EER respectively; then we average their performance.

The results reflect that we can set an appropriate threshold according to normal samples for the specific device. To balance usability and security, we prescribe the threshold value based on $FRR = 2.0\%$ in the remaining experiments, if not stated otherwise.

6.2 Impact Factors on Performance

Impact of Distance. We evaluate the detection performance at seven distances with 20 commands and 15 devices, shown in Fig. 13(b). As the distance increases, we increase the SPL of the modulated signal accordingly so that the ultrasonic wave keeps a good attack effect when reaching the victim devices. We can observe that there is no significant difference among varying distances, with an average precision of 98.14% and recall of 97.64%, respectively. Nevertheless, it is worth noting that the 30cm case seems slightly complicated for NormDetect to distinguish inaudible voice commands from normal counterparts. The ultrasonic beam focuses on the sound input hole at that distance, getting a better SNR. By contrast, the ultrasonic beam of the 10cm attack is compelling, which causes some phenomena, e.g., harsh screams or high-frequency noise, making the 10cm pattern relatively more distinguishable from normal patterns.

Impact of Speech Content. We divide all speech content into four groups, which are recorded by ten devices: English wake-up words, Chinese wake-up words, common phoneme phrases and typical speech commands in Appendix A. Fig. 13(c) indicates that the precision/recall of the English wake-up words is more stable than the Chinese, which is probably due to the language mismatch between the training dataset (entirely in English) and the testing Chinese wake-up words. We can also conclude that NormDetect



(a) Impact of Training Size Ratio (b) Impact of Speaker (c) SDR of Inaudible Voice Attacks (baseline)

Figure 14: (a) The impact of training size ratio, ranging from 0.4 to 1.0 of the Fluent Speech Commands dataset; (b) The impact of speakers on the false acceptance rate of varying threshold, which is decided by varying FRRs; (c) SDR of inaudible voice attack on representative devices, as a reference with adaptive adversaries. (i): Mix2; (ii): iPhone X; (iii): Find X2; (iv): iWatch.

performs better on the phrases and longer commands, which contain richer phonemes and information, with the average precision/recall up to 98.31 and 98.52%, respectively.

Impact of Noise. In this section, we focus on exploring the influence of ambient noise. As mentioned in Sec. 5.1, we collected our inaudible voice commands dataset in a quiet environment (40 dB), such as the living room. Furthermore, we set another three scenes, such as the office (50 dB), café (60 dB), and the street (over 70 dB), respectively. We perform the noise removal described in Sec. 4.3 to make the data recorded in noisy environments as clean as possible because the embedding distribution of clean data is similar to that of the training data. It can be seen that as the noise level increases, the detection success rate of the algorithm drops. Nevertheless, the success rates of around 40dB and 50dB environments are still considerable, with the ACC of 97.56% and 96.47%. As for the over 70dB cases, the performance drops to an ACC of 87.79% due to excessive noise, while the attack itself is too challenging to perform the successful injection [1].

Impact of Training Size Ratio. To evaluate the dependence of our method on the amount of training data, we divide the whole open-source dataset with 30,043 pieces of audio into 4 levels, 100%, 80%, 60%, 40% training samples. Then we perform experiments on the inaudible voice command corpus, including 25 types of content and 18 devices. As Fig. 14 (b) depicts, we can conclude that the best performance 97.9% (with 100% training data) to the worst 97.5% (with 40% training data) is very close. In this way, we could infer that `NormDetect` can learn well based on a small amount of normal data, making it easy to deploy in new scenarios.

Impact of Speaker. It is conceivable that the specific speaking patterns of different speakers, gender, and pitch significantly impact the audio characteristics. Although the inaudible voice commands modulation process will distort the speaker's characteristics, the speaker-related patterns still remain. We perform with 26 speakers' data, including 20 types of content from ten devices. As depicted in Fig. 13 (c), stars represent females (p0, p4, p22, and p25), and dots represent the male. We find that `NormDetect` is robust to speaker factors such as gender, age, and speech speed. By setting FRR from 3 to 1%, we can observe that the FAR of each speaker slightly increases, while the overall FAR is still good at 1.42%, 1.93%, and 2.80%, respectively.

Table 4: The overall performance against adaptive attacks

Attack Type	(Baseline) Inaudible Voice Attacks	(1) Noise Injection Attacks	(2) Noise Removal Attacks	(3) Adversarial Examples Attacks
Average SDR	97.55%	95.76%	98.44%	92.64%

7 Evaluation against Adaptive Attacks

While our defense framework can accurately detect anomalies from benign examples for inaudible voice attacks, it only offers security in a "zero-knowledge" attack scenario where the attacker is unaware of the defense. We envision an adaptive adversary at two-level: (a) **Gray-box**: partial knowledge of our defense, i.e., the attacker knows our basic idea of learning from normality. (b) **White-box**: the attacker has full knowledge of `NormDetect`'s defense methods and parameters. With gray-box knowledge, the attack may **(1)** inject common noises to the attack audio to pretend as noisy benign samples; or **(2)** remove unique noises in the attack to improve the audio quality. With white-box knowledge, the attacker may **(3)** create adversarial examples against `NormDetect` using end-to-end optimization.

