Learning-based Practical Smartphone Eavesdropping with Built-in Accelerometer

Zhongjie Ba*†§, Tianhang Zheng‡§, Xinyu Zhang*, Zhan Qin*, Baochun Li‡, Xue Liu† and Kui Ren*
*School of Cyber Science and Technology, Zhejiang University
Email: xinyuzhang53@zju.edu.cn, qinzhan@zju.edu.cn, kuiren@zju.edu.cn
†School of Computer Science, McGill University
Email: zhongjie.ba@mcgill.ca, xueliu@cs.mcgill.ca
‡Department of Electrical and Computer Engineering, University of Toronto
Email:th.zheng@mail.utoronto.ca, bli@ece.toronto.edu

***Abstract*—Motion sensors on current smartphones have been exploited for audio eavesdropping due to their sensitivity to vibrations. However, this threat is considered low-risk because of two widely acknowledged limitations: First, unlike microphones, motion sensors can only pick up speech signals traveling through a solid medium. Thus, the only feasible setup reported previously is to use a smartphone gyroscope to eavesdrop on a loudspeaker placed on the same table. The second limitation comes from a common sense that these sensors can only pick up a narrow band (85-100Hz) of speech signals due to a sampling ceiling of 200Hz. In this paper, we revisit the threat of motion sensors to speech privacy and propose AccelEve, a new side-channel attack that employs a smartphone's accelerometer to eavesdrop on the speaker in the same smartphone. Specifically, it utilizes the accelerometer measurements to recognize the speech emitted by the speaker and to reconstruct the corresponding audio signals. In contrast to previous works, our setup allows the speech signals to *always* produce *strong* responses in accelerometer measurements through the shared motherboard, which successfully addresses the first limitation and allows this kind of attacks to penetrate into real-life scenarios. Regarding the sampling rate limitation, contrary to the widely-held belief, we observe up to 500Hz sampling rates in recent smartphones, which almost covers the entire fundamental frequency band (85-255Hz) of adult speech. On top of these pivotal observations, we propose a novel deep learning based system that learns to recognize and reconstruct speech information from the spectrogram representation of acceleration signals. This system employs adaptive optimization on deep neural networks with skip connections using robust and generalizable losses to achieve robust recognition and reconstruction performance. Extensive evaluations demonstrate the effectiveness and high accuracy of our attack under various settings.**

I. Introduction

Smartphones have permeated into our daily lives as an indispensable communication interface to the rest of the world. Among all different modalities of communication, voice communication is always considered as one of the top choices. Due to its importance, the system permission level of microphone usage is the highest by default in most operating systems [2]. A significant amount of research in the literature focused on how to eavesdrop on a user's phone call by exploiting the vulnerability of communication protocols, or by implanting a backdoor to access the permission of utilizing a microphone.

In this paper, we consider the problem of eavesdropping on the speaker in a smartphone by side-channel attacks without the requirement of sensitive system permissions. Instead of hacking into the operating system and gaining access to the administrator authority, we recognize and reconstruct the speech signals emitted by a smartphone's speaker through analyzing the measurements of motion sensors on the same smartphone. This attack could arise due to the following reasons. First, because the acceleroemeter and gyroscope are considered low-risk, they are usually set to be zero-permission sensors and can be accessed without warning smartphone users. Second, motion sensors can response to external vibrations, which allows them to pick up certain audio signals. Additionally, there is an overlap between the fundamental frequencies of human voice and the sampling frequencies of smartphone sensors. Therefore, it is theoretically possible to capture speech signals by zero-permission motion sensors.

In the literature, motion sensor based speech recognition has attracted a number of studies. Michalevsky *et al.* [32] is the first work towards this direction, which utilizes a smartphone's gyroscope to pick up surface vibrations incurred by an independent loudspeaker placed on the same table. The captured vibrations are then analyzed to recognize the speech information played by the loudspeaker. The proposed method suffers from its feasibility and the poor performance on recognition accuracy, i.e., the accuracy is only 26% in differentiating single digit pronunciations. Another line of research focuses on exploiting the air as the medium rather than a solid surface. Zhang *et al.* [44] use the accelerometer as a "microphone" to detect the voice input of the smartphone. Recently, Anand *et al.* [5] (S&P 2018) study the problem of detecting speech signals through either solid surfaces (such as desks) or air. Their experimental results show that among all tested audio sources and medium, only the loudspeaker placed on the desk has sufficient power and sound path for generating and transmitting vibrations to motion sensors. Based on this observation, [5] claims that the threat under investigation will not go beyond the loudspeaker setup studied in [32].

§The first two authors contribute equally to the paper.

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.24076
www.ndss-symposium.org

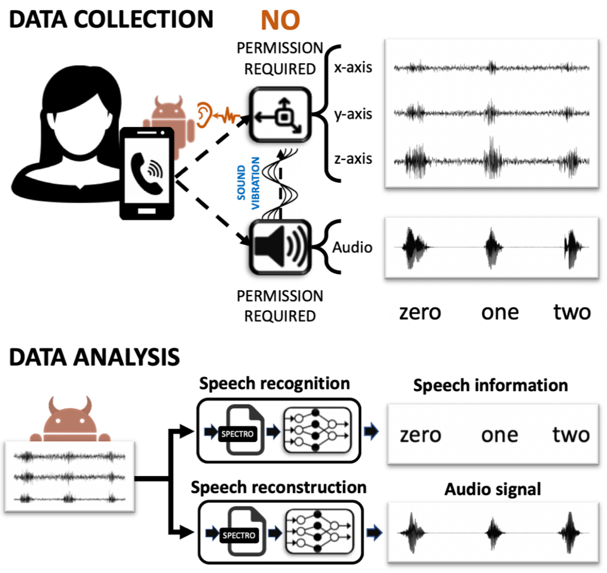


Fig. 1. Accelerometer-based smartphone eavesdropping and the workflow of speech recognition and speech reconstruction.

However, all of the above works failed to cover the most adverse setup, where the motion sensors are utilized as a side-channel to capture the speech signals played by the speaker *on the same smartphone*. In this case, the motion sensors and the speaker are in physical contact with the same board and locate in very close proximity to each other. Hence, contrary to the claim in [5], speech signals emitted by the speaker will always have a significant impact on the motion sensors like the gyroscope and the accelerometer, no matter where and how the smartphone is placed (on a table or in your hand).

Furthermore, all previous works in the literature share a misleading common sense that the sampling rates of the accelerometer and gyroscope in Android-powered smartphones cannot exceed 200Hz [32], [44], [5]. Because the typical fundamental frequency for adult speech lies in the range 85-255Hz [38], [7], the previously believed sampling ceiling leads to a consensus that motion sensors can only capture a very small range of human speech in 85-100Hz. However, we show that this is not the case. According to the official documents for Android [2], an Android application selecting the `SENSOR_DELAY_FASTEST` mode will receive sensor measurements as fast as possible. Thus the actual sampling rates of motion sensors are determined by the performance of the smartphone, which has been confirmed by our experiments. In particular, for certain smartphones released in 2018, we observe a sampling frequency of up to 500Hz, which indicates that the accelerometer is able to capture speech information in the range of 85-250Hz. It covers almost the entire fundamental frequency band (85-255Hz) of adult speech.

In this paper, we address the limitations of previous works by presenting a novel and practical setup and a deep learning based system for speech recognition and reconstruction, which outperforms all similar related works. In our setup, the adversary is a spy app whose objective is to eavesdrop on the speaker in the same smartphone. When the speaker emits speech signals (e.g., during a phone call), the spy app collects accelerometer measurements in the background and utilizes the collected signals to recognize and reconstruct the played speech signals. It is worth noting that the spy app could be disguised as any kind of mobile apps since accessing the accelerometer does not require any permission.

The main intent of the proposed system is to recognize and reconstruct the speech signals from the accelerometer measurements. Since raw acceleration signals usually capture multiple "words" and could be severely distorted by human movement, our system first implements a preprocessing module to automatically eliminate significant distortions from acceleration signals and to cut long signals into single word segments. We then convert each single word acceleration signal to its spectrogram representation and pass it to a recognition module and a reconstruction module for further analysis.

The recognition module adopts the DenseNet [24] as the base network to recognize the speech information (text) carried by the spectrogram of the acceleration signal. Extensive evaluations demonstrate that our recognition module achieves new state-of-the-art results under different settings. In particular, our recognition module has 78% accuracy on recognizing 10 digits and 70% accuracy on identifying 20 speakers when the smartphone is placed on a table, while the previous SOTA results are 26% accuracy in the digit task and 50% accuracy on recognizing only 10 speakers. Also, evaluations under different noisy ambient conditions demonstrate the robustness of our recognition model. Except for digits and letters, we also demonstrate that our recognition and reconstruction models can be used to identify hot (sensitive) words in phone calls. With the help of our speaker-identification model, the adversary might acquire multiple pieces of sensitive information for the callers' contacts by linking the hot words identified across multiple phone calls to a specific caller. Furthermore, we also realize an end-to-end attack based on our recognition model in real-world conversations.

In the reconstruction module, we implement a reconstruction network that learns the mapping between the accelerometer measurements and the audio signal played by the smartphone speaker. Because most of the speech information in the high frequency band are the harmonics of the fundamental frequency, the reconstruction module can convert an acceleration signal into an audio (speech) signal with enhanced sampling rates (1500Hz). According to our experimental results, the reconstruction module was able to recover nearly all the vowel information, including the fundamental frequency components in the low frequency band and its harmonics in the high-frequency band. The unvoiced consonants are not recovered because these components have no information distributed in the frequency band below 2000Hz.

Our contributions are summarized as follows:

1) We propose **AccelEve**, an accelerometer-based side channel attack against smartphone speakers. Contrary to the previous belief, the proposed setup infiltrates this kind of attacks into common scenarios in daily life, e.g., answering a phone call or receiving voice messages. Comprehensive experiments are conducted to evaluate the feasibility of the setup.
2) We first report an important observation that accelerometers on recent smartphones almost cover the

entire fundamental frequency band of adult speech.

3) We design a deep learning-based system to recognize and reconstruct speech signals only from accelerometer measurements. Extensive evaluations on the existing and our datasets show that the system significantly and consistently outperforms existing solutions[1]. To the best of our knowledge, the proposed system gives the first trail on accelerometer-based speech reconstruction.

II. Background and Related Work

In this section, we first describe the design of the motion sensors on current smartphones. We then review the existing works that exploit motion sensors to capture speech signals and other topics related to AccelEve.

A. MEMS Motion Sensors

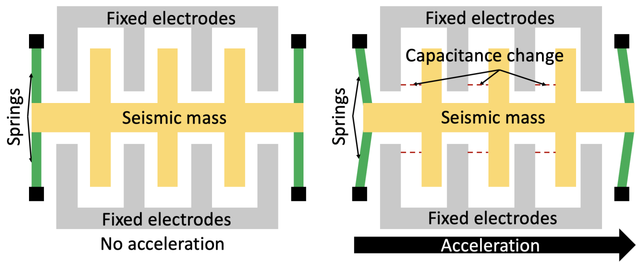
Modern smartphones typically come equipped with a three-axis accelerometer and a three-axis gyroscope. These sensors are highly sensitive to the motion of the device and have been widely applied to sense orientation, vibration, shock, etc.

Accelerometer: a three-axis accelerometer is a device that captures the acceleration of its body along three sensing axes. Each axis is normally handled by a sensing unit consisting of a movable seismic mass, several fixed electrodes, and several spring legs, as shown in Fig.2(a). When the accelerometer experiences an acceleration along a sensing axis, the corresponding seismic mass shifts to the opposite direction and creates a change in the capacitance between the electrodes. This change yields an analog signal that is then mapped to acceleration measurements.

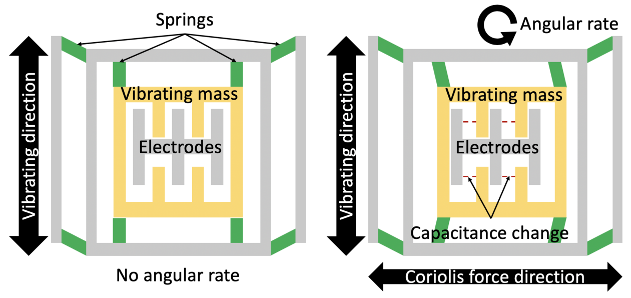
Gyroscope: gyroscopes on smartphones normally leverage the Coriolis force [1] to measure the angular rate around three axes. As shown in Fig.2(b), the sensing unit for each axis has a similar structure to the accelerometer's, except that the mass is constantly vibrating and is allowed to move along two axes. When the gyroscope experiences an external angular rate, due to the Coriolis effect, the mass tends to continue vibrating in the same plane and exerts a Coriolis force perpendicular to the rotating axis and the moving direction of the mass. This force creates a displacement of the mass and changes the capacitance between the electrodes. Through measuring the capacitance change, the angular rate of the device can be obtained.

