Towards More Robust Keyword Spotting for Voice Assistants

Shimaa Ahmed
University of Wisconsin-Madison

Ilia Shumailov
University of Cambridge

Nicolas Papernot
University of Toronto and Vector Institute

Kassem Fawaz
University of Wisconsin-Madison

Abstract

Voice assistants rely on keyword spotting (KWS) to process vocal commands issued by humans: commands are prepended with a keyword, such as "Alexa" or "Ok Google," which must be spotted to activate the voice assistant. Typically, keyword spotting is two-fold: an on-device model first identifies the keyword, then the resulting voice sample triggers a second on-cloud model which verifies and processes the activation. In this work, we explore the significant privacy and security concerns that this raises under two threat models. First, our experiments demonstrate that accidental activations result in up to a minute of speech recording being uploaded to the cloud. Second, we verify that adversaries can systematically trigger misactivations through adversarial examples, which exposes the integrity and availability of services connected to the voice assistant. We propose EKOS (Ensemble for KeywOrd Spotting) which leverages the semantics of the KWS task to defend against both accidental and adversarial activations. EKOS incorporates spatial redundancy from the acoustic environment at training and inference time to minimize distribution drifts responsible for accidental activations. It also exploits a physical property of speech—its redundancy at different harmonics—to deploy an ensemble of models trained on different harmonics and provably force the adversary to modify more of the frequency spectrum to obtain adversarial examples. Our evaluation shows that EKOS increases the cost of adversarial activations, while preserving the natural accuracy. We validate the performance of EKOS with over-the-air experiments on commodity devices and commercial voice assistants; we find that EKOS improves the precision of the KWS task in non-adversarial settings.

1 Introduction

Voice assistants (VAs) interpret voice commands from their users to assist in different tasks, access services, and control smart devices. A typical voice assistant continuously samples audio through its microphone to detect a user saying a keyword, such as "Alexa," "Siri," or "Google." This process, referred to as Keyword Spotting (KWS), serves as the primary access control to an active voice assistant. Once it detects the wake keyword, the voice assistant streams the subsequently recorded audio to be analyzed as a voice command.

The Keyword spotting (KWS) task is a two-stage process spanning the device and cloud: a local on-device model first detects the keyword and sends a speech segment to the cloud, which verifies the keyword and processes the accompanying command [53]. Verification is necessary since on-device models are typically less accurate; they are optimized to minimize their compute footprint and latency of predictions [13, 44, 55], whereas the cloud model can be a full-fledged natural language model with higher precision.

In this paper, we find that unauthorized accidental activations due to poor precision of the on-device KWS model can lead to significant privacy violations with up to a minute of private speech being uploaded to the cloud. In addition, adversaries, who wish to get unauthorized access to the private VA, may systematically trigger such unauthorized activations with adversarial examples. This adversarial activation puts the device integrity and the user's security at risk, given the numerous appliances and services connected to voice assistants (e.g., garage door, lights, and credit cards) [15, 35, 45].

As the entry point for any interaction with the VA, improving the precision of on-device KWS directly limits the extent of private conversations leaked to the cloud and reduces the attack surface available to adversaries. Existing defenses to these problems rely on generic machine learning approaches, such as adversarial training [38]. Such approaches typically harm the natural accuracy—an unacceptable proposition for VAs—or fail to provably increase the cost of an adversary launching an over-the-air attack. Other approaches employ liveness detection mechanisms [3] that potentially introduce additional privacy problems and do not address the accidental activations problem. In short, this paper considers the question of *how to improve the robustness of KWS against accidental and adversarial activations while preserving its precision?*

In this paper, we design, implement, and evaluate EKOS (Ensemble for KeywOrd Spotting) as an affirmative answer to the above question. EKOS leverages the semantics of the KWS task to arrive at a more favorable tradeoff between the robustness and precision of the KWS model. First, EKOS incorporates spatial diversity from the acoustic environment at both training and inference time to minimize distribution drifts responsible for accidental activations. Second, it exploits a physical property of speech—its spectrum redundancy—to deploy an ensemble of models trained on different harmonics.

It provably forces the adversary to modify more of the frequency spectrum to obtain successful adversarial examples.