Experimental Setup. To validate the effectiveness of adaptive adversaries, we chose 4 representative devices, on which our system has the best and the worst detection performance, and performed non-adaptive inaudible voice attacks on them as the baseline.

Metrics. To evaluate our performance on detecting adaptive adversaries, we employ SDR (successful defense rate) as the metric, which characterizes the rate at which the system

successfully rejects inaudible voice attacks.

Baseline. We derive the NormDetect's performance against attacks on above devices with SDRs for 100% (Mix2), 100% (iPhone X), 94.64% (Find X2), and 95.54% (iWatch), shown in Fig. 14 (c). We regard these results as a baseline to better evaluate the adaptive adversary, and summarize the results in Tab. 4, indicating NormDetect is robust against 3 proposed adaptive attacks.

Injecting Common Noises to Pretend as Benign. To deceive NormDetect, a gray-box attacker may inject common noises in benign human voices to make the attack samples appear similar to benign samples. We carefully select four typical noises in daily scenarios from FreeSound: people talking (café), keyboard tapping (office), TV show (living room), and sound of fans (bedroom). We obtain the mixed attack audios on victim devices by conducting inaudible voice attacks while a JBL plays noises. For these specially crafted attack audios, Fig. 15 (a) illustrates the SDR for noise-injection attacks with an average SDR of 95.76%. Although noise-injection attacks slightly improve the probability of fooling our system, NormDetect still holds discrimination ability against the attack.

Improving the Attack Audio's Quality. In our experiments, we find that inaudible voice attacks by themselves can introduce unique noises that degrade the audio quality. Therefore, an adaptive attacker may try to remove such unique noises and improve the audio quality. Thus, we term it the noise-removal attack. We investigate the feasibility of this attack using software-based signal compensation, as sophisticated hardware devices (signal generator, ultrasonic speakers) have been employed. We implement a neural network that models the transformation process of inaudible voice attacks in Fig. 16, then utilize the model to infer a desired baseband that would result in a high-quality attack audio. Fig. 15 (b) shows NormDetect's good detection capability on the improved attack, with an average SDR up to 98.44%, even slightly higher than the baseline. More details are described in Appendix. F.

Creating Adversarial Examples. An adaptive attacker with full knowledge of NormDetect may try to spoof the ML-based defense by creating end-to-end adversarial examples. We implement a white-box adversarial example attack against NormDetect based on PGD optimization. The results in Fig. 15 (c) show an average SDR of 92.64%, indicating our system still defends the end-to-end adversarial examples well. We also feed the optimized attack audios to commercial ASR APIs for recognition and find that ASR systems cannot recognize attack audios well. The Character Error Rate is 88.96%, suggesting that the attack may fail to spoof the voice assistant even if it can bypass our defense. More details are described in Appendix. F.

8 Discussion

Runtime Overhead of NormDetect. We split and deploy the NormDetect system on smartphones (device-side) and a GPU-based server (server-side) respectively, recalling the workflow in Fig. 9, where the device-side runs "Noise Level Perception". Meanwhile, the server-side performs the remaining computation-intensive tasks, i.e., "Speech Preprocessing, Spectrum Augmentation, and Model Inference". Therefore, the runtime overhead of our system is mainly divided into the energy overhead brought by the long-term operation of the device-side APP as well as the latency overhead brought by processing the voice command, data transferring between client and server, server-side processing & inference, and result feedback.

Energy Overhead. The power consumption of NormDetect APP varies with its time interval for "Noise Level Perception". We derive the average power consumption on various smartphones at 5s, 30s, and 300s time intervals as 0.27, 0.13, and 0.10 mAh/m (mAh per minute), respectively. We finally set 30s as the period because it balances real-time and energy efficiency. For a direct comparison, we also count the power consumption of running navigation software such as Apple Maps and Google Maps, which are between 1.54 and 1.93 mAh/m.

Latency Overhead. We obtain an overall latency of NormDetect about 278.5ms, of which the average denoising delay on device-sides, audio uploading delay, server-side processing& inference delay, and result feedback delay are 14.5, 81.2, 177.4, and 5.4 ms, respectively. NormDetect can meet the latency requirements of commercial ASRs (480ms [33]).