In practice, the information captured by a motion sensor is determined not only by its sensitivity to the surrounding environment, but also by the sampling frequency. On Android-powered smartphones, motion sensors can be accessed with four delay options as listed in Table I. Each option specifies an interval at which sensor measurements are sent to the application. In particular, if an application selects `SENSOR_DELAY_FASTEST`, sensor measurements will be sent to that application as fast as possible and the actual sampling rate will be mainly determined by the performance of the smartphone. In 2014, the actual sampling rate achieved 200Hz [32], which allows the motion sensors to accurately capture frequency components below 100 Hz, according to the *Nyquist sampling theorem*.

[1]Our code and collected datasets are available on https://github.com/tianzheng4/learning_speech_from_accelerometer.



(a) Accelerometer structure



(b) Gyroscope structure

Fig. 2. Sketches of an accelerometer and gyroscope.

TABLE I. Sampling frequencies supported by Android [2].

Delay Options	Delay	Sampling Rate
`SENSOR_DELAY_NORMAL`	200 ms	5 Hz
`SENSOR_DELAY_UI`	20 ms	50 Hz
`SENSOR_DELAY_GAME`	60 ms	16.7 Hz
`SENSOR_DELAY_FASTEST`	0 ms	As fast as possible

B. Speech Recognition via Motion Sensors

Human speech signals have a fundamental frequency that carries important linguistic and non-linguistic information such as naturalness, emotion, and speaker idiosyncrasy [18]. It is defined as the vibration rate of the vocal folds and varies widely depending on age, gender, individual physiological dispositions, etc [13], [19]. Typically, the fundamental frequencies for an adult male and adult female lie in the range 85-180Hz and 165-255Hz, respectively. [38], [7]. *Because the frequency range of this fundamental frequency partially overlaps with that of smartphone sensors, the accelerometer and gyroscope have been exploited to capture a small portion of the speech signals in the low frequency band.*

Michalevsky *et al.* [32] (Usenix Security 2014) study the setup where a smartphone is placed on the *same solid surface* as a *loudspeaker*. They employ the smartphone's *gyroscope* to "pick up" speech signals emitted by the loudspeaker and use the captured information to conduct speech recognition and speaker identification. In this scenario, the signals captured by the gyroscope are actually surface vibrations. Because the gyroscope shows low sensitivity to surface vibrations and suffers from limited sampling rate (200Hz), it is difficult to achieve high success rates for the recognition tasks.

Zhang *et al.* [44] study the setup where a *user* speaks to a smartphone held in her hand or placed on a desk. In this setup, the authors employ the *accelerometer* to pickup speech signals traveling through the *air* and used the obtained accelerometer readings to conduct hot words detection ("Okay Google" and "Hi Galaxy"). However, the experimental results in [5] suggest that the speech signals traveling through the air are unlikely to have any noticeable impact on motion sensors. Therefore, the accelerometer may not be able to collect sufficient speech information through airborne vibrations.

In order to better understand the threat of motion sensors to speech privacy, Anand *et al.* [5] (S&P 2018) systematically study the response of accelerometers and gyroscopes to speech signals in various settings. They stimulate both sensors with human-rendered, laptop-rendered and loudspeaker-rendered speech signals traveling through the air or a solid surface. Their experimental results show that only loudspeaker-rendered speech signals traveling through a solid surface can create noticeable impacts on motion sensors. Based on this observation, Anand *et al.* [5] claim that the threat under investigation does not go beyond the *Loudspeaker-Same-Surface* setup studied by Michalevsky *et al.* [32].

However, all the above works failed to cover the most adverse setup where the motion sensors are on the same smartphone as the target speaker. ***In this case, the motion sensors and the speaker will be in physical contact with the same board and locate in very close proximity to each other. Thus here, contrary to the claim in [5], speech signals emitted by the speaker will always have a significant impact on the gyroscope and the accelerometer. It does not matter where and how the smartphone is placed, which could be on a table (solid surface), on a bed (soft surface), or in your hand.*** Moreover, a smartphone speaker is more likely to reveal sensitive information than an independent loudspeaker. For instance, when a user is making a phone call, an application running in the background can access to the zero-permission motion sensors and use the collected signals to recover sensitive speech information. In this paper, we look into this setup and explore accelerometer-based speech recognition and reconstruction using deep learning techniques.

In parallel and independent work, Anand *et al.* (arXiv 2019) [6] also study accelerometer-based speech recognition under the setup that the accelerometer is on the same smartphone as the speaker. While Anand *et al.* employ existing feature selection and classification tools that work well on small datasets, we implement deep learning-based speech recognition that achieves higher accuracy. The proposed model is also more robust as we investigate a comprehensive set of factors and address them with effective preprocessing approaches. Moreover, we are the first to implement accelerometer-based speech reconstruction and report that accelerometers on recent smartphones almost cover the entire fundamental frequency band of adult speech.

C. *Other Topics Related to* **AccelEve**

In the literature, accelerometers and gyroscopes have been extensively studied to sense vibrations in various application scenarios. Marquardt *et al.* [29] use the accelerometer in a smartphone to collect and decode vibration signals incurred by the keypresses on a nearby keyboard, so as to infer the text entered by the user. Owusu *et al.* [34], Miluzzo *et al.* [33], Xu *et al.* [42] and Cai *et al.* [10] show that motion sensors on a smartphone can be utilized to infer keystrokes on its touch screen. Matovu *et al.* [31] show that a smartpone's accelerometer measurements can be used to detect and classify songs played by the smartphone. Son *et al.* [37] explore the possibility of incapacitating a drone through generating a sound noise at the resonant frequencies of the gyroscope on that drone. Feng *et al.* [17] propose an accelerometer-based continuous authentication system for voice assistants (e.g., Siri and Google Now). Through cross-checking the voice input with a wireless accelerometer attached to the user's skin, the system enables the voice assistant to differentiate the owner's command from voice signals that originate from others. VibWrite [28] enables fingerprint-input on ubiquitous surfaces leveraging vibration signals generated by a vibration motor. They show that vibration signals on a solid surface can be used to extract unique features that reflect the touching location and force of a user's finger pressing. Another line of research has focused on accelerometer-based trajectory recovery and activity recognition. Han *et al.* [21] demonstrate that the accelerometer on a device can be used to determine the trajectory and location of the device owner if he/she is driving in a vehicle. [27], [30], [35] explore how smartphone accelerometers and gyroscopes can be used to detect and identify user activities (e.g., walking, jogging, bicycling etc).

III. Problem Formulation

In this paper, we consider the novel problem of launching a side-channel attack against smartphone speakers: recognize and reconstruct the speech signals played by the smartphone speaker through exploiting the accelerometer on the same device. Our attack mainly focuses on the Android system due to its prevalence as the open source mobile operating system (not taking its variants into account). Please note that the proposed approach may also be extended to attack iOS since the maximum sampling rate of the accelerometer in iOS is also determined by the maximum frequency supported by the hardware. The reason we prefer to use the accelerometer is that it is more sensitive to vibrations than the gyroscope. A comparison between the gyroscope and the accelerometer is shown in Fig. 3.

A. *Threat Model*

We assume a victim user with a high-end smartphone. The smartphone plays a speech signal that contains private information. In this paper, we focus on private information made up of numbers, letters, and hot words, such as social security number, a password, a credit card number, etc. The smartphone could be on a table or in the user's hand.

The adversary is a spy app whose objective is to extract the private information contained in the speech signal. The spy app continuously collects accelerometer measurements in the background and tries to extract speech information when the smartphone speaker plays an audio signal (e.g., during a phone call or voice message). The detection of the playing activity can be achieved through checking the high-frequency components of the collected accelerometer measurements. Although the accelerometer can also be affected by daily activities, these activities rarely affect frequency components above 80Hz (as will be shown in section IV-C).

For the extraction of the private information, we implement accelerometer-based *speech recognition* and *speech reconstruction*. *Speech recognition* converts acceleration signals to

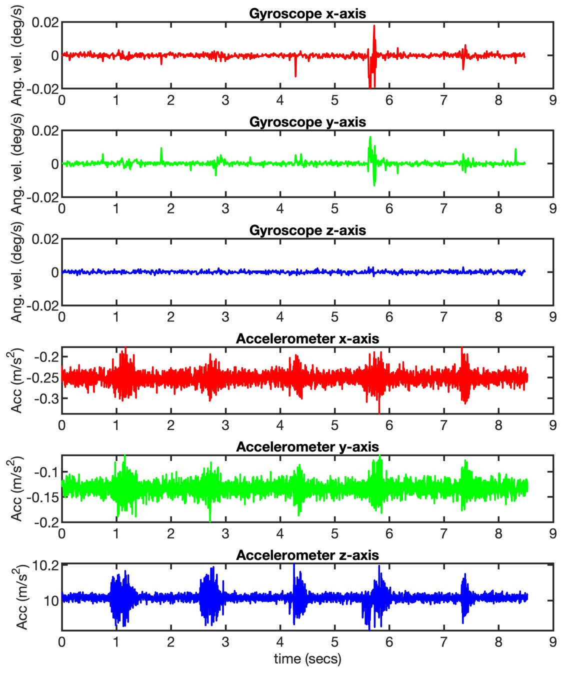


Fig. 3. The response of a smartphone's gyroscope and accelerometer to the same audio signal. The gyroscope can also response to speech signals emitted by the speaker on the same device. However, the gyroscope exhibit much lower audio response than the accelerometer.

text. It allows the adversary to recognize pre-trained numbers, letters, and hot words from the accelerometer measurements. *Speech reconstruction* reconstructs voice signals from acceleration signals. It allows the adversary to double-check the recognition results with human ears. Because the reconstruction model mainly learns the mapping between signals rather than the semantic information, it is more generalizable to untrained words compared with recognition model.

The spy app could be disguised as any app running on the smartphone since accessing the accelerometer does not require any permission. Without loss of generality, in this paper, we collect accelerometer readings (signals) through a third-party Android application named AccDataRec running in the background. This application requires zero permission to record three-axis accelerometer readings along with a timestamp. Ear speakers and headphones are not considered in this paper as they can hardly affect the accelerometer.

B. Attack Scenarios

The side channel attack described in this paper allows an adversary to recognize pre-trained hot words emitted (played) by a smartphone through analyzing its accelerometer measurements. This attack will affect not only the owner of the smartphone but also his/her contacts.

For the smartphone owner, the spy app on the smartphone may steal the following information: 1) Voice memo: the spy app can catch pre-trained key words when a user listens to his/her voice memos. Because the voice memo is usually used to record important private information (e.g., passwords, phone number, and postcode), exposure of certain key words could lead to serious privacy leakage. 2) Location information: because most navigation apps on current smartphones support voice guidance, the spy app may track user location through analyzing the geographical information emitted by the smartphone speaker. 3) Music and video preferences: our attacks can be extended to detect the music or video played by a smartphone, which could be further analyzed to construct the user's listening and watching habits.

The spy app can also eavesdrop on remote callers who make phone calls or send voice messages to the victim's smartphone. For instance, Alice (the victim) makes a phone call to Bob and request for a credit card number with its CVV number. Because Bob's voice will be played by Alice's smartphone, the spy app on Alice's phone will be able to extract the spoken digits and hot words. In this attack, the adversary can only eavesdrop on the remote caller since the voice of the smartphone owner will not be played by his/her smartphone. Although listening to one side of the conversation may miss important context information in one phone call, the adversary can further identify the remote caller through analyzing the accelerometer measurements (as will be shown in section VI-B). This allows the adversary to link the private information extracted across multiple phone calls to a specific caller. Once the adversary gathers multiple pieces of information for a specific contact (e.g., social security number, credit card number, phone number, password, etc), the privacy of that contact will be seriously threatened.

IV. Feasibility Study

As mentioned earlier, our core idea is to exploit the accelerometer on a smartphone as a zero-permission "microphone" to eavesdrop on the speaker of the same device. We now present our experimental validations to demonstrate the feasibility (severity) of this attack from three aspects: significance, effectiveness and robustness.