Modeling distribution drifts responsible for accidental activations is challenging because the physical environment evolves constantly. EKOS addresses this issue by exploiting the natural randomness from the physical environment (such as room impulse responses) and ensembling other voice-aware devices available in the vicinity of the virtual assistant. In particular, EKOS performs KWS with an *ensemble* of models, each served by a device with varying internal sensors, hardware, and channel from the user. EKOS uses the *diversity* ensuing from the ubiquity of smart devices in a given environment, such as tablets, computers, and smartphones, to improve the precision of the KWS task by combining the detection results from these devices.

Improving robustness to adversaries is more challenging because they can still overcome ensembles of models [27, 59], especially when the feature space is *common to all* models. EKOS addresses this challenge by utilizing the redundancies in speech signals and properties of the KWS task. A speech signal carries replicas of the same content (i.e., a word) at different frequency components: harmonics. It is thus possible to slice the signal's spectrogram into different slices and assign each slice to a different model without much impact on the natural accuracy. We design these slices and architectures to exhibit poor transferability. Further, EKOS randomizes the slice-architecture combinations in the ensemble at run-time. This approach increases the cost of an adversary because they now have to perturb a majority of the frequency slices before they can control the predictions of the ensemble.

In summary, our contributions are as follows:

1. We show that privacy leakage is greater than previously believed when on-device models send private conversations to the cloud due to accidental activations. Previous analysis [23] reported misactivations resulting in 10 seconds of speech being leaked; our evaluation shows that some misactivations lead to up to a minute of speech leaking to the cloud (Sec. 6.2.3 — Fig. 7).

2. We design an ensemble of KWS detectors that can run on distributed devices in an environment. This ensemble leverages the semantics of the KWS task, the properties of the audio channel, and the nature of the speech signal to introduce real diversity to the prediction task (Sec. 5).

3. Our end-to-end evaluation shows that an ensemble of three to five devices, with random slicing and architectures, increases the cost of adversarial attacks (Sec. 6.1.3, 6.2.4). At the same time, EKOS preserves the natural accuracy, approximating the baseline accuracy and has little performance overhead (Fig. 3, 5). We validate the performance of EKOS with over-the-air experiments on commercial devices; we find that EKOS improves the precision of the KWS task in non-adversarial settings (Sec. 6.1.2, 6.2.3).

4. We generate and release[1] a dataset of the Amazon Echo's wake keywords: $\{Alexa, Computer, Amazon, Echo\}$. We use this dataset to validate EKOS robustness on Amazon's Echo devices. The same methodology can be followed for other commercial devices and keywords.

2 Background on Keyword Spotting

The KWS task is responsible for detecting a set of predefined *keywords* in an audio stream. Typically, the VA's microphone(s) capture the over-the-air audio stream. Then, the VA performs audio pre-processing and KWS classification.

Physical Environment. When an audio signal is transmitted over-the-air, the signal reflects off the room walls and the objects in the room. The received signal at a microphone is the sum of the line-of-sight and reflected audio copies, known as reverberations or echo, as shown in Fig. 1. The reverberation can be modeled via a room impulse response (RIR) $h(t)$, and the received signal is the convolution of the transmitted audio and the RIR, $r(t) = s(t) * h(t)$, where $h(t)$ depends on the speaker and microphone locations, the room dimensions, objects, and the materials absorption factors. Hence, $h(t)$ is unique per every room and speaker-microphone setup.

Feature Extraction. The mel-frequency cepstrum coefficients (MFCC) are the conventional features used for speech recognition tasks including ASR and KWS; they reduce the dimensionality of an audio signal, $r(t)$, to a 2D temporal-spectral map. The MFCCs are computed as follows [40]: (1) divide $r(t)$ into short time frames (20–40ms); (2) compute the short-time Fourier transform (STFT) of these frames; (3) map the STFT linear frequency scale to the mel-scale using a mel-spaced filterbank. The mel-scale approximates the human auditory system as it applies more (fewer) filters in the low (high)-frequency range; (4) take the log of the power; and (5) apply the discrete cosine transform (DCT). The MFCCs are the coefficients of the resultant spectrum at each time frame.