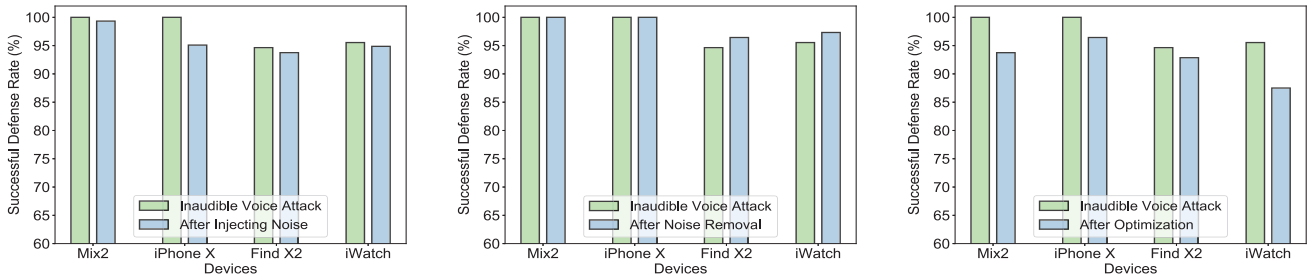
Potential Optimization. We envision that both overheads can be further optimized. Energy overhead would decrease with deploying the device-side tasks on low-power co-processors, similar to the voice assistants, e.g., Siri [34]. For latency overhead, performance can be improved with more powerful servers and real-time file transfer protocols such as webRTC.

Comparison with Existing Defenses. We investigate prior works against the inaudible voice attacks (focus on air-mediated propagation). With reference to Tab. IV in [35], we summarize them in terms of three representative characteristics shown in Tab. 5. We discuss existing works and ours from two perspectives: qualitative and quantitative analysis.

1) Qualitative Analysis:

• *Hardware Independent:* Our system is applicable to any device with one or more microphones, e.g., smartphones and smart speakers, while [6, 7, 36] are based on prototype hardware or require multiple microphones.

• *Attack Samples Independent:* Our system requires no attack samples for model training, which enables it to be applied to a wide variety of unseen devices that are impossible to enumerate in training. Most existing work [1, 2, 7, 36] adopt supervised learning and require both benign and attack samples for training. As we have mentioned in the paper, the patterns of inaudible voice attacks are diverse on different



(a) SDR for Noise-injection Attacks (b) SDR for Signal-compensation Attacks (c) SDR for Optimized attacks

Figure 15: SDR results for adaptive adversaries by adopting different methods vs. the baseline.

devices, and collecting a sufficient amount of attack samples can be very costly.

• *Evaluation Scale:* The related defenses were evaluated on 13,000 samples and 3 devices at most. In comparison, we have evaluated `NormDetect` on a much larger scale based on 383,320 samples (29.5 times more) collected from 24 devices (8 times more). We believe a large-scale evaluation can provide more information for the robustness of a defense in practice.

2) Quantitative Analysis: For a quantitative comparison with state-of-the-art defenses, we reproduced two representative works, LipRead (software-based) and EarArray (hardware-based), strictly following the instructions in their papers. The results show that our approach is relatively more robust.

• *LipRead*: To train the LipRead's classifier, we used 30,042 benign samples from the Fluent Speech Commands and randomly selected 30,000 attack samples from our collected corpus. Then we tested LipRead and `NormDetect` on the rest benign and attack samples which were not involved in LipRead's training stage. The accuracy of LipRead is 88.86%. In comparison, that of `NormDetect` is 97.41%. We consider the deviation of LipRead from the initial results is due to a larger test scale.

• *EarArray*: To compare with EarArray properly, we implemented its feature extraction algorithms and conducted experiments on smartphones that support recording stereo audios. Since audios in our corpus are all mono-channel, we recollected stereo benign and attack samples for this experiment, including single- and multi-injection settings shown in Fig. 19. We trained EarArray's classifiers with 20% samples. Results show that EarArray can defend against single-injection attacks well with 96.83% accuracy, but performs poorly in multi-injection scenarios with an accuracy of 78.55%. In comparison, `NormDetect` still defends well in both scenarios, maintaining 97.04% and 96.55% accuracy, respectively.

9 Related Work

The research community has driven rapid development in both attack and defense of speech security. Audible voice attacks such as replay, speech synthesis, and voice conversion are sometimes limited. Due to their obviousness, the victim could easily perceive and locate such attacks. Therefore, inaudible voice attacks have attracted the interest of many researchers thanks to their concealment.