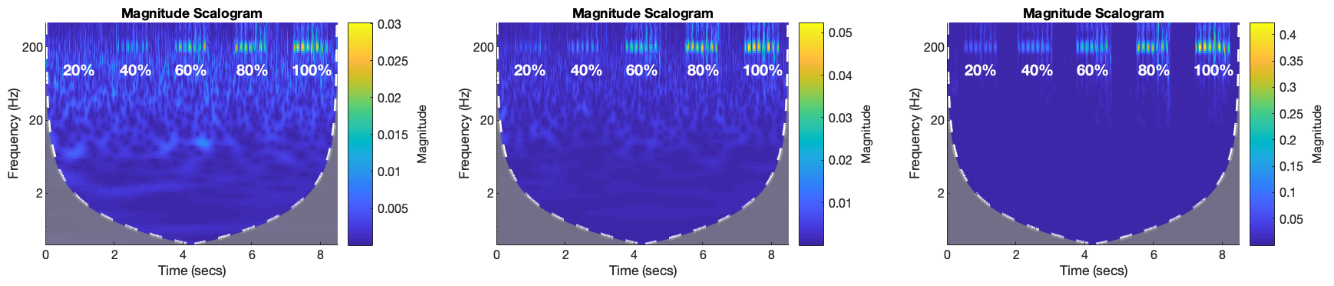
We note that all the acceleration signals presented in IV-A and IV-C are preprocessed with the interpolation approach proposed in section V. We use this approach to resolve the problem of unstable sampling interval so that spectrum analysis can be applied. The resulting acceleration signals have a fixed sampling rate of 1000 Hz.

A. Significance

The core intuition behind this attack is that the accelerometer and speaker of the same device will always be in physical contact with the same board and locate in very close proximity to each other, thereby enabling speech signals to always produce significant responses in accelerometer measurements.

In order to validate this hypothesis, we evaluate the responses of a smartphone's accelerometer to its loudspeaker at different volume levels. Specifically, we generate a single tone signal at 200 Hz and play it on a Samsung S8 at 20%, 40%, 60%, 80%, and 100% of the highest volume supported by the smartphone. For each setting, we place the smartphone on a table and play the speech signals for one second. The accelerometer readings are collected simultaneously through the AccDataRec APP running in the background.

After recording the acceleration signals, we calculate the continuous wavelet transform for each axis and generate the



(a) Scalogram: x-axis (b) Scalogram: y-axis (c) Scalogram: z-axis

Fig. 4. The response of a smartphone accelerometer to a 200 Hz single tone signal played by its loudspeaker at different volume levels. The acceleration signals obtained at different volume are concatenated for better display and comparison. Please note that the range of magnitudes is different for different scalograms.

corresponding scalograms, which show how the magnitude of frequency components changes over time. ***The obtained scalograms are depicted in Fig. 4, where a brighter region in the scalogram indicates stronger frequency components. It can be observed that, starting from the volume level at 20% of the highest volume, the regions around 200 Hz are becoming brighter (especially for the z axis), indicating that the accelerometer successfully captures the voice signals emitted by the loudspeaker.***

To facilitate quantitative comparison between the accelerometer's response under different settings, we further quantify the accelerometer's audio response as:

$$AR_{dB} = 10log_{10}(\frac{P(S)}{P(N)}),$$

where P is the summed squared magnitude, S and N are two acceleration signals recorded with and without the presence of speech signals (played by the speaker). This AR_{dB} is similar to the definition of signal to noise ratio (SNR) except that the signal (S) here is a mixture of the captured speech signal and noise. In the ideal case where the noise signal remains constant overtime, an audio response AR_{dB} higher than zero indicates that the speech signal under investigation has affected the accelerometer. However, due to the changing nature of noise, the AR_{dB} calculated from pure noise signals (without any audio information) could also fluctuate within a small range around zero. Through studying the pure noise signals from several smartphone accelerometers, we observe that a threshold of three can effectively determine if the accelerometer has been significantly affected by speech signals.

Table II lists the audio response calculated from each specific setting. It can be observed that the accelerometer's audio response varies significantly with axis and increases with higher volume. The sensing units along the x-axis, y-axis, and z-axis are able to capture speech signals above 60%, 60% and 20% volume level respectively. One important observation is that, for each speech signal under investigation, the tested accelerometer always has the strongest response along the z-axis, followed by the y-axis, and then the x-axis. In fact, this relationship remains constant regardless of whether the smartphone is placed on the table or held by hand. This consistency can be explained by the structure of the accelerometer's sensing unit shown in Fig. 2(a). For each sensing unit, the seismic mass only vibrates along its sensing axis and therefore is less sensitive to vibration signals coming from other directions. Because the vibrations rendered by the smartphone speaker will always transmit through the motherboard of the smartphone and "hit" the smartphone accelerometer from the same direction, the accelerometer will always have the most significant audio response along the same axis. This consistency is important as it helps determine the portion of the speech information captured from each sensing axis. In this paper, we refer to the axis with strongest response as the *dominant axis* of the smartphone.

TABLE II. THE AUDIO RESPONSE OF EACH SETTING IN FIG. 4.

Volume	20%	40%	60%	80%	100%
AR_{dB} of the x-axis	1.0593	1.1715	4.2761	4.1370	5.2496
AR_{dB} of the y-axis	0.5710	2.0818	7.1051	8.8069	9.5171
AR_{dB} of the z-axis	14.8148	18.6134	23.0665	25.6657	27.1832

B. Effectiveness

In the previous literature, it is a common sense that, the sampling rates of the accelerometer and gyroscope in Android-powered smartphones cannot exceed 200 Hz [32], [44], [5]. Since the typical fundamental frequency for an adult male and adult female respectively lie in the range 85-180Hz and 165-255Hz [38], [7], this common sense implies that the sensors can only pickup a very small range of human speech in the 85-100 Hz frequency band (according to Nyquist Theorem) and thus the effectiveness of the attack is limited. ***However, as stated in section II-A, if an Android application selects*** SENSOR_DELAY_FASTEST***, the official documents [2] claim that sensor measurements will be sent to the application as fast as possible. In this context, we hypothesize that this fastest sampling rate for the recent smartphone models may surpass 200Hz.***

To verify the above hypothesis, we test eight smartphones released in different years and list the actual sampling rates of the accelerometers in table III. The results confirm that the actual sampling rate of the accelerometer increases rapidly with the development of the smartphone models. For high-end smartphones released after 2017, their accelerometers have reached sampling frequencies above 400 Hz and therefore should be able to pickup a considerable range of human speech. In particular, for Huawei P20 Pro and Mate 20, the sampling rate of their accelerometers have reached as high as 500 Hz, which allows them to pick up frequency components up to 250 Hz. Because the highest fundamental frequency for adult speech is just 255 Hz, these two smartphones almost can cover the entire fundamental frequency band of adult speech. *The takeaway message is that the threat of motion sensors to*

TABLE III. ACTUAL SAMPLING RATE OF DIFFERENT SMARTPHONES WHEN CONFIGURED AT SENSOR_DELAY_FASTEST.

Model	Year	CPU	Sampling Rate
Moto G4	2016	4x1.5 GHz & 4x1.2 GHz	100 Hz
Samsung J3	2016	4x1.3 GHz	100 Hz
LG G5	2016	2x2.15 GHz & 2x1.6 GHz	200 Hz
Huawei Mate 9	2016	4x2.4 GHz & 4x1.8 GHz	250 Hz
Samsung S8	2017	4x2.35 GHz & 4x1.9 GHz	420 Hz
Google Pixel 3	2018	4x2.5 GHz & 4x1.6 GHz	410 Hz
Huawei P20 Pro	2018	4x2.4 GHz & 4x1.8 GHz	500 Hz
Huawei Mate 20	2018	2x2.6 GHz & 2x1.92 GHz & 4x1.8 GHz	500 Hz

speech privacy has become a serious issue and will continue to grow due to the rapid improvement of smartphone models.

C. Robustness

Accelerometers on smartphones are highly sensitive to the environmental noise. In the adversarial setup under investigation, the noise may come from following sources: *hardware distortion*, *acoustic noise*, *human activities*, *self-noise and surface vibration*. ***We look into all these kinds of noise, and find that most of these noises are either unlikely to affect accelerometer readings or can be effectively eliminated, except for the acoustic noise contained in the audio signal played by the smartphone speaker. The impact of such noise on the recognition accuracy is evaluated in section VI-C***

Hardware distortion is a systematic distortion incurred by manufacturing imperfection. Small variations in electromechanical structures (e.g., gap between the fixed electrodes and the flexibility of the seismic mass) result in slightly different measurement values [15]. For illustration, we place four smartphones on the same table and record their responses to gravity along six directions (+x, -x, +y, -y, +z, -z).

TABLE IV. THE GRAVITY RESPONSE OF DIFFERENT ACCELEROMETERS ALONG THE SIX DIRECTIONS. A AND B ARE TWO SMARTPHONES OF THE SAME MODEL.

Device	Response to Gravity (m^2/s)					
	+x	-x	+y	-y	+z	-z
Samsung S8 A	9.64	10.01	9.94	9.78	9.83	9.83
Samsung S8 B	9.59	10.05	10.02	9.76	9.71	9.82
Google Pixel 3 A	10.16	9.41	9.89	9.67	9.59	9.82
Google Pixel 3 B	9.78	9.81	9.77	9.79	9.80	9.81

As shown in Table IV, the gravity measured along different directions are slightly different, which indicate the existence of hardware distortion. Given a specific accelerometer, its actual measurement along ith axis can be modeled as [16]

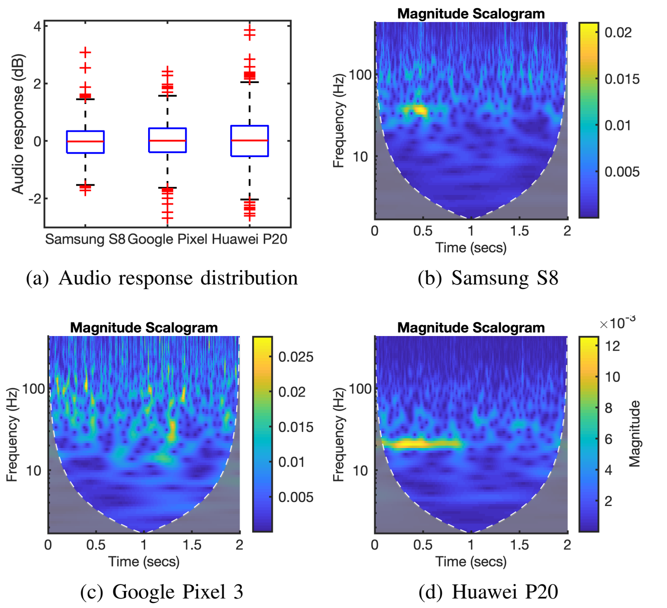
$$a_i^M = S_i(a_i) + O_i,$$

where a_i is the actual acceleration, S_i and O_i respectively represent the gain and offset errors. Therefore, the actual acceleration signal along the ith axis can be recovered by

$$a_i = (a_i^M - O_i)/S_i,$$

where S_i and O_i are calculated through analyzing the accelerometer's response to gravity[12], [9].

In the proposed attack, it is even not necessary for the adversary to recover the actual acceleration signal. This is because that the speech information captured by the accelerometer mainly distributed in the frequency components above 85 Hz while the offset error can only affect the DC (0 Hz) component. For the gain error, it only affects the "loudness" of captured speech signal and thus will not distort its spectrum. *Therefore, we address the hardware distortion simply by eliminating the DC component of the captured signals.*



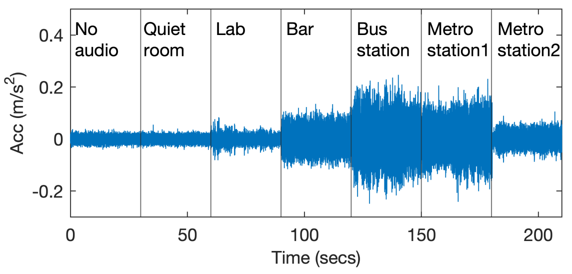
(a) Audio response distribution (b) Samsung S8

(c) Google Pixel 3 (d) Huawei P20

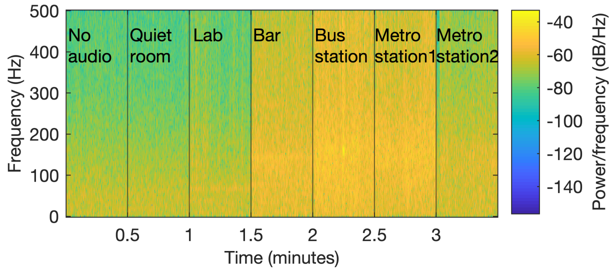
Fig. 5. Resonant frequency search. Fig. 5(a) presents the audio response of different smartphone accelerometers to airborne voice signals. The airborne signal is a series of single tone signals from 1000Hz to 22000Hz with an interval of 50Hz. Fig. 5(b), 5(c), and 5(d) respectively presents the scalogram of the accelerometer signal for the corresponding smartphone to achieve the highest audio response.