Classification. The KWS task employs a multi-class model $f(\cdot)$ to classify an input audio $r(t)$ as a label corresponding to the detected keyword, with the "*unknown*" label for non-keyword speech. The model consists of three components: (1) extracting MFCC features from $r(t)$, (2) feeding the MFCCs to a deep neural network (DNN), and (3) computing an average score of the individual frames' posterior scores to report the keyword score. Earlier research on KWS considered DNN architectures which treated MFCCs as 2D features [48, 58].

Choi et al. [16] were the first to treat the MFCCs as a 1D time signal, where the frequency coefficients are the input channels. They proposed TC-ResNet, a temporal convolution residual network architecture. The 1D temporal convolution reduces the feature map size and has a large receptive field

[1] github.com/wi-pi/EKOS

since the filter covers the whole range of frequencies (channels). It achieves better performance at a smaller number of parameters and computations, hence, lower latency. We utilize these architectures in the design of EKOS.

3 Background on KWS Misactivations

The KWS performance is crucial for the VA's user experience [46]. A near-optimal true-positive rate is essential for the device's responsiveness and utility. On the other hand, a KWS *misactivation* compromises the user's privacy and the VA's integrity. A misactivation takes place when the VA is activated by an unauthorized command, i.e., a sound that is not the correct keyword. In this work, we consider two types of misactivations: accidental and adversarial activations.

3.1 Accidental Activations

An *accidental* activation happens when the KWS model *mistakenly* interprets a sound that is not the keyword as a positive activation, i.e., a false-positive detection. In such a case, the VA inadvertently records the user's private conversations and sends them to the cloud for transcription and execution.

The privacy threats stemming from having an *always* listening microphone in private spaces have been extensively studied [1, 9, 12, 24, 37, 39, 66]. Recently, two studies [23, 53] performed a comprehensive analysis of the accidental activation triggers on a variety of VA devices and keywords. They use TV shows, newscasts, and speech datasets to locate phrases that accidentally trigger each VA. Dubois et al. [23] observed 0.95 misactivations per hour, where they identified some activations lasting for at least 10 seconds. Likewise, Schönherr et al. [53] located hundreds of accidental activations in the evaluated media. They observe that the cloud-based KWS verification model reduces the number of local misactivations. Yet, more than half of the evaluated triggers still incorrectly activate the cloud's model. Moreover, they created a dataset of more than 1000 English n-gram phrases that are phonetically similar to the commercial keywords; these phrases are likely to cause misactivations. Both studies noted that the VA's operation is non-deterministic; it is hard to predict when a device may be accidentally activated.

3.2 Adversarial Activations

As far as their integrity[2] is concerned, KWS models are vulnerable to inference time adversarial examples [26, 57], where an adversary constructs *imperceptible* commands hidden in a non-suspicious audio utterance, such as music or a YouTube video, to wake up and interact with the VA [15, 45, 52].

[2]We note that integrity is not the only property adversaries may target. Attackers also jeopardize the availability of the ML system, as shown in recent work on the presence of adversarial music [35] or Sponge Examples [54].

Given an audio signal $r(t)$, and a KWS model $F(\cdot)$, the attacker's objective is to find a small perturbation δ, such that $F(r(t)+\delta) = y$, where y is the target keyword that triggers the VA. We refer to this attack as an *adversarial* activation.

Adversarial Examples on Audio. Carlini and Wagner [11] constructed a targeted white-box attack on the neural ASR system, Deep Speech. The attack is *digital*; i.e., it does not consider a physical channel and assumes the audio stream is directly fed to the model. The attack optimizes this objective:

$$\min \ \ell(F(s+\delta),\, y) \ + \ \alpha \cdot \|\delta\|_\infty \quad s.t. \quad \|\delta\|_\infty < \varepsilon, \qquad (1)$$

where s is the input to the neural network $f(\cdot)$, δ is the perturbation, y is the target label, ℓ is the loss function, ε is the attack budget which bounds the maximum added perturbation, and α is a hyperparameter; the adversarial example is $s'(t) = s(t) + \delta$. The authors choose ℓ to be the CTC (Connectionist temporal classification) loss and use the max-norm ($\|\cdot\|_\infty$) which has the effect of adding a small perturbation consistently throughout the utterance samples. This attack, however, is against ASR, not KWS; both ASR and KWS have similar preprocessing pipelines involving MFCCs, but the task solved by each model is different.