Table 5: Comparison with exsiting defenses

Name	Hardware Indep.	Attack Samp. Indep.	Evaluation Scale	
			Samples[1]	Device
Zhang et al. [1]	✓	✗	∼24	1 device
LipRead [2]	✓	✗	∼2,368	3 devices
GuardSignal [6]	✗	✓	/	1 prototype
EarArray [7]	✗	✗	∼13,000	1 prototype mic. array
Li et al. [36]	✗	✗	1,560	3 mic. arrays
`NormDetect` (this work)	✓	✓	383,320	24 devices

[1] Samples: audio clips containing 1-3 seconds of voice commands

Inaudible Attack on VCSs. Most inaudible voice attacks leverage the non-linearity loophole of the microphone, which was revealed in [37]. [1, 3, 38] introduce injecting commands silently to various IoT devices, making such an attack extremely harmful, and [2] enhanced its attack range up to 25 ft. Yan et al. [5] further implement inaudible voice transmission in solid materials. Recent work [4] enables modulation of ultrasound with capacity in smart devices and control voice assistants. Apart from sound-based attacks, there are also works injecting voice commands into VCSs via EMI [39, 40] and light [41]. To defend against inaudible voice attacks, [1, 2, 5] proposed feature-extraction methods. These works convert audio into classical statistical features as well as three-dimensional features based on non-linearity, and monitor high-frequency range, respectively. As we have explained in Fig. 6 of Sec. 3, the patterns revealed by these works are not applicable for various devices because different microphones hold distinct frequency response characteristics. In contrast, we found the similarity among audible sounds and built an efficient detection model without requiring any attack samples. To explore the method that works for all devices, [7] utilizes the prorogation characteristic of sound in the air. Nevertheless, it requires hardware modification and at least three microphones, which sacrifices usability and could not apply

to many devices equipped with a limited number of microphones. In this work, NormDetect is designed to process with only one microphone channel, indicating its availability to various VCSs. Another hardware defense strategy is [6], which designs a "guard" signal transmitter to eliminate the attack signal. However, it requires the peripheral signal transmitter, making it impractical for current compact smart devices. By comparison, we design NormDetect to be a lightweight software-based strategy with model parameters less than 1.2M to meet the various resource-limited IoT devices.

Anomaly Detection in Sound Domain. Anomaly sound detection (ASD) methods mainly address scenarios such as audio surveillance [42, 43] and machine condition monitoring [44]. Because of the absence of anomaly data, mainstream works could be grouped into two categories. The basic idea of unsupervised ASD methods is close to our proposed methods. [45] considers combining a statistical hypothesis objective function with autoencoder structure and works well in inspecting devices such as 3D-printer, air blower pumps, and water pumps. By comparison, the pattern of inaudible voice attacks is more complex than those prolonged and frequency-specific abnormal patterns due to the variable voice content, language information, speaker characteristics, etc. The basic idea of few-shot ASD methods requires one or several abnormal samples to guide the model to fit the decision boundary better [46]. However, the patterns of inaudible voice attacks are too diverse to represent several samples, which probably leads the model to non-robustness.

10 Conclusion

In this paper, we discover that the patterns of inaudible voice attacks are incredibly diverse and distinct from that of the audible, while audible voices share similar patterns. This phenomenon motivates us to ask three research questions, and we statistically perform experiments to analyze and answer them in Sec. 3. To overcome the challenge that the attack patterns vary between devices, we design a universal detection model NormDetect, which is different from the supervised learning approach of existing studies. We adopt unsupervised learning inspired by anomaly detection. We design effective preprocessing and spectrum augmentation to converge the normal speech patterns and distinguish them from anomalies, to reduce the decision burden on NormDetect. We also built a large-scale audible & inaudible voice commands dataset of 383,320 samples and evaluated our method on it, and NormDetect performs an average AUC of 99.48% and EER of 2.23%, suggesting its effectiveness in detecting inaudible voice attacks.

Acknowledgement

We thank the anonymous reviewers for their valuable comments. We also thank the OPPO Co., Ltd. and Shilin Xiao for their support on smartphones and hardware analysis. This work is supported by China NSFC Grant 61925109, 62222114, 62071428, 62201503.

References

[1] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 103–117, 2017.

[2] Nirupam Roy, Sheng Shen, Haitham Hassanieh, and Romit Roy Choudhury. Inaudible voice commands: The long-range attack and defense. In *Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 547–560, 2018.

[3] Liwei Song and Prateek Mittal. Poster: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2583–2585, 2017.

[4] Xiaoyu Ji, Juchuan Zhang, Shui Jiang, Jishen Li, and Wenyuan Xu. Capspeaker: Injecting voices to microphones via capacitors. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021.

[5] Qiben Yan, Kehai Liu, Qin Zhou, Hanqing Guo, and Ning Zhang. Surfingattack: Interactive hidden attack on voice assistants using ultrasonic guided waves. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2020.

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

[7] Guoming Zhang, Xiaoyu Ji, Xinfeng Li, Gang Qu, and Wenyuan Xu. Eararray: Defending against dolphinattack via acoustic attenuation. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, 2021.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[9] fluent.ai. Fluent speech commands. Website, 2020. https://www.kaggle.com/tommyngx/fluent-speech-corpus.