Acoustic noise refers to all the unwanted sound signals in the surrounding environment, which could be from speech, music, vehicles, industrial sources, etc. For the attack proposed in this paper, the acoustic noise mainly comes from the surroundings of the victim smartphone and the noise components of the played audio, that is, the noise around the remote caller.

For the acoustic noise around the victim smartphone, the sound will travel through the air to arrive the accelerometer. In the literature, Anand et al. [5] show that airborne *speech signals* are unable to have any noticeable impact on accelerometer measurements even at high sound pressure level. To study the impact of other noise signals, we first bring the smartphone to three noisy environments (bar, crowded bus station, and metro station) and collect accelerometer measurements for 30 seconds. For all three environments, we observe no significant impact on the accelerometer measurements and the AR_{dB} values along the z_axis are only 0.1900, 0.0724, and -0.0431, respectively. We then evaluate the impact of resonant frequency. The resonant frequency of the accelerometer's mass-spring system typically lies in the range from a few kHz [26] to tens of kHz [4]. According to the literature [40], [14], [39], airborne audio signals around the resonant frequency of a MEMS motion sensor can affect its measurements at high sound pressure level. To find out the impact of resonant frequency on our system, we test the audio response of Samsung S8, Google Pixel 3, and Huawei P20 with airborne audio signals within the normal frequency range from 1000Hz to 22000Hz. For each smartphone under investigation, we stimulate its accelerometer with the speaker of a Huawei



(a) Time domain



(b) Frequency domain

Fig. 6. The impact of acoustic noise around the remote caller. We eliminate the DC component of the acceleration signals and concatenate them for better comparison. The first segment is the self-noise of the accelerometer.

Mate 20 configured at its *highest volume*. The speaker and the accelerometer are placed on two different tables at a distance of 10 centimeters in order to eliminate the impact of surface vibration and maximize the sound pressure on the accelerometer. The audio signal is a series of two-second single tone signals ranging from 1000Hz to 22000Hz with a step frequency of 50Hz. We calculate the accelerometer's audio response at each frequency and plot the distribution of the obtained AR_{dB} values (Fig. 5(a)). The resulting AR_{dB} values seem to be normally distributed for each smartphone and have no notable outlier. Most of the recorded acceleration signals have an AR_{dB} below three. Samsung S8, Google Pixel 3, and Huawei P20 respectively achieve the highest AR_{dB} value at 4150 Hz (z-axis), 9450Hz (z-axis), and 11450Hz (x-axis). Fig. 5(b), 5(c), and 5(d) show the scalograms of the acceleration signals recorded at these frequencies. For Samsung S8 and Google Pixel 3, the accelerometers do not have a constant response at any specific frequency, which indicates that the high AR_{dB} values are caused by the variation of environmental vibrations. For Huawei P20, its accelerometer seems to have a constant but weak response at 20Hz. We repeat the experiment with the same stimulation signal for 10 times and the response has not been successfully reproduced, which indicates that the AR_{dB} value is caused by environmental vibrations. Based on these experimental results, it can be concluded that airborne acoustic noises at regular frequency (below 22000Hz) and sound pressure level are unlikely to distort the accelerometer measurements. The proposed attack will not be affected by the acoustic noise around the victim smartphone.

For the acoustic noise around the remote caller, because the noise signals will enter the caller's smartphone and be played by the victim device, it is likely that the accelerometer of the victim smartphone will be affected. To study the impact of such noise, we set the victim smartphone at its highest

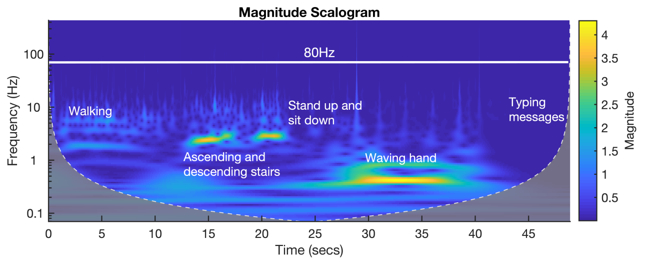
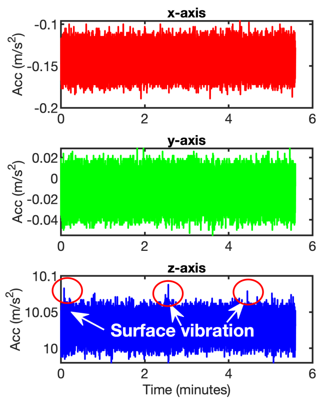


Fig. 7. The response of a smartphone accelerometer to five human activities. The obtained acceleration signals are concatenated for better comparison.

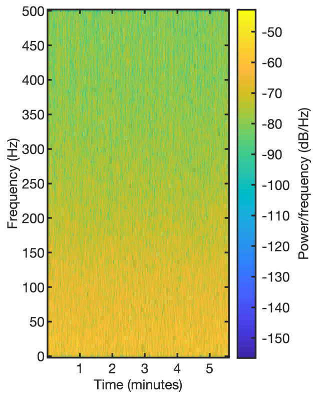
volume and make phone calls to volunteers in six realistic environments with different noise levels: 1) Quiet room, 2) Lab with people talking, 3) Bar with music playing, 4) Crowded bus station, 5) Metro station with trains running, 6) Metro station without trains running. For each environment, we collect accelerometer measurements for 30 seconds and calculate the average AR_{db} along the z-axis. The AR_{db} obtained from the six environments are -0.85, 1.67, 9.15, 13.87, 12.18, 4.89. Fig. 6 plots the time and frequency domain of the collected accelerometer measurements. It can be observed that the noise signals in environment 3, 4, and 5 have significantly affected all frequency components of the accelerometer measurements. The noise signals in environment 1, 2, and 6 have less effect on the accelerometer measurements and mainly affect the low frequency band of the acceleration signal, and thus can be significantly suppressed through a high-pass filter. Because the acoustic noise around the remote caller can significantly affect the accelerometer measurements, we evaluate its impact on speech recognition in Section VI-C.

Human activities can significantly affect the measurements of smartphone accelerometers and thus might be able to distort the speech information inferred by the adversary. In order to evaluate the impact of human activities, we studied the accelerometer's response to five human activities: walking, ascending and descending stairs, stand up and sit down, waving hand, and typing messages. During each test, the user hold a Samsung S8 with AccDataRec running in the background and conduct the activity for about 10 seconds. Because the accelerometer exhibits very similar response along the three axes, we concatenate the obtained acceleration signals and display the scalogram of the y-axis in Fig. 7. It can be observed that each of the tested activity produces a relatively unique and constant pattern in the acceleration signals. *However, none of these activities has significant impact on frequency components above 80 Hz.* As the typical fundamental frequency for adult speech signals is from 85Hz to 255Hz, a high-pass filter with 80Hz cut-off frequency can eliminate most of the distortions incurred by human activities (as shown in Fig. 7). The remaining distortions mainly exist as very short-time pulses in the high frequency domain, which actually has little impact on recognition/reconstruction according to our observation, but will influence the way of segmentation as introduced in V-A.

Self-noise and surface vibration: self-noise refers to the noise signal output by the smartphone's accelerometer at no external stimulus. This noise is an inherent feature of the accelerometer itself and contributes to the primary noise components of the proposed system. Because it is almost impossible to keep the accelerometer in a state without any



(a) Acceleration signal along the x-axis, y-axis, and z-axis



(b) Spectrogram of the signal along the z-axis

Fig. 8. The impact of self-noise and surface vibration. The accelerometer is placed on a table and is only affected by the vibration of the surface.

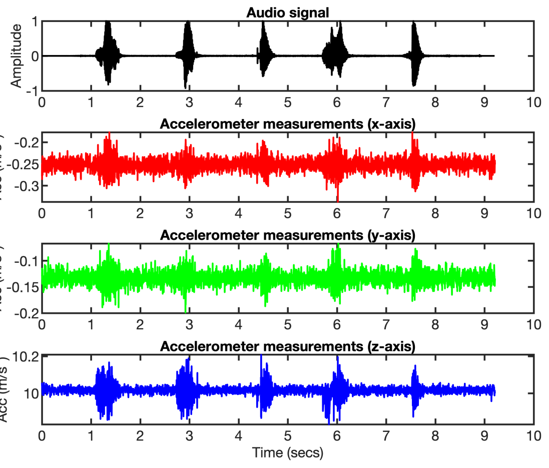
external stimulus, we investigate the combined effect of self-noise and surface vibration. Surface vibration could affect the accelerometer's measurements along the z-axis when the smartphone is placed on a table. To measure the impact of these two noise sources, we place a Samsung S8 on a table and record its accelerometer measurements for 330 seconds. The table has a solid surface that could effectively hand over vibration to the smartphone and is placed in a building under construction. The output signal of the accelerometer is depicted in Fig. 8(a). It can be observed that the accelerometer has a constant noise output along the x-axis and the y-axis. The self-noise of the accelerometer contributes to the majority of these noise outputs. For the z-axis, the accelerometer outputs a constant noise signal as well as surface vibrations. The frequency distribution of the acceleration signal along three axes are similar. For illustration, Fig. 8(b) plots the spectrogram of the signal along the z-axis (with the DC offset removed). In this spectrogram, around 57% of the energy are distributed below 80Hz. Because the typical fundamental frequency for adult speech is from 85Hz to 255Hz, we address the self-noise and the surface vibration through eliminating frequency components below 80 Hz. The impact of the remaining noise signal will be evaluated in section VI.

V. The Proposed System

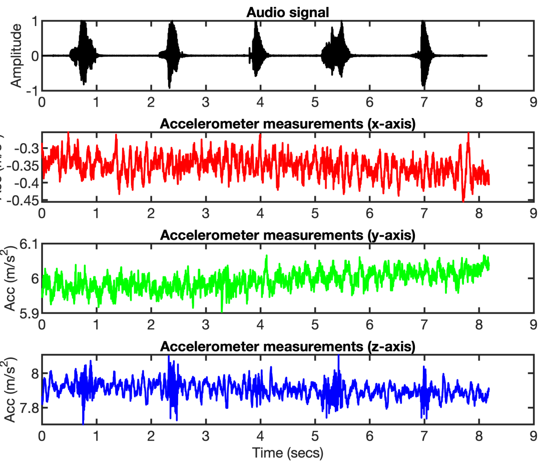
In this section, we will detail our proposed system, which mainly includes three modules, i.e. the preprocessing module, the recognition module, and the reconstruction module.

A. *Preprocessing*

The main intent of our system is to recognize and reconstruct the speech information captured by smartphone accelerometers. Compared with analyzing raw waveform data, a more prevalent and elegant way to recognize speech signals is to analyze their spectrogram representations [23], [8]. Such representation shows the frequency components of the signal and how their intensities change overtime. In conventional audio signal based speech recognition tasks, the spectrogram is normally further processed on a Mel-scale to calculate the Mel-Frequency Cepstrum Coefficients (MFCCs). This is because



(a) Table setting

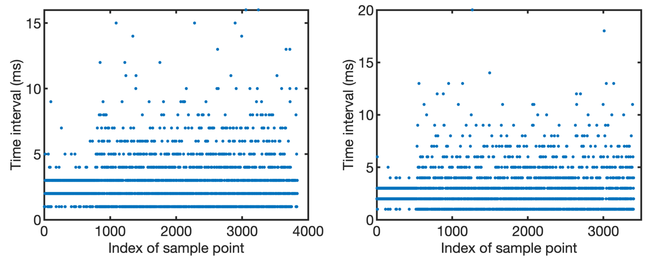


(b) Handhold setting

Fig. 9. Raw acceleration signals captured by smartphone accelerometer.

the Mel-scale mimics the non-linear perception characteristics of human ear and is good for discarding redundant and superfluous information. In our system, however, Mel-scale representations are of little help as the accelerometers in modern smartphones can only pick up speech signals in the low frequency band. Therefore, in the proposed system, we pre-process the acceleration signals into spectrograms for speech recognition and reconstruction. The spectrogram representation explicitly reflects the multi-scale information of a signal in the frequency domain, and enables the employment of some network structures widely-used in the computer vision tasks, such as ResNet & DenseNet.