The adversarial example $s'(t)$ constructed with Eq. 1 is neither completely imperceptible nor effective over-the-air. The former requires that $s'(t)$ sounds very similar to $s(t)$ to a human listener. The latter requires that $F(s'(t) * h) = y$ for any h, where h is the physical environment room impulse response (RIR) (Sec. 2). Following this initial attack, recent works have focused on solving these two challenges.

Imperceptibility. Schönherr et al. [52] examine a different bound on the perturbation that better addresses the human auditory system perception. They propose psychoacoustic masking, as in MP3 encoding, to hide the perturbations around the original speech frequency components, where they are barely perceptible to humans. However, their attack assumed a perfect channel; i.e., it is not robust over-the-air.

Over-the-air Robustness. Adversarial examples are not robust in the physical world when the input signal is subject to environmental variations (transformations)—as initially observed in vision [5]. The adversary can adapt by considering the distribution of possible transformations, and optimizing the perturbation over the Expectation over Transformation (EoT) [5], such that the resulting perturbation transfers across these transformations *on average*. Qin et al. [45] and Schönherr et al. [52] apply EoT to the acoustic domain to capture room reverberation. They convolve the audio signal with RIR:

$$\min \ \mathop{\mathbb{E}}_{h\sim\mathcal{H}}[\ell(F((s+\delta)*h),\, y)] \ + \ \alpha \cdot \|\delta\|_p \quad s.t. \quad \|\delta\|_p < \varepsilon, \qquad (2)$$

where $\mathcal{H}$ is the RIR distribution of the possible room dimensions, and speaker and microphone locations.

4 System and Threat Models

System Model. We assume the VA to exist in an environment that contains a set of trusted devices, such as smartphones, computers, and tablets. Each device has at least one microphone, a network interface, and computing capabilities. We believe these assumptions are realistic about the households or spaces with a VA. As in any realistic setting, these devices are randomly located within the environment, experiencing random acoustic channels, and have inherent hardware variations, as shown in the setup at Fig. 1 (left). The user deploys EKOS by installing an app on their microphone-equipped devices. The app runs in the background, reads the microphone, performs KWS, and communicates with the VA.

Threat Model. We consider two independent threat vectors that result from false VA activations due to the KWS model's imperfections. Both vectors are different in the adversary definition, attack implementation, and the subsequent privacy and security violations. We do not suggest that the same adversary can execute both threat vectors; yet, both threats are enabled by the same vulnerability: a false VA activation.

The first threat vector covers a *remote* and *passive* adversary with access to the VA's recordings once they are uploaded to the cloud. Because of imperfections of KWS models, the VA can be accidentally triggered, causing it to record conversations not intended as commands. Although the cloud has access to the users' legitimate commands, accidental activation poses real privacy threat [46, 53]. Under a legitimate activation, the user is aware that their commands will be recorded and uploaded to the cloud. Detecting the legitimate keyword forms an implicit consent to be recorded. However, in the case of accidental activation, the recorded conversations are private; the users are unaware and did not approve the recording. The privacy concerns stem from the content of the private conversation, the context, and the background noise. Under this setting, the user's privacy can be compromised in different ways: (1) the cloud uses these recordings to train ML models [43, 49], these models can memorize the training data [10]; (2) an adversary compromises the cloud servers and leaks such conversations [8, 19, 65]; or (3) third-party transcription contractors or law enforcement agencies can potentially have access to the private recordings [21, 36, 61].

The second threat vector covers a *remote* and *active* adversary who activates the VA with imperceptible perturbations hidden in a non-suspicious audio utterance, e.g., music. This adversary can remotely trick the user into playing audio from a TV, YouTube, or SoundCloud, which embeds the imperceptible perturbation – scaling the attack to many users. Prior research has demonstrated the feasibility of generating adversarial samples in the form of inconspicuous background music [15, 35]. Once the VA is activated, the adversary can push commands to activate malicious skills or interact with physical devices in the user's environment. Such adversarial activation puts the device's integrity and user's security at risk given the numerous services and appliances connected to the VA (e.g., garage door, bank accounts). We consider a white-box attacker who has access to the KWS model parameters as well as EKOS's setup internals. This adversary can launch adaptive attacks in an attempt to circumvent EKOS. Note that the adversary has no *physical* access to the VA; otherwise, the adversary can interact with the device using their own voice without the need to launch adversarial perturbations.