[10] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. *arXiv preprint arXiv:1503.08895*, 2015.

[11] Rose Bruce. Comparing mems and electret condenser (ecm) microphones. Website, 2019. https://www.cuidevices.com/blog/comparing-mems-and-electret-condenser-microphones.

[12] Knowles Acoustics. Sisonic design guide. *Application Note rev*, 3, 2011.

[13] Zheyao Wang. *Micro Systems Design and Manufacturing*. Tsinghua University Press, 2015.

[14] Infineon. *MEMS microphone mechanical & acoustical implementation*. Infineon Technologies AG, Munich, Germany, 2018.

[15] Pranay Manocha, Adam Finkelstein, Richard Zhang, Nicholas J Bryan, Gautham J Mysore, and Zeyu Jin. A differentiable perceptual audio metric learned from just noticeable differences. *arXiv preprint arXiv:2001.04460*, 2020.

[16] Norman R French and John C Steinberg. Factors governing the intelligibility of speech sounds. *The journal of the Acoustical society of America*, 19(1):90–119, 1947.

[17] Bruce Kushnick. What are the public switched telephone networks, "pstn" and why you should care?, 2013.

[18] Analog Devices Inc. Admp401. Website, 2012. https://www.analog.com/media/en/technical-documentation/obsolete-data-sheets/ADMP401.pdf.

[19] InvenSense Inc. Inmp441. Website, 2012. https://invensense.tdk.com/wp-content/uploads/2015/02/INMP441.pdf.

[20] GoerTek Inc. Sd18ob371-020. Website, 2016. http://static6.arrow.com/aropdfconversion/ba847f083dbdcd2c671794bdf9a768d6b696a8bb/65d10e907c6fee6332f59a89cff87f59.pdf.

[21] Knowles Electronics Inc. Sph0644lm4h-1. Website, 2017. https://www.knowles.com/docs/default-source/model-downloads/sph0644lm4h-1_revc.pdf.

[22] Knowles Electronics Inc. Spv0842lr5h-1. Website, 2017. https://www.mouser.com/datasheet/2/218/pv0842lr5h-1_revc-1488826.pdf.

[23] STMicroelectronics Inc. Mp34dt01-m. Website, 2014. https://www.st.com/resource/en/datasheet/mp34dt01-m.pdf.

[24] Apple Inc. Apple watch user manual. Website, 2021. https://support.apple.com/guide/watch/apd8b5deac7b/watchos.

[25] Saeed V Vaseghi. Spectral subtraction. In *Advanced Signal Processing and Digital Noise Reduction*, pages 242–260. Springer, 1996.

[26] Xingliang Cheng, Mingxing Xu, and Thomas Fang Zheng. Cross-database replay detection in terminal-dependent speaker verification. *Proceedings of the Interspeech 2021*, pages 4274–4278, 2021.

[27] Xugang Lu and Jianwu Dang. Physiological feature extraction for text independent speaker identification using non-uniform subband processing. In *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–461. IEEE, 2007.

[28] Baidu. Baidu text-to-speech. https://ai.baidu.com/tech/speech/tts_online, 2021.

[29] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, Ju-Chieh Chou, Sung-Lin Yeh, Szu-Wei Fu, Chien-Feng Liao, Elena Rastorgueva, François Grondin, William Aris, Hwidong Na, Yan Gao, Renato De Mori, and Yoshua Bengio. SpeechBrain: A general-purpose speech toolkit, 2021. arXiv:2106.04624.

[30] Tecent Cloud. Tecent speech-to-text. https://cloud.tencent.com/product/asr, 2021.

[31] iFlytek Cloud. Xunfei speech-to-text. https://global.xfyun.cn/products/lfasr, 2021.

[32] Microsoft Azure. Azure speech-to-text. https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/, 2021.

[33] Kshitiz Kumar, Chaojun Liu, Yifan Gong, and Jian Wu. 1-d row-convolution lstm: Fast streaming asr at accuracy parity with lc-blstm. In *INTERSPEECH*, pages 2107–2111, 2020.

[34] Apple Inc. Siri Team. Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant. https://machinelearning.apple.com/research/hey-siri, 2017.

[35] Hadi Abdullah, Kevin Warren, Vincent Bindschaedler, Nicolas Papernot, and Patrick Traynor. Sok: The faults in our asrs: An overview of attacks against automatic speech recognition and speaker identification systems. In *2021 IEEE symposium on security and privacy (SP)*, pages 730–747. IEEE, 2021.

[36] 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.

[37] Nirupam Roy, Haitham Hassanieh, and 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.