Without loss of generality, we now use a Samsung S8 to help illustrate how spectrograms are generated from raw acceleration measurements. Fig. 9(a) and Fig. 9(b) show the raw acceleration signals collected from two different settings. In the table setting, we place the smartphone on a table and play a speech signal of five isolated digits (from zero to four) through its loudspeaker. Acceleration signals collected from this setting show strong audio response along all axes. For the handhold setting, we play the same speech signal with the smartphone held in hand. The acceleration signals are severely distorted due to the unintentional movement of the



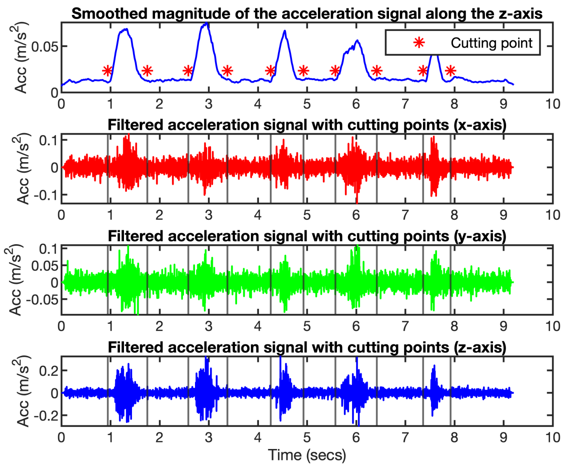
(a) Table setting (b) Handhold setting

Fig. 10. The time interval between two adjacent accelerometer readings. Raw accelerometer readings have unstable sampling interval because the system is configured to send measurements to the application as fast as possible.

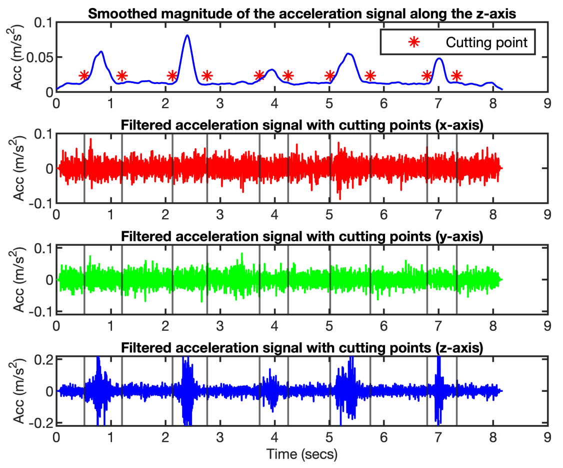
hand. There are three main problems in the raw acceleration signals: 1) Because the system is configured to send accelerometer measurements to the application as fast as possible, raw accelerometer measurements are not sampled at fixed interval (Fig.10). 2) Raw accelerometer measurements can be severely distorted by human movement. 3) Raw accelerometer measurements have captured multiple digits and needs to be segmented. *To address these problems, we use the following steps to convert raw acceleration signals to spectrograms.*

Interpolation: We first use linear interpolation to deal with unstable intervals of accelerometer measurements. Because the timestamps of the sensor measurements are with millisecond accuracy, a natural way to resolve unstable intervals is to upsample the accelerometer measurements to 1000 Hz. Therefore, we use timestamps to locate all time points that have no accelerometer measurement and use linear interpolation to fill in the missing data. The resulting signal has a fixed sampling rate of 1000 Hz. We note that this interpolation (upsampling) process does not increase the speech information of the acceleration signal. Its primary purpose is to generate acceleration signals with a fixed sampling rate.

High-pass filtering: We then use a high-pass filter to eliminate significant distortions caused by gravity, hardware distortion (offset error), and human activities. In particular, we first convert the acceleration signal along each axis to the frequency domain using the Short-Time Fourier Transform (STFT). It divides the long signal into equal-length segments (with overlaps) and calculates the Fourier transform on each segment separately. We then set the coefficients of all frequency components below the cut-off frequency to zero and convert the signal back to the time domain using inverse STFT. Because the fundamental frequency for adult male and female is usually higher than 85 Hz, and human activities rarely affect frequency components above 80 Hz (as shown in Fig. 7), the cut-off frequency for speech recognition is set to 80Hz so that the impact of noise components can be minimized. For speech reconstruction, since the reconstruction network mainly learns the mapping between acceleration signals and audio signals, we use a cut-off frequency of 20 Hz in order to preserve more speech information. Fig. 11(b) and 11(a) display the filtered acceleration signals with a cut-off frequency of 20 Hz. For the table setting, all the acceleration signals are shifted to zero mean, which indicates the successful removal of the offset error and the gravity (for the z-axis). For the acceleration signals collected under the handhold setting, the high-pass filter has



(a) Table setting

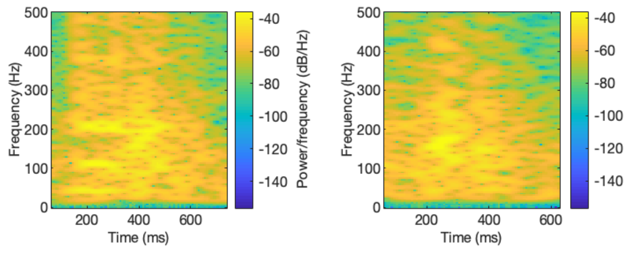


(b) Handhold setting

Fig. 11. Acceleration signals processed with interpolation, high-pass filtering, and segmentation. The cut-off frequency for high-pass filtering is 20 Hz. The magnitude sequences are directly calculated from the filtered signals since they do not involve intense human movements.

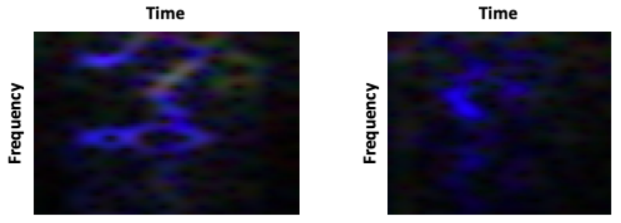
also eliminated the impact of hand movement. The filtered signals obtained after this step mainly consist of the target speech information and the self-noise of the accelerometer.

Segmentation: As the acceleration signals along the three axes are completely synchronized, we use the dominant axis (z-axis) described in section IV to locate cutting points and then use the obtained cutting points to segment the filtered acceleration signal along three axes. The cutting points are located as follows: Given the acceleration signal along the dominant axis, we first sanitize the signal through another round of high-pass filtering with a cut-off frequency of 160 Hz. Through studying signals collected from noisy experimental setups, we observe that this cut-off-frequency could eliminate a significant amount of noise components including the short-time pulses caused by human movement. *Typically, this process is needed only when the smartphone experiences external vibrations or intense movement.* We then calculate the magnitude (absolute value) of the sanitized signal and smooth the obtained magnitude sequence with two rounds of moving average. The sliding window for the first round and the second round are 200 and 30 respectively. The smoothed magnitude sequences



(a) Table setting (b) Handhold setting

Fig. 12. The spectrogram of the first single word signal (z-axis).



(a) Table setting (b) Handhold setting

Fig. 13. The spectrogram-image of the first single-word signal. These images cover the frequencies from 80Hz to 300Hz.

of the two settings are shown in Fig. 11. Next, we find the maximum value M_{max} and the minimum value M_{min} of the smoothed magnitude sequence. In this process, the first and last 100 magnitude values are discarded as they do not have sufficient neighboring samples for averaging. The obtained minimum value is approximately the magnitude of the noise signal. After that, we traverse through the smoothed magnitude sequence and locate all the regions with magnitudes higher than a threshold of $0.8M_{min} + 0.2M_{max}$. Each located region indicates the existence of a speech signal. In order to make sure that the segmented signal will cover the whole speech signal, the start and end points of each located region are then moved forward and backward by 100 and 200 samples, respectively. The cutting points calculated from each setting are marked in Fig. 11. Finally, we use the obtained cutting points to segment the filtered acceleration signal into multiple short signals, each of which corresponds to a single word.

Signal-to-spectrogram conversion: To generate the spectrogram of a single-word signal, we first divide the signal into multiple short segments with a fixed overlap. The lengths of the segment and the overlap are set as 128 and 120 respectively. We then window each segment with a Hamming window and calculate its spectrum through STFT, which generates a series of complex coefficients for each segment. The signal along each axis is now converted into a STFT matrix that records the magnitude and phase for each time and frequency. Finally, the 2D spectrogram can be calculated through

$$spectrogram\{x(n)\}(m, w) = |STFT\{x(n)\}(m, w)|^2, \quad (1)$$

where $x(n)$ and $|STFT\{x(n)\}(m, w)|$ respectively represents a single-axis acceleration signal and the magnitude of its corresponding STFT matrix. Because we have acceleration signals along three axes, three spectrograms can be obtained for each single-word signal. For illustration, Fig. 12 plot the spectrogram (z-axis) of the first single-word signal of each setting. The frequency components below 20 Hz are close to zero due to the high-pass filtering process.

Spectrogram-Images: To directly feed the spectrograms into the neural networks used in computer vision tasks, we further convert the three 2-D spectrograms of a signal into one RGB image in PNG format. To do so, we first fit the three $m \times n$ spectrograms into one $m \times n \times 3$ tensor. Then we take the *square root* of all the elements in the tensor and map the obtained values to integers between 0 and 255. The reason of taking the square root is that the majority of the elements in the original 2-D spectrograms are very close to zero. Directly mapping these elements to integers between 0 and 255 will result in considerable information loss. Finally, we export the $m \times n \times 3$ tensor as an image in PNG format. In the obtained spectrogram-image, the red, green, and blue channels correspond to the x-axis, y-axis, and z-axis of the acceleration signal, respectively. For the recognition task, the spectrogram-images are cropped to the frequency range from 80 Hz to 300 Hz in order to reduce the impact of self-noise. Fig. 13 plots the spectrogram-image of the first single-word signal of each setting. A brighter region indicates that the acceleration signal has stronger energy at that frequency range during that time period. It can be observed that the blue channel (the acceleration signal along the z-axis) provides the most speech information under both settings.

B. Recognition

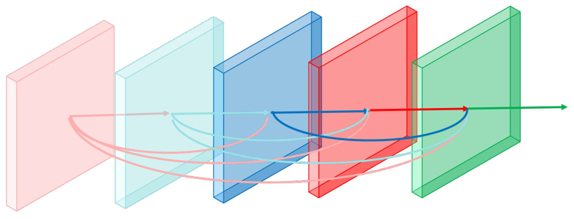
With the above preprocessing operations, the resulting acceleration spectrogram-images can be fed into various standardized neural networks such as VGG [36], ResNet [22], Wide-ResNet [43], and DenseNet [24] after resizing. We now detail the design of our recognition module.

Spectrogram-image Resizing To feed those spectrogram-images into standardized computer vision networks, it is better to resize them into $n \times n \times 3$ images. Note that fine-grained information and correlations of the acceleration-spectrograms may be influential to the recognition results, especially the results of speaker identification. To preserve sufficient information, we resize the spectrogram-images into $224 \times 224 \times 3$.

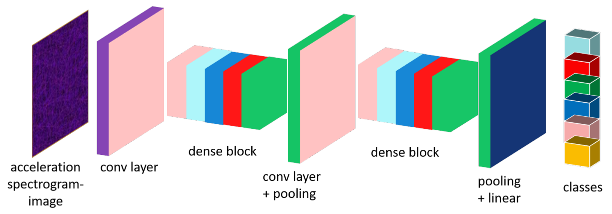
Network Choice Generally, we choose DenseNet as the base network for all our recognition tasks. Compared with traditional deep networks like VGG and ResNet, DenseNet introduces connections between each layer and all its preceding layers, i.e., totally $\frac{(L+1)L}{2}$ connections in an L-layer network. For instance, as shown in the diagram of a common block in DenseNet (Fig. 14(a)), the first to fourth layers all have direct links to the fifth layer. In another word, l-th takes the concatenation of feature maps from 0-th layer (input image) to the $(l-1)$-th layer as input, which can be mathematically represented by

$$\mathbf{x}_l = \mathcal{H}_l([\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_{l-1}]). \quad (2)$$

$\mathcal{H}_l$ and $\mathbf{x}_l$ denote the function and the feature map of the l-th layer, respectively. $[\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_{l-1}]$ denote the concatenation of the feature maps of the 0-th layer to the $l-1$-th layer. These direct connections enable all layers to receive and reuse the features from their preceding layers, and thus, DenseNet does not have to use some redundant parameters

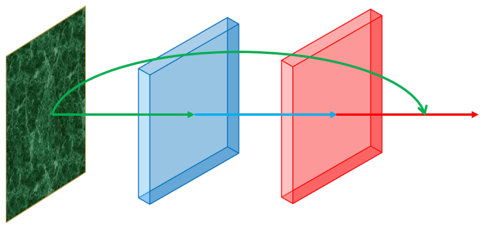


(a) Diagram of Dense Block

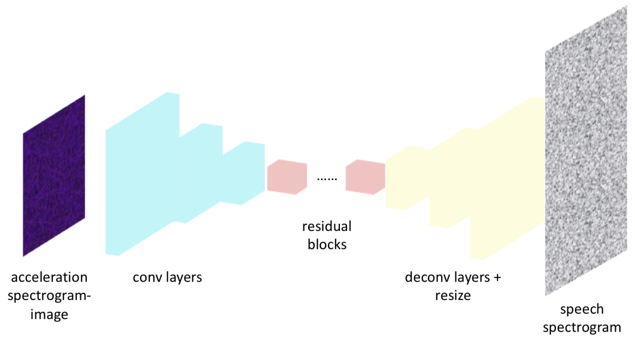


(b) Diagram of DenseNet

Fig. 14. The network structure of DenseNet. The connections are established between every layer and its previous layers to improve information flow.