Threat vectors that directly attack the microphone interface, such as ultrasound [47, 67] and laser attacks [56], are outside the scope of this work as they are not based on false activations of the VA. Our work is orthogonal and can compose well with approaches to defeat these other threats [7, 62]. In Sec. 7, we discuss how EKOS can address these threats.

5 EKOS: Ensemble for KeywOrd Spotting

5.1 High-level Overview

EKOS comprises two components: a machine learning-based component (ensemble learning) to improve the robustness of the KWS task against accidental activations, and a signal processing-based component (feature slicing) to handle adversarial examples against KWS.

Fig. 1 illustrates the high-level operation (left) and the processing pipeline of EKOS (right) at each device. EKOS deploys *diverse* keyword spotting models on a set of commodity devices, such as smartphones, smart TVs, laptops, and edge devices, and combines their decisions to improve the overall classification performance, a technique known as ensemble learning. EKOS views the output of each model as an independent random variable (vote) specifying the identified keyword from the input audio. All devices run a lightweight webserver and are connected to the same wireless network. The ensemble integration happens as follows: (1) the main VA (server) listens continuously; it initiates the KWS vote collection from the other devices (clients) upon detecting a keyword; (2) the main VA issues parallel requests to the client devices and waits for their response; (3) each client device buffers its microphone signal and waits for an inference request; (4) upon receiving one, it runs its KWS model and returns the predicted hard label; and (5) the main VA outputs the final prediction through a majority vote mechanism.

Accidental Activations. A key point to harvest the gain of ensemble learning is to ensure (as much as possible) that the models' errors are uncorrelated [22, 60]; a voting mechanism in such a case would reduce the false positive rate responsible for accidental activations. EKOS satisfies this condition by introducing different levels of diversity at the model design and the received input signal.

EKOS processes speech samples using a set of l KWS models. We introduce diversity into the models decisions by selecting different architectures and hyperparameters for each model (Sec. 5.3). EKOS allows this KWS ensemble to

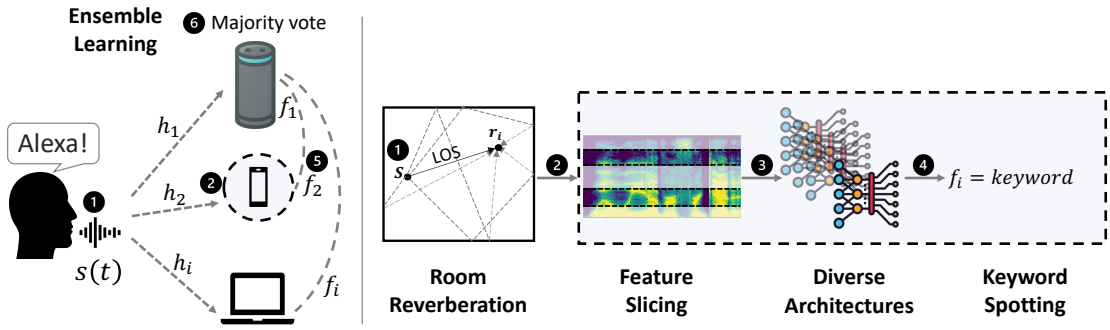


Figure 1: EKOS Overview. Left: High-level operation. Right: Details of the signal processing pipeline at the device. Step (1) is the spoken speech signal, step (2) is the received signal after experiencing the acoustic channel, step (3) is applying a random feature slicing filter, step (4) is passing the filtered signal through a randomly chosen architecture, step (5) refers to sending the decisions from individual devices to the VA, and step (6) is the final ensemble output of a majority vote.

be either centralized in the VA or distributed over a set of N smart devices existing in the environment. These devices experience different acoustic propagation channels and have inherent hardware diversity. Hence, they capture uncorrelated samples of the audio stream [29].