[38] Chen Yan, Guoming Zhang, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. The feasibility of injecting inaudible voice commands to voice assistants. *IEEE Transactions on Dependable and Secure Computing*, 2019.

[39] Chaouki Kasmi and Jose Lopes Esteves. Iemi threats for information security: Remote command injection on modern smartphones. *IEEE Transactions on Electromagnetic Compatibility*, 57(6):1752–1755, 2015.

[40] Zhifei Xu, Runbing Hua, Jack Juang, Shengxuan Xia, Jun Fan, and Chulsoon Hwang. Inaudible attack on smart speakers with intentional electromagnetic interference. *IEEE Transactions on Microwave Theory and Techniques*, 69(5):2642–2650, 2021.

[41] Takeshi Sugawara, Benjamin Cyr, Sara Rampazzi, Daniel Genkin, and Kevin Fu. Light commands: laser-based audio injection attacks on voice-controllable systems. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security 20)*, pages 2631–2648, 2020.

[42] Stavros Ntalampiras, Ilyas Potamitis, and Nikos Fakotakis. Probabilistic novelty detection for acoustic surveillance under real-world conditions. *IEEE Transactions on Multimedia*, 13(4):713–719, 2011.

[43] Pasquale Foggia, Nicolai Petkov, Alessia Saggese, Nicola Strisciuglio, and Mario Vento. Audio surveillance of roads: A system for detecting anomalous sounds. *IEEE transactions on intelligent transportation systems*, 17(1):279–288, 2015.

[44] Yaoguang Wang, Yaohao Zheng, Yunxiang Zhang, Yongsheng Xie, Sen Xu, Ying Hu, and Liang He. Unsupervised anomalous sound detection for machine condition monitoring using classification-based methods. 2021.

[45] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Yuta Kawachi, and Noboru Harada. Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(1):212–224, 2018.

[46] Yuma Koizumi, Masahiro Yasuda, Shin Murata, Shoichiro Saito, Hisashi Uematsu, and Noboru Harada. Spidernet: Attention network for one-shot anomaly detection in sounds. In *Proceedings of the ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 281–285. IEEE, 2020.

[47] Dayana Ribas, Jorge Llombart, Antonio Miguel, and Luis Vicente. Deep speech enhancement for reverberated and noisy signals using wide residual networks. *arXiv preprint arXiv:1901.00660*, 2019.

[48] 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.

[49] Sara Popham, Dana Boebinger, Dan PW Ellis, Hideki Kawahara, and Josh H McDermott. Inharmonic speech reveals the role of harmonicity in the cocktail party problem. *Nature communications*, 9(1):1–13, 2018.

Appendix

A List of content in our corpus: Tab. 6

B Memory Module

For each memory slot m_i, we compute a weight w_i with the following operation:

$$w_i = \frac{\exp\left(d\left(\mathbf{z}, \mathbf{m}_i\right)\right)}{\sum_{j=1}^{N} \exp\left(d\left(\mathbf{z}, \mathbf{m}_j\right)\right)} \tag{7}$$

Where $d(...)$ here is a similarity measurement. In our implementation, we choose cosine similarity. To restrict the decoder to perform reconstruction only using a small range of choices to reduce the surplus ability of generalization, a range shrinkage is applied.

$$\widehat{w}_i = h\left(w_i; \lambda\right) = \begin{cases} w_i, & \text{if } w_i > \lambda \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

After we get the weight, the contents relevant to latent variable $\boldsymbol{Z}$ is aggregated by $\widehat{\boldsymbol{Z}}' = \widehat{\mathbf{w}}\mathbf{M}$ to form the new latent variable $\boldsymbol{Z}'$. During the training stage, the entropy loss of the memory addressing weights $\hat{w}$ is applied. It is minimized when $\hat{w}$ is in the one-hot form, so it encourages the $\hat{w}$ to be as sparse as possible.

$$E\left(\widehat{\mathbf{w}}\right) = \sum_{i=1}^{T} -\widehat{w}_i \cdot \log\left(\widehat{w}_i\right) \tag{9}$$

Table 6: List of Audible & Inaudible Voice Commands Dataset

Developer	English Wake-up Words	Developer	Chinese Wake-up Words	Phrases	Command
Amazon	Echo	JD	DingDongDingDong	serious everywhere	Take the picture
Amazon	Computer	MeiZu	NiHaoMeiZu	everyone hundred	Call my mom
Amazon	Amazon	Tencent	XiaoWeiXiaoWei	period anywhere	Navigate to my home
Google	OK Google	Lenovo	NiHaoLianXiang	unfortunately	Restart phone now
Google	Hey Google	Huawei	XiaoYiXiaoYi	together difficult	Turn off the light
Microsoft	Hey Cortana	Huawei	NiHaoYoYo	didn't agreement	
Samsung	Hey Bixby	Mobvoi	NiHaoWenWen	immediately connect	
Apple	Hey Siri	OPPO	XiaoBuXiaoBu		
OPPO	Hey Breeno	OPPO	NiHaoXiaoBu		