(a) Diagram of Residual Block



(b) Diagram of Reconstruction Network

Fig. 15. The network structure of our reconstruction network. It will much easier to optimize residual mappings in (3) by residual blocks.

or nodes to maintain the information from the previous layers. Therefore, DenseNet can use fewer nodes (parameters) to achieve performance that is comparable with VGG and ResNet. Moreover, improved flow of information and gradients throughout the network also alleviates gradient-vanishing and makes DenseNet easier to train. Empirically, we found that in our recognition tasks, DenseNet indeed achieves the best accuracy with fewer parameters and less computational cost (compared with VGG and ResNet). Fig. 14(b) demonstrates the overall network structure we utilize, which consists of multiple dense blocks shown in Fig. 14(a).

Training Process In the training stage, we use the cross-entropy as the training loss and optimize the model weights by a piecewise momentum optimizer to learn more generalizable features and also facilitate convergence. Specifically, the adaptive momentum optimization process is first executed by a large step size (e.g., 0.1) to learn generalizable features and then fine-tuned by smaller step sizes to facilitate convergence. We also add weight decay into the training loss and set the dropout rate as 0.3 to enhance generalizability.

C. Reconstruction

Except for recognition, reconstruction of the speech signals from the corresponding acceleration signals (spectrograms) is also one function we want to include in our proposed system, since this function can be used for double-checking our recognition results. *Note that although accelerometers in current smartphones can only pick up the low frequency components, many components in the high frequency band are mainly the harmonics of these fundamental frequency components, which makes it possible for us to reconstruct speech signals with enhanced sampling rates from the corresponding acceleration signals.* To achieve speech-signal reconstruction, we first reconstruct the speech-spectrograms by the following reconstruction network, with the acceleration spectrogram-images as input. Then the speech signals are estimated from the reconstructed speech-spectrograms by the Griffin-Lim algorithm proposed in [20]. Next, we will detail the reconstruction network & the speech signal estimation method.

1) Reconstruction Network: The reconstruction network consists of three sub-networks, i.e., an encoder, residual blocks, and a decoder. The input of the reconstruction network is a $128 \times 128 \times 3$ spectrogram-image which covers the frequency components from 20 to 500 Hz. Each channel corresponds to one axis of the acceleration signal. However, a problem of standardizing the input size here is that the acceleration signals may have different time-lengths due to the various time-lengths of speech signals from different speakers, *but resizing the spectrogram-images is not desired here* since the time-scale information is better to be maintained in the reconstruction process. A simple solution is to repeat the speech signal until it reaches a pre-defined time-length. This solution is valid because the reconstruction task enforces no restriction on the content of the speech/acceleration signals (spectrograms), unlike the above recognition module whose input has to be a single word spectrogram-image. The output of the reconstruction network is a 384×128 gray image that represents the corresponding speech-spectrogram since the speech signal only has one axis. Due to the limited sampling rate of the accelerometer, our reconstruction network only aims at reconstructing the frequency components of the speech signal from 0 to 1500 Hz.

Encoder The first sub-network is an encoder for encoding the acceleration spectrogram-images (*i.e.,* conv layers in Fig. 15(b)). The encoder starts with a convolutional layer with 32 kernels of size $9 \times 9 \times 3$ to learn the large-scale features, followed by two convolutional layers with 64 kernels of size $3 \times 3 \times 32$ and 128 kernels of size $3 \times 3 \times 64$ respectively to learn the small-scale features. Besides, a stride of 2 is applied on the first two layers for downsampling.

Residual Blocks Inspired by the architecture of [25], we add five residual blocks (as shown in Fig. 15(a)) after the encoder to explicitly let the features fit a residual mapping

$\mathcal{H}(\cdot)$, i.e.,

$$\mathcal{H}(\mathbf{x}) = \mathcal{F}(\mathbf{x}, \mathbf{W}_i) + \mathbf{x}, \tag{3}$$

where $\mathcal{F}(\mathbf{x}, \mathbf{W}_i)$ are the nonlinear mappings learned by convolutional layers. Considering the structural similarity between the spectrograms of the acceleration and speech signals, it is very likely that identity mapping is an optimal/near optimal mapping to establish some feature-connections. When the optimal mapping is or close to an identity function, it is easier to optimize $\mathcal{H}$ than an unreferenced block $\mathcal{F}$. This is because pushing the parameters of $\mathcal{F}$ into zero should be easier than optimizing $\mathcal{F}$ as an identity mapping. Therefore, we add a number of residual blocks $\mathcal{H}$ in the middle of our reconstruction network (*i.e.,* residual blocks in Fig. 15(b)).

Decoder Finally, the speech-spectrograms are decoded from the features learned by the encoder & residual blocks by a decoder (*i.e.,* deconv layers in Fig. 15(b)). The decoder also consists of 3 deconvolutional layers, with respectively 64 kernels of size $3 \times 3 \times 128$, 32 kernels of size $3 \times 3 \times 64$, and 3 kernels of size $9 \times 9 \times 32$. A stride of 1/2 is applied on the first two layers for upsampling. The initial output of the decoder is a $128 \times 128 \times 3$ matrix, and the matrix will be further resized into a 384×128 gray image to represent the corresponding speech-spectrogram as mentioned before.

Training Process Compared with recognition, reconstruction is a task whose training process is more **unstable** and **computationally expensive**. The instability issue is probably caused by the sparse outliers in a training minibatch, due to the sparsity of spectrograms. *To fix it, we employ the L_1 distance between the reconstructed images and the targeted images as the training loss instead of the MSE loss in [41]. This is because L_1 loss is more robust than the MSE loss to outliers [11].* Besides, We also apply a weight decay on the L_1 loss to enhance generalizability. To reduce the computational cost, we accelerate the optimization process by applying a time-based decay to the learning rate. Specifically, we use a momentum optimizer with a learning-rate scheduler that decays the learning rate by a factor of 0.9 per training epoch.

2) Speech Signal Estimation: Griffin-Lim algorithm is an iterative algorithm for signal estimation from spectrograms. And each iteration contains two steps: the first step is to modify the STFT of the current signal estimation by the spectrogram; The second step is to update the current signal estimation by the modified STFT. Next, we will detail the two steps.

Modify STFT Given the current estimation of the speech signal $x^i[n]$ in the ith iteration and the reconstructed magnitude (square root of spectrogram) $\|Y(m,w)\|$, the STFT $X^i(m,w)$ of $x^i[n]$ is modified as

$$\hat{X}^{i+1}(m,w) = X^i(m,w)\frac{\|Y(m,w)\|}{\|X^i(m,w)\|}, \tag{4}$$

to ensure the magnitude of the modified STFT $\hat{X}^{i+1}(m,w)$ to be same as the reconstructed magnitude $\|Y(m,w)\|$.

Update Signal Estimation Note that the modified STFT $\hat{X}^{i+1}(m,w)$ may be not a valid STFT if there is no signal whose STFT is given by $\hat{X}^{i+1}(m,w)$. In this sense, we prefer to find a sequence $x^{i+1}(n)$ whose STFT $X^{i+1}(m,w)$ is the closest to the modified STFT $\hat{X}^{i+1}(m,w)$ by minimizing the following mean square error between $X^{i+1}(m,w)$ and $\hat{X}^{i+1}(m,w)$,

$$\sum_{m=-\infty}^{+\infty}\sum_{w=-\infty}^{+\infty}[X^{i+1}(m,w) - \hat{X}^{i+1}(m,w)]^2. \tag{5}$$

The solution to the above minimization problem is

$$x^{i+1}(n) = \frac{\sum_{m=-\infty}^{+\infty} w(n-mS)\hat{x}^{i+1}(m,w)}{\sum_{m=-\infty}^{+\infty} w^2(n-mS)}, \tag{6}$$

where $\hat{x}^{i+1}(m,w) = \frac{1}{2\pi}\int_{w=-\pi}^{\pi}\hat{X}^{i+1}(m,w)e^{-jwn}dw$ and S refers to the sampling period w.r.t. n. These two steps are iterated for multiple steps until convergence, and the final $x^i(n)$ is output as the estimation of the speech signal.

VI. Implementation and Evaluation

A. Experimental setup and Datasets

We mainly evaluate our proposed system on accelerometer measurements collected from a Samsung S8. The scalability of the proposed model is evaluated in section VI-D. *For each specific setting, we play a series of speech signals on the smartphone and collect accelerometer readings through the third party Android application AccDataRec running in the background.*

The speech signals are mainly from two datasets. The first dataset consists of 10k single-digit signals from 20 speakers, which are from the AudioMNIST dataset[2]. The signals in this dataset are concatenated into long audio signals with an interval of 0.1 seconds in order to simulate the scenario where the victim is telling others his/her passwords. The second dataset consists of 36×260 digits+letters speech signals collected from volunteers. We hire volunteers from the university and collect data in the lab. The volunteers were asked to hold the smartphone and read a long series of digits and letters with the speech rate they would use for telling others their passwords. There are totally 36 classes including 10 digits (0-9) plus 26 letters (A-Z), and each class contains 260 samples collected from 10 speakers. We collect accelerometer readings from these two speech sources and evaluate the proposed system under different settings. ***We note that all the experimental results presented in this paper are user-independent.*** *For each setting under investigation, we randomly divide all collected signals into* 80% *training data and* 20% *testing data. In the following, we only report the testing accuracy.*

B. Recognition

As described earlier, the state-of-the-art (SOTA) model [32] uses the gyroscope on smartphones to capture speech signals emitted by a loudspeaker placed on the same solid surface. To make a fair comparison, we first evaluate the performance of our model in a similar setting – i.e., place the smartphone on a table. Table V lists the top1, top3, top5 (testing) accuracy of our system in digit-recognition, digit+letter-recognition, and speaker-identification. Top N accuracy is the probability that the correct label is within the top N classes predicted by our network. Remarkably, the top1 accuracy of our model on digit-recognition in the user-independent setting even surpasses

[2]https://github.com/soerenab/AudioMNIST

the SOTA accuracy in the user-dependent setting by 13%. Our system also achieves 55% top1 accuracy and 87% top5 accuracy on recognizing 10 digits plus 26 letters (totally 36 classes). In terms of speaker-identification, our system achieves 70% accuracy on classifying 20 speakers, while the previous SOTA model only has 50% accuracy on classifying 10 speakers. Overall, our model achieves new SOTA results in all the tasks. The increase of accuracy is not only because of the usage of an advanced model, but also because of the increased sampling rate and the proposed setup. Our setup allows the voice signal to have a much more significant impact on the motion sensor, and thus the SNR of the acceleration signal is significantly improved compared with the SOTA setup. As will be shown in table VII and XII, the recognition accuracy increases smoothly with the SNR of the acceleration signal and the sampling rate of the accelerometer.

TABLE V. COMPARISON BETWEEN OUR RESULTS AND STATE-OF-THE-ART (SOTA) RESULTS FROM [32].

Tasks	Our model (DenseNet)			SOTA (user dependent)	SOTA (user independent)
	top1 acc	top3 acc	top5 acc		
Digits	78%	96%	99%	65%	26%
Digits + Letters	55%	78%	87%	-	-
Speakers	70% (20)	88% (20)	95% (20)	50% (10 speakers)	

In addition to the table setting, our system is also applicable to other settings, e.g., a more common scenario – the smartphone is held in a user's hand. Compared with the table setting, an accelerometer in the handhold setting will exhibit lower SNR along the x-axis and y-axis. Therefore, more attention (weight) should be allocated to the z-axis. In Table VI, we show the testing accuracy of our model in the "Table" and "Hand-hold" settings. As shown in Table VI, if our model is only trained by either the "Table" or "Hand-hold" training set, it only has no more than 20% on the other testing set, due to the aforementioned difference between those two settings. However, if we train the model on both the "Table" and "Hand-hold" training sets, the accuracy will be improved to above 60% on both testing sets.

TABLE VI. TOP 1 ACCURACY OF OUR MODEL UNDER DIFFERENT SETTINGS (ALL THE CASES BELOW ARE USER INDEPENDENT).

Train (80%) \ Test (20%)	Table	Handhold	Table + Handhold
Table	78%	17%	47%
Handhold	19%	77%	48%
Table + Handhold	69%	63%	66%

C. *The impact of noise:*

As discussed in section IV-C, the proposed attack might be affected by the self noise of the accelerometer and the acoustic noise around the remote caller.