Adversarial Examples. In its second component, EKOS leverages diversity from the feature space and the environment to increase the cost of generating adversarial examples against KWS. EKOS decomposes the speech spectrum into a set of possibly overlapping spectrum slices (feature slicing in Sec. 5.2). Because of the nature of the speech signal, the spectrum slices contain harmonics that encode replicas of the speech content. These frequency components, however, undergo different transformations as they travel across the physical channel. As a result, each spectrum slice is useful in identifying keywords found in speech but requires the adversary to inject a perturbation specific to this spectrum slice.

Ensemble devices behave independently at run-time; each device chooses a subset of spectrum slices at random. Then, it passes the slice into a randomly chosen architecture (randomized ensemble in Sec. 5.3). Each model assigns the slice with a classified keyword and relays its label to the VA.

The robustness in this approach arises from three insights related to the classification of audio signals. First, the channels are spatially independent; the adversary has to account for more transformations in generating the adversarial examples. As specified in Sec. 3.1, an adversary uses the expectation over transformation technique to generate adversarial examples that are adaptive to this defense, where each transformation represents a simulated channel. Having a set of simultaneous independent channels constrains the attacker's optimization problem further; the result is a less optimal (larger) perturbation (Sec. 6.1.3). Second, adversarial examples in the audio domain have poor transferability properties; an adversarial example optimized for a slice-architecture combination does not transfer easily to other combinations. This insight is supported by previous results about the transferability of audio adversarial examples [2], as well as our results presented later in Sec. 6 and Fig. 10. Third, EKOS chooses the slices and architectures randomly at run-time, which forces the adversary to cover more slice-architecture combinations to ensure a sufficiently large probability of attack success. In the following, we discuss EKOS's feature slicing and randomized ensemble.

5.2 Feature Slicing

The feature slicing in EKOS applies bandpass filters to the speech spectrogram to select frequency slices. EKOS leverages a key property of audio signals: they carry replicated information across the different frequency bands. This information content, however, is not uniform; bands in lower frequencies contain more information than bands in the higher frequency range. This insight forms the basis for the MFCCs, which perform non-linear mel-scaling of bands as inspired by the human auditory system [40]. In EKOS, we follow a similar methodology; we define six bandpass filters, three at the lower end of the spectrum spanning the bands: (1)-[0Hz, 750Hz], (2)-[700Hz, 1700Hz], (3)-[1650Hz, 2900Hz], and another three at the higher end of the spectrum: (4)-[2850Hz, 4350Hz], (5)-[4300Hz, 6050Hz], (6)-[6000Hz, 8000Hz]. Notice that the bandwidth of the filters increases linearly from 750Hz to 2000Hz with 250Hz increments. These bandpass filters are the building blocks of the feature slicing filters.

The design of these filters involves two tradeoffs between natural and adversarial robustness. The first tradeoff is the width of the filter. A set of narrow filters force the attacker to add the perturbation in more concentrated frequency regions, making it harder to hide imperceptible perturbations [45] – at the cost of reduced natural accuracy. The bandwidth has to be wide to capture more content of the speech signal.

The second tradeoff concerns the overlap between the filters. If filters overlap to the extent that they all share a common frequency band, the attacker's strategy would be to target

this single shared band, resulting in a single perturbation that transfers across all the filters; the attacker's optimization function resolves to a single objective as in Eqn. 3. On the other hand, if the filters have no overlap, the number of possible filters will be limited. Moreover, the attacker can target such mutually exclusive bands separately, where the final perturbation is the sum of the individual perturbations. Hence, it leads to less robustness and randomness in the EKOS ensemble.

As such, we design the set of filters G such that each feature slicing filter $g \in G$ includes two bandpass filters, one chosen from the set of lower bands (filters 1, 2, and 3) and the other chosen from the set of higher bands (filters 4, 5, and 6). This design results in G comprising nine combinations $\{(1,4),(1,5),(1,6),(2,4),(2,5),(2,6),(3,4),(3,5),(3,6)\}$. Any single band is repeated only three times across the filters set G. Hence, $g = w[s_l, e_l] + w[s_h, e_h]$, where w is a rectangular window function, s_l, e_l are the low-frequency window start and end frequencies, and same for s_h, e_h for the high-frequency window. This design balances the amount of information passed by each slicing filter and intentionally adds overlap between the filters without sharing any single band among all of them. We show later (Sec. 6.1.3) that the designed filters preserve the model's natural accuracy and provide feature space diversity such that their ensemble accuracy approximates the baseline accuracy.