Algorithm 1: Training algorithm for `NormDetect`

Input: $X^{(1)}_{normal},\ldots,X^{(N)}_{normal}$
Output: encoder f_ϕ, decoder g_θ, memory slots m

$S_{normal} = Preprocessing(X_{normal})$
$\phi,\theta,m \leftarrow$ *Initialize parameters*
while *parameters* ϕ,θ,m *do not converge* **do**
  for $i = 1$ *to* N **do**
    $N(\mu_z^{(i)},\sigma_z^{(i)}) \leftarrow f_\phi(S^{(i)}_{normal})$
    $z^{(i)}$ samples from $N(\mu_z^i,\sigma_z^i)$
    $\widehat{w}^{(i)} \leftarrow d(z^{(i)},m)$
    $z'^{(i)} \leftarrow \widehat{w}^{(i)}m$
    $S^{(i)}_{rec} \leftarrow g_\theta(z'^{(i)})$
    $E = \mathcal{L}_{MenVAE}(\phi,\theta,m;S^{(i)}_{normal})$
    $\phi,\theta,m \leftarrow$ Update parameters using gradients of E (e.g. Stochastic Gradient Descent)
  end
end

Algorithm 2: `NormDetect` algorithm

Input: Anomalous dataset $x^{(1)},\ldots,x^{(N)}$, threshold γ
Output: nomal or attack

$s_{normal} = Preprocessing(x_{normal})$
$\phi,\theta,m \leftarrow$ pretrained parameters with normal samples
for $i = 1$ *to* N **do**
  $N(\mu_z^{(i)},\sigma_z^{(i)}) \leftarrow f_\phi(s^{(i)}_{normal})$
  for $l = 1$ *to* L **do**
    $z^{(i,l)}$ samples from $N(\mu_z^{(i)},\sigma_z^{(i)})$
    $\widehat{w}^{(i,l)} \leftarrow d(z^{(i,l)},m)$
    $z'^{(i,l)} \leftarrow \widehat{w}^{(i,l)}m$
    $s^{(i,l)}_{rec} \leftarrow g_\theta(z'^{(i,l)})$
  end
  $anomalyscore(s^{(i)}) = -\sum_{i=1}^{N} log\frac{p(s^{(i)}|z^{(i,l)})p(z^{(i,l)})}{q(z^{(i,l)}|s^{(i)})}$
  if $anomalyscore(s^{(i)}) < \gamma$ **then**
    $x^{(i)}$ is normal audio
  else
    $x^{(i)}$ is attack audio
  end
end

C Training process of `NormDetect` model

We use a VAE to learn the patterns from standard samples and a memory module to limit the surplus ability of generalization. The structure is shown in Fig. 11. Algorithm. 1 shows the training framework of our `NormDetect` model.

D Testing process of `NormDetect` Model

For the anomalies detection task, Algorithm. 2 describes the process. With the separated distribution between standard audio and attack audio, a threshold can be decided directly according to the requirements of the device.

E Impact of Microphones

Investigating microphones is meaningful because it can eliminate minor influence factors existing in the smart devices scenario and focus on the microphone itself. We set up the ten-microphone testbed, covering five leading manufacturers, three output types, and MEMS and ECM. Those microphones simultaneously record eight audible and inaudible commands modulated by the 25 kHz ultrasonic carrier signal at seven distances. We utilize the metrics of AUC and EER to represent the discriminatory ability of our method. The results show that `NormDetect` can distinguish inaudible voice attacks captured by microphones well, with the maximum AUC of 100%, while it gets a relatively lousy performance on SPQ0410 due to its internal noise suppression.

F Details of Adaptive Attacks

Improving the Attack Audio's Quality. We implemented the model in [47] to represent the inaudible voice attacks transformation process, similar to adopting frequency sweeping [48] to obtain a mapping function on each frequency for the original and attack audio pairs. The model can facilitate the reverse transformation process in by simply swapping the input/output pairs. Fig. 16 indicates how to perform the noise-removal attacks. We trained $4(d) \times 7(D) = 28$ reverse trans-

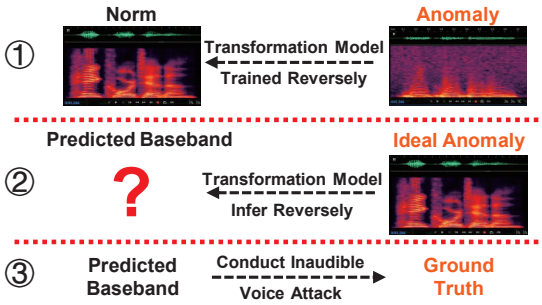


Figure 16: ①: Train the transformation model reversely by attacks to basebands. ②: Infer the basebands from ideal attack audios. ③: Utilize the predicted basebands to perform inaudible voice attacks.