For the self-noise of the accelerometer, although this noise component has decreasing power across the whole frequency band, it may still weaken the features in the acceleration signals and thus reduce the recognition accuracy. To test the robustness of our recognition model against this self-noise, we utilize the white Gaussion noise to simulate this noise and generate acceleration signals with different SNRs. The resulting signals simulate accelerometer measurements collected at lower volume levels. The results for digit-recognition and speaker-identification are shown in Table VII. Although the accuracy is reduced as the SNR decreases, our system is actually very robust in the sense that the accuracy of digit-recognition on the SNR=2 data even surpasses the previous SOTA accuracy on the clean data.

TABLE VII. PERFORMANCE OF OUR RECOGNITION MODEL ON NOISY ACCELERATION SIGNALS (SPECTROGRAMS).

SNR \ Tasks	Digit-recognition (0-9)		Speaker-identification (20 speakers)	
	top1 acc	top3 acc	top1 acc	top3 acc
SNR = 2	42%	73%	34%	64%
SNR = 4	52%	82%	43%	71%
SNR = 6	61%	87%	51%	77%
SNR = 8	66%	91%	58%	81%

For the acoustic noise around the remote caller, we hire four volunteers (two females and two males) and ask them to send voice messages to the victim smartphone from four realistic environments with different noise levels: 1) No-noise (quiet room). 2) Low-noise (lab with people talking). 3) Medium-noise (bar with music playing). 4) High-noise (crowded bus station). These environments are selected based on the experimental results in Fig. 6. We then play the received voice messages under the table setting and record the accelerometer measurements. The dataset for each environment contains 200 $\times$ 10 digits spectrograms collected from four speakers. Table VIII lists the results for digit-recognition. Surprisingly, the recognition model achieves over 80% accuracy in the first three environments. For the high-noise environment, the recognition accuracy is greatly decreased because the segmentation algorithm can hardly distinguish between speech signals and sudden loud noise. In order to find out if the recognition model can recognize well-segmented high noise signals, we manually adjust the segmentation of the signal and repeat the experiment. With manually segmented signals, our model achieves 78% top 1 accuracy in the high-noise environment, which suggests that the recognition model is very robust against ambient acoustic noise. Since the proposed attack can achieve high accuracy in most environments and few people would make phone calls in a high-noise environment, we believe that the proposed attack is practical.

TABLE VIII. RECOGNITION ACCURACY UNDER REALISTIC NOISY ENVIRONMENTS. *The recognition accuracy in the no-noise environment is slightly higher than the accuracy in table V because this experiment only involves four speakers (two females and two males).*

Noise level	top 1 acc	top3 acc	top5 acc
No-noise	86%	97%	100%
Low-noise	86%	98%	99%
Medium-noise	80%	96%	99%
High-noise	47%	73%	88%

D. *Scalability study:*

Different smartphones may have different sampling rates and dominant axes, which makes it difficult to generalize a recognition model trained from a smartphone to other smartphone models. To study the scalability of the proposed attack, we collect acceleration signals from six smartphones of three different models: 1) Samsung S8: the sampling rate is 420 Hz and the dominant axis is the Z-axis. 2) Huawei Mate 20: the sampling rate is 500 Hz and the dominant axis is the Z-axis. 3) Oppo R17: the sampling rate is 410 Hz and the smartphone has similar audio response across three axes. We collect 10k digits acceleration signals from each smartphone model and evaluate the possibility of deploying one model globally. We observe that the acceleration signals collected from Huawei Mate 20 and Oppo R17 have much less noise signals than the Samsung S8. As shown in Table IX, it is not easy for a recognition model trained by data from a smartphone model to generalize to other smartphones, due to the diverse hardware features of different smartphone models. However, we still observe a 5% accuracy increase for the Samsung S8 when the recognition model is trained with data from both Oppo R17 and Huawei Mate 20. Therefore, we conjecture that the recognition model can be scalable to unseen smartphones, if the model is trained by the data from enough smartphone models that can capture the diversity of the hardware features. Besides, Table IX also indicates that the model capacity of our recognition model is adequate to fit the data from multiple smartphone models without loss in accuracy.

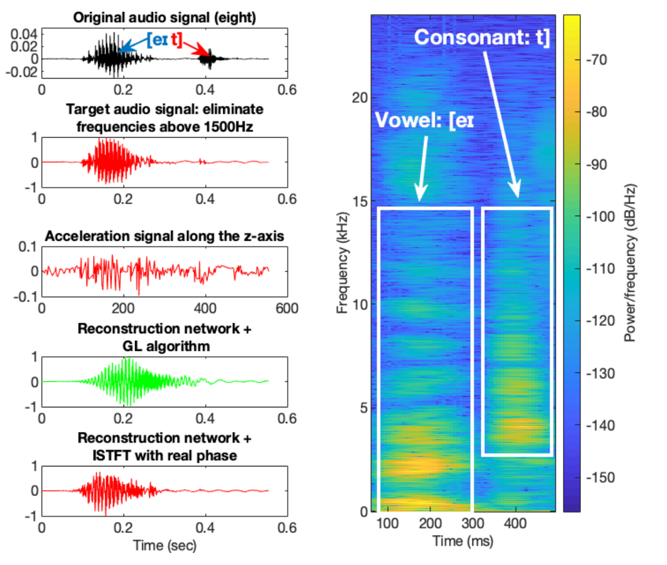
TABLE IX. MULTI-DEVICE TRAIN AND TEST (TOP 1 ACCURACY).

Test / Train	Samsung S8	Huawei Mate 20	Oppo R17
Samsung S8	80%	15%	20%
Huawei Mate 20	12%	82%	21%
Oppo R17	21%	23%	91%
Oppo R17, Huawei Mate 20	26%	83%	90%
Samsung S8, Oppo R17, Huawei Mate 20	79%	84%	90%

E. Reconstruction

The performance of our reconstruction network is evaluated by two metrics, i.e., the averaged testing ℓ_1 and mean square error. Given the reconstructed speech-spectrogram as $\tilde{\boldsymbol{x}}$ and the ground truth speech-spectrogram as $\boldsymbol{x}$, the ℓ_1 error can be calculated by $\sum_i |\tilde{\boldsymbol{x}}_i - \boldsymbol{x}_i|$, where i represents the index w.r.t. pixels. *The final testing ℓ_1 error is close to* $1e3$*, i.e., the absolute error per pixel is approximately* 0.02 *(pixel range is [-1, 1]).* The mean square error can be calculated by $\frac{\sum_i (\tilde{\boldsymbol{x}}_i - \boldsymbol{x}_i)^2}{N}$, where N is the number of pixels per image. The final testing mean square error is approximately 3.5e-3. These results indicate the ability of our reconstruction network to reconstruct speech-spectrograms from acceleration-spectrograms with very small errors.

We further use the Griffin-Lim (GL) algorithm to estimate the speech signals from the reconstructed spectrograms and demonstrate the results in Fig. 16(a). For comparison, we show the original speech signal in the first row. The second row shows the original speech signal but without frequency components higher than 1500Hz, which is actually the ground-truth (target) audio signal that we attempt to reconstruct. Although this frequency cut-off may lead to loss of certain consonant information, due to the limited frequency range of the acceleration signals, 1500Hz is almost the highest (harmonic) frequency that could be reconstructed for the speech signals here. The third row shows the raw acceleration signal, which has similar structures but completely different details compared to the cut-off audio signal, indicating reconstruction of speech signals from acceleration signals should be a complicated task. In the fourth row, we demonstrate the speech signals reconstructed by our reconstruction network and the GL algorithm, which already captures most structures



(a) Reconstructed signal (b) Spectrogram of eight

Fig. 16. Comparison between the target speech signal and the reconstructed one. The speech information is eight.

and details of the cut-off speech signals. We argue that the remaining difference between the reconstructed signal and the cut-off signal is mainly due to the errors caused by the GL algorithm, because if we simply apply the phases of the cut-off speech signal to the magnitudes (spectrograms) reconstructed by our reconstruction network, almost the same signal can be recovered as the cut-off audio signal, as shown in the fifth row.

To our knowledge, our reconstruction module is the first trial on reconstructing speech signals from acceleration signals, which is successful in the sense that most structures and details are roughly recovered. However, there still remains two limitations: One is that the largest (harmonic) frequency recovered by our reconstruction module is 1500Hz, which may lead to consonant information loss. The other limitation comes from the GL algorithm, which may not be the optimal choice to compensate the phase information. We will detail these two limitations and the potentials for improvement in Section VII.

F. Hot Words Search: Recognition and Reconstruction

In this subsection, we conduct an experiment to show that our models can also be used to search hot (sensitive) words from sentences. In this experiment, we first use a hot word search model to identify pre-trained hot words from sentences. We then use the reconstruction model to reconstruct the audio signal and to double check the identified hot words by human ears. The experiment is conducted with 200 short sentences collected from four volunteers (two males and two females). Each short sentence contains several insensitive words and one to three hot words listed in Table X.

The hot word search model is based on a recognition model that can distinguish between eight hot words (listed in Table X) and other insensitive words. To train this model, we collect a training dataset with 128*8 hot words and 2176 insensitive words (negative samples) from the volunteers. It can be observed that this dataset is class-imbalanced since the number of the samples in each hot-word class is far less than the number of the negative samples. To address this problem, we re-weight the losses for the nine classes. Since the total number of negative samples is 17 times the number of the samples in each hot-word class, we weight the loss computed by the hot-word samples with a factor 17α, and the loss computed by negative samples with a factor α. α is a hyper-parameter and is set to 0.1 in the training process. We then segment the acceleration signals of the test sentences into single word spectrograms and use the hot word search model to recognize them. As shown in Table X, our recognition model can achieve over 90% recognition accuracy averagely on those eight hotwords, which is slightly higher than the recognition accuracy on the 10 digits. We note that this is because there are only nine classes in this recognition task, and also, the spectrograms of these eight hot words are more distinctive in comparison with digits and letters.

TABLE X. TRUE POSITIVE RATE (TPR) AND FALSE POSITIVE RATE (FPR) FOR EACH HOT WORD.

Word	TPR	FPR	Double Check FPR
Password	94%	0.4%	0.2%
Username	97%	0.4%	0.3%
Social	100%	0.3%	0.0%
Security	91%	0.0%	0.0%
Number	88%	0.1%	0.0%
Email	88%	1.4%	0.8%
Credit	88%	0.3%	0.3%
Card	97%	1.4%	0.3%

We then implement a reconstruction model that can reconstruct *full-sentence audio signals* from acceleration signals. Because the reconstruction model mainly learns the mapping between signals rather than semantic information, it does not require signal segmentation and is more generalizable to unseen (untrained) data than the recognition model. To train such a model, we collect 6480 acceleration spectrograms and audio spectrograms with sentences different from the testing sentences. The resolutions of the acceleration spectrogram and the audio spectrogram are $128 \times 1280 \times 3$ and 384×1280 respectively, which allows the reconstruction model to reconstruct audio signals up to 12 seconds. We use this model plus the GL algorithm to reconstruct the audio signals of all test sentences and hire two volunteers to listen to them. In this process, we first provide basic training for each volunteer, where the volunteer will hear the audio signals of 20 sentences and their reconstructed versions. We then ask the volunteer to listen to reconstructed signals (sentences) and relabel the hot words *falsely* identified by the recognition model. In this process, the label of a hot word will not be changed unless both volunteers agree to change it. It turns out the volunteers can easily tell whether a hot word is falsely identified. The false positive rates for all hot words are reduced bellow 1% and the true positive rates are not changed. This is primarily because that listening to full sentences allow the adversary to leverage valuable contextual information.

G. End-to-end Case Study: Steal Passwords from Phone Conversations

We now evaluate the proposed models with an end-to-end attack in phone conversations. We consider a real-world scenario where the victim makes a phone call to a remote caller and requests a password during the conversation. The objective of the adversary is to locate and recognize the password

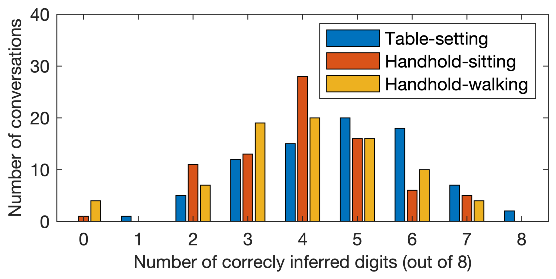


Fig. 17. The number of correctly inferred digits in each conversation.

from the victim's accelerometer measurements. In this attack, we assume that the password is preceded by the hot word "password (is)". As shown in section VI-F, this attack can be easily extended to support other hot words.

In the experiment, we use the victim smartphone to make phone calls to four volunteers (two females and two males) in three different scenarios: 1) *Table-setting*: the victim smartphone is placed on a table. 2) *Handhold-sitting*: the victim sits on a chair and holds the smartphone in hand. 3) *Handhold-walking*: the victim holds the smartphone in hand and walks around. We conduct 16 scripted conversations and 4 free ones per person per scenario (*i.e.,* 240 conversations in total). During all conversations, the volunteers are asked to tell us a random 8-digits password following the phrase "password is".

After recording the acceleration signals for a conversation, we first convert the acceleration signals into multiple single-word spectrograms and employ a password search model to find the spectrogram corresponding to the hot word "password". Then a digit recognition model is used to recognize the 8-digits password following it.

The password search model searches for the word "password" in the recorded acceleration signals by a classifier that can distinguish between "password" and the other words. To train such a classifier, we collect a training dataset with 200 "password"s and 2200 negative samples including digits and some other words. The class imbalance problem is also addressed through re-weighting the loss as in Section VI-F. Specifically, we weight the loss computed by the "password" samples with a factor 11α, and the loss computed by negative samples with a factor α. In the case that the model recognizes multiple "password"s in a conversation, the one with the highest confidence value will be reported. Using this model, we successfully locate the password for over 85% conversations in all scenarios, as shown in table XI.

TABLE XI. PERCENTAGE OF CORRECT DIGITS INFERRED BY EACH MODEL.

Setting	Password search	Digit recognition		
		top1	top3	top5
Table-setting	92%	59%	84%	92%
Handhold-sitting	85%	51%	83%	94%
Handhold-walking	91%	50%	81%	91%

The digit recognition model for each scenario is trained with 280×10 digits spectrograms. Table XI lists the overall accuracy of the recognition model. We also calculate the number of correctly inferred digits in each conversation and plot the distribution in Fig. 17. It can be observed that the recognition accuracy in the phone call scenario is lower compared with the record-and-play scenario. This is primarily because that the audio signal transmitted during a phone call has lower quality than the audio signal recorded by a recording application. One important observation is that the recognition model achieves over 80% top 3 recognition accuracy in all scenarios. Although the proposed attack only recognizes the complete password in a few conversations, it will greatly assist the adversary to narrow his search for the victim's password.

VII. Further Improvement

As mentioned at the end of Section VI-E, the performance of our reconstruction module is still limited by the frequency range of the acceleration signals and the veracity of the GL algorithm. The first limitation restricts the audio frequency range that could be reconstructed, thus the consonant information, which is mainly distributed in the high frequency domain as shown in Fig. 16(b), will lose. However, we believe that this limitation can be mitigated by the hardware improvement of smartphones in the future. The second limitation is probably due to the GL algorithm since this algorithm attempts to compensate all the phase information from nearly scratch. To improve the performance, we propose to involve the phase information of the acceleration signals in the algorithm, which is expected to be detailed in our future work.

VIII. Defense

We now discuss possible directions to defend the proposed attack. As the lowest fundamental frequency of typical human speech is 85 Hz, one promising defense is to limit the sampling rate of the accelerometer. According to the *Nyquist sampling theorem*, an accelerometer working bellow 170Hz will not be able to reproduce any frequency components above 85Hz. Although the accelerometer can still be affected by the high-frequency audio signal, the captured information will be distorted and the recognition accuracy is likely to decrease.

In order to find an optimal threshold, we conduct speech recognition with accelerometer measurements sampled at 300 Hz, 200 Hz, 160 Hz, 100 Hz, and 50 Hz. Table XII shows the recognition accuracy on the digits dataset (10k single-digit signals from 20 speakers) under the table setting. It can be observed that, the recognition accuracy drops with the decreasing sampling rate and reduces to 30% at 50 Hz. In actual attacks, the recognition accuracy at 50 Hz could further decrease since the acceleration signal below 25 Hz can be significantly affected by human movement.

TABLE XII. The impact of the sampling rate (Table setting).

Sampling rate	300 Hz	200 Hz	160 Hz	100 Hz	50 Hz
Recognition accuracy	73%	64%	56%	47%	30%

According to Android Developer [2], the recommended sampling rates for the user interface and mobile games are 16.7 Hz and 50 Hz respectively. For activity recognition, 50 Hz is also more than sufficient since the frequencies of most human activities are below 20 Hz. Therefore, we suggest that applications requiring sampling rates above 50 Hz should request a permission through $< uses-permission >$, which will affect how Google play filters them [3].

Another effective defense is to notify the user when some applications are collecting accelerometer readings in the background with a high sampling rate. For instance, iOS presents a flashing "microphone" icon on the status bar when some applications are collecting voice signals in the background. A similar mechanism can be deployed on the Android system to remind users when, where and how their accelerometer readings are used.

IX. Conclusion

In this paper, we revisit the threat of zero-permission motion sensors to speech privacy and propose a highly practical side channel attack against smartphone speakers. We first present two fundamental observations that extend motion sensor based audio eavesdropping to everyday scenarios. First, speech signals emitted by smartphone speakers will always create significant impacts on the accelerometer of the same smartphone. Second, the accelerometer on recent Android smartphones can almost cover the entire fundamental frequency band of adult speech. On top of these pivotal observations, we propose **AccelEve**, a learning based smartphone eavesdropping attack that could recognize and reconstruct speech signals emitted by smartphone speakers, no matter where and how the smartphone is placed. With deep networks, adaptive optimizers, and robust & generalizable losses, our attack significantly and consistently outperforms baseline and existing solutions in all recognition and reconstruction tasks. In particular, **AccelEve** achieves three times the accuracy of previous work in digit recognition. For speech reconstruction, **AccelEve** is able to reconstruct speech signals with enhanced sampling rate, which covers not only the fundamental frequency components (vowels) in the low frequency band but also its harmonics in the high-frequency band. The consonants are not recovered because their frequencies (above 2000Hz) are far beyond the sampling rates of current smartphones.

Acknowledgment

We thank the reviewers for their helpful comments. This research is supported in part by National Natural Science Foundation of China (Grants No. 61772236), Zhejiang Key R&D Plan (Grant No. 2019C03133), Alibaba-Zhejiang University Joint Institute of Frontier Technologies, Research Institute of Cyberspace Governance at Zhejiang University, and McGill University's William Dawson Scholar Chair Professor Fund.

References

[1] "Coriolis force," https://en.wikipedia.org/wiki/Coriolis_force.

[2] "Sensor Overview," https://developer.android.com/guide/topics/sensors/sensors_overview.

[3] "uses-permission," https://developer.android.com/guide/topics/manifest/uses-permission-element.

[4] ANALOG DEVICES, "ADXL150/ADXL250," https://hibp.ecse.rpi.edu/~connor/education/EIspecs/ADXL150_250_0.pdf.

[5] S. A. Anand and N. Saxena, "Speechless: Analyzing the threat to speech privacy from smartphone motion sensors," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 1000–1017.

[6] S. A. Anand, C. Wang, J. Liu, N. Saxena, and Y. Chen, "Spearphone: A speech privacy exploit via accelerometer-sensed reverberations from smartphone loudspeakers," *arXiv preprint arXiv:1907.05972*, 2019.

[7] R. J. Baken and R. F. Orlikoff, *Clinical measurement of speech and voice*. Cengage Learning, 2000.

[8] S. Becker, M. Ackermann, S. Lapuschkin, K.-R. Müller, and W. Samek, "Interpreting and explaining deep neural networks for classification of audio signals," *arXiv preprint arXiv:1807.03418*, 2018.

[9] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile device identification via sensor fingerprinting," *arXiv preprint arXiv:1408.1416*, 2014.

[10] L. Cai and H. Chen, "Touchlogger: Inferring keystrokes on touch screen from smartphone motion." *HotSec*, vol. 11, no. 2011, p. 9, 2011.

[11] R. Chen and I. C. Paschalidis, "A robust learning approach for regression models based on distributionally robust optimization," *The Journal of Machine Learning Research*, vol. 19, no. 1, pp. 517–564, 2018.

[12] A. Das, N. Borisov, and M. Caesar, "Exploring ways to mitigate sensor-based smartphone fingerprinting," *arXiv preprint arXiv:1503.01874*, 2015.

[13] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

[14] R. N. Dean, G. T. Flowers, A. S. Hodel, G. Roth, S. Castro, R. Zhou, A. Moreira, A. Ahmed, R. Rifki, B. E. Grantham *et al.*, "On the degradation of mems gyroscope performance in the presence of high power acoustic noise," in *2007 IEEE International Symposium on Industrial Electronics*. IEEE, 2007, pp. 1435–1440.

[15] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi, "Accelprint: Imperfections of accelerometers make smartphones trackable," in *NDSS*, 2014.

[16] J. Doscher and M. Evangelist, "Accelerometer design and applications," *Analog devices*, vol. 3, p. 16, 1998.

[17] H. Feng, K. Fawaz, and K. G. Shin, "Continuous authentication for voice assistants," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. ACM, 2017, pp. 343–355.

[18] H. Fujisaki, "Dynamic characteristics of voice fundamental frequency in speech and singing," in *The production of speech*. Springer, 1983, pp. 39–55.

[19] S. Grawunder and I. Bose, "Average speaking pitch vs. average speaker fundamental frequency–reliability, homogeneity, and self report of listener groups," in *Proceedings of the International Conference Speech Prosody*, 2008, pp. 763–766.

[20] D. Griffin and J. Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[21] J. Han, E. Owusu, L. T. Nguyen, A. Perrig, and J. Zhang, "Accomplice: Location inference using accelerometers on smartphones," in *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*. IEEE, 2012, pp. 1–9.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2016, pp. 770–778.

[23] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "CNN architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2017, pp. 4700–4708.

[25] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711.

[26] T. Kaya, B. Shiari, K. Petsch, and D. Yates, "Design of a mems capacitive comb-drive accelerometer," in *COMSOL Conference, Boston*, 2011.

[27] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[28] J. Liu, C. Wang, Y. Chen, and N. Saxena, "Vibwrite: Towards finger-input authentication on ubiquitous surfaces via physical vibration," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 73–87.

[29] P. Marquardt, A. Verma, H. Carter, and P. Traynor, "(sp) iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers," in *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 2011, pp. 551–562.

[30] A. Matic, V. Osmani, and O. Mayora, "Speech activity detection using accelerometer," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2012, pp. 2112–2115.

[31] R. Matovu, I. Griswold-Steiner, and A. Serwadda, "Kinetic song comprehension: Deciphering personal listening habits via phone vibrations," *arXiv preprint arXiv:1909.09123*, 2019.

[32] Y. Michalevsky, D. Boneh, and G. Nakibly, "Gyrophone: recognizing speech from gyroscope signals," in *Proceedings of the 23rd USENIX conference on Security Symposium*. USENIX Association, 2014, pp. 1053–1067.

[33] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury, "Tapprints: your finger taps have fingerprints," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACm, 2012, pp. 323–336.

[34] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang, "Accessory: password inference using accelerometers on smartphones," in *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*. ACM, 2012, p. 9.

[35] M. Shoaib, H. Scholten, and P. J. Havinga, "Towards physical activity recognition using smartphone sensors," in *2013 IEEE 10th international conference on ubiquitous intelligence and computing and 2013 IEEE 10th international conference on autonomic and trusted computing*. IEEE, 2013, pp. 80–87.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[37] Y. Son, H. Shin, D. Kim, Y. Park, J. Noh, K. Choi, J. Choi, and Y. Kim, "Rocking drones with intentional sound noise on gyroscopic sensors," in *Proceedings of the 24th USENIX Conference on Security Symposium*. USENIX Association, 2015, pp. 881–896.

[38] I. R. Titze and D. W. Martin, "Principles of voice production," *Acoustical Society of America Journal*, vol. 104, p. 1148, 1998.

[39] T. Trippel, O. Weisse, W. Xu, P. Honeyman, and K. Fu, "WALNUT: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2017, pp. 3–18.

[40] Y. Tu, Z. Lin, I. Lee, and X. Hei, "Injected and delivered: fabricating implicit control over actuation systems by spoofing inertial sensors," in *Proceedings of the 27th USENIX Conference on Security Symposium*. USENIX Association, 2018, pp. 1545–1562.

[41] C. Xu, Z. Li, H. Zhang, A. S. Rathore, H. Li, C. Song, K. Wang, and W. Xu, "WaveEar: Exploring a mmwave-based noise-resistant speech sensing for voice-user interface," in *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2019, pp. 14–26.

[42] Z. Xu, K. Bai, and S. Zhu, "Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors," in *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2012, pp. 113–124.

[43] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[44] L. Zhang, P. H. Pathak, M. Wu, Y. Zhao, and P. Mohapatra, "Accelword: Energy efficient hotword detection through accelerometer," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2015, pp. 301–315.