formation models and inferred $4(d) \times 16(C) \times 7(D) = 448$ compensated basebands based on the TTS audios that are regarded as the ideal attacks. Ultimately, we got 448 attack audios on the 4 devices and obtained an average SDR at 98.44% for noise-removal attacks, even better than the baseline. We believe it is because the inaudible voice attack is fundamentally a lossy process due to the hardware's inherent features. For example, the nonlinearity that the attack exploits will inevitably cause undesired harmonic noises, which are difficult to remove. For this reason, using the compensated baseband for the attack introduces severe harmonic mistuning effect [49], making the attack audio more disparate from the normal pattern.

Creating Adversarial Examples. We use the transformation model proposed in the noise-removal attack to implement the end-to-end pipeline. Under the white-box setting, we can get each attack sample's anomaly score and optimize it based on the gradient for 2000 iterations. The optimized basebands are expected to get anomaly scores like normal voices even after modulated and emitted by the inaudible voice attack. We performed inaudible voice attacks based the above optimized basebands to validate the robustness of NormDetect against such optimized attacks. We suspect that the adversarial examples fail for two main reasons:

1. The real-world transformation process is not stable. With the attacking device and the victim device in a completely fixed position, we perform three consecutive attacks, where each attack signal consists of 16 commands. We also replay the same voice commands audibly by laptop three consecutive times. Therefore, we can compare the stability of audible and inaudible voice commands by quantifying their similarity to another referenced inaudible voice commands with DPAM (an audio similarity metric tool).

Fig. 17 shows that the similarities of three consecutive audible audio plays (solid lines) are very close, indicating their stability, while the similarities of the inaudible attacks (dashed lines) change drastically. We believe the attack's lower stability is caused by the longer transformation process including signal generator modulation, over-the-air propagation, and

[‖]d: device; C: Speech Command; D: Distance

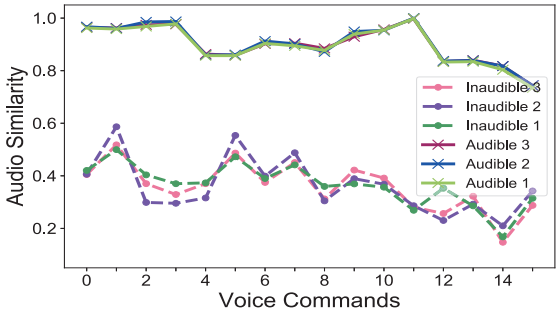


Figure 17: Audio similarity of 6 groups of different voice commands to a normal audios group as the reference, based on DPAM (an audio similarity metric tool).

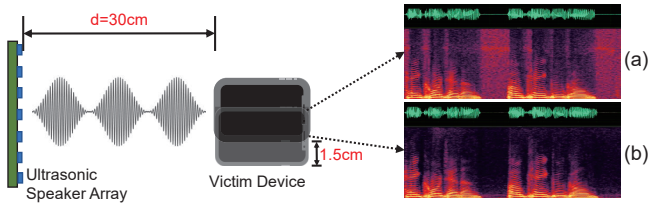


Figure 18: Perform inaudible voice attacks in different relative positions (with a 1.5cm displacement), obtaining (a) & (b), respectively.

nonlinear demodulation on the devices. Therefore, it is difficult to deliver a precise adversarial example through the physical inaudible voice attack process.

2. The real-world transformation process is sensitive to relative positions. Fig. 18 shows that the patterns of the attack spectrums change greatly when the same attack signal is emitted at slightly different locations (around 1.5cm apart), implying that the transformation process is very sensitive to locations. Therefore, in practice, the attacker needs to create adversarial examples that are robust at different positions, which is more challenging than traditional over-the-air audio adversarial examples, and existing methods based on the room impulse response (RIR) may not apply to ultrasounds directly. In summary, the different nature between audible sounds and ultrasounds significantly increases the bar of adversarial examples for inaudible voice attacks, which we call for future work.

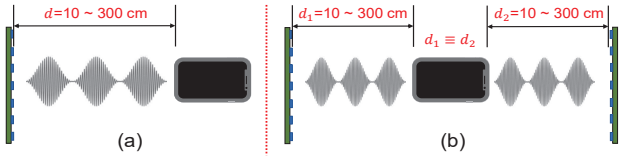


Figure 19: (a) single-injection scenario: attack the bottom microphone of smartphones directly; (b) multi-injection scenario: attack both of the microphones directly, keeping $d_1 \equiv d_2$. Attacks are all conduced at 7 distances.