5.3 Runtime Ensemble

In an environment with N devices, the user's speech signal $s(t)$ travels over a set of channels $h_i(t)$; each device d_i receives a signal $r_i(t) = s(t) * h_i(t)$. A device d_i has access to the set of G filters as defined in Sec. 5.2 and a set $\mathcal{F} = \{F_k |\ 1 \leq k \leq K\}$ of architectures, where K is the number of baseline KWS architectures. We refer to $F_{j,k}$ as the architecture F_k trained after applying filter g_j. Each device can run one or a subset of KWS models simultaneously based on its processing capabilities, the availability of other devices, and the user's preferred level of privacy and utility (Sec. 7). The device chooses randomly a subset $G_i \subseteq G$ frequency filters. It applies each $g_{i,j} \in G_i$ to r_i resulting in a set of signals $r_{i,j}(t) = r_i(t) * g_{i,j}(t)$. Then, the device assigns each $r_{i,j}(t)$ a random architecture $F_k \in \mathcal{F}$; each model outputs $f_{i,j} = F_{j,k}(r_{i,j}(t))$, where $f_{i,j}$ indicates the output class (keyword). Each device d_i sends the set $\{f_{i,j} | 1 \leq j \leq |G_i|\}$ to the VA for the final decision. The VA receives a set of l decisions from all the devices, such that $l = \sum_i^N |G_i|$. It performs majority voting by choosing the class with the highest number of votes.

We exhaustively searched through the models trained over slice-architecture combinations to ensure adversarial examples have low transferability. Fig. 10, in Appendix, shows that these models exhibit poor transferability. We conjecture that reducing the overlap between the filters in G contributes to this observation. This poor transferability is an important property for EKOS's robustness, as discussed in Sec. 5.4.

An adaptive attack can target the ensemble models simultaneously [27, 59] given a higher perturbation budget. Thus, we introduce inference time randomization to EKOS's operation: we randomize the slice-architecture combination. At each T_i interval, each device i randomly selects a frequency filter subset $G_i \subseteq G$ and assigns each filter $g_{i,j} \in G_i$ a random architecture $F_k \in \mathcal{F}$, where T_i is independently set by each device. Hence, the slice-architecture combinations independently and randomly change every T_i.

Finally, EKOS design is flexible and can be optimized towards a customized utility-robustness level. The user has the option not to apply the feature slicing prior to the ensemble. In such a case, EKOS does not apply the randomized feature and architecture selection. It just passes the received signal at each device to a model F_i and aggregates the decisions at the VA. This mode improves the KWS accuracy against accidental activations but not against adversarial activations. Moreover, the user sets EKOS' hyperparameters, such as N, l, K, and $|G_i|$, to optimize the computational overhead (Sec. 7).

5.4 Robustness Properties

The robustness of EKOS arises from the increase in the attacker's cost. The original attack requires optimizing over a single constraint to force a label y, such that:

$$min||\delta||_p, \ \text{s.t.} \ \mathop{\mathbb{E}}_{h\sim\mathcal{H}}[F((s(t)+\delta) * h(t) * g(t))] = y, \qquad (3)$$

where $h \sim \mathcal{H}$ is a random variable describing the channel between the speaker and possible devices.

Introducing the ensemble of slice-architecture combinations, and assuming the attacker knows the chosen slices and architectures, the attacker's optimization objective comprises multiple constraints. Without loss of generality, assume that each device d_i runs a single model $F_{j,k}$ for a specific filter $g_{i,j}$. The attacker's objective can be represented as:

$$\begin{aligned} &\min \|\delta\|_p \ \text{s.t.} \\ &\Bigg(\mathop{\mathbb{E}}_{h_0\sim\mathcal{H}} \left[F_{j,k}\left((s(t)+\delta) * h_0(t) * g_{0,j}(t)\right)\right] = y \\ &\qquad \cdots \\ &\wedge \mathop{\mathbb{E}}_{h_{l/2}\sim\mathcal{H}} \left[F_{j,k}\left((s(t)+\delta) * h_{l/2}(t) * g_{l/2,j}(t)\right)\right] = y \Bigg). \end{aligned} \qquad (4)$$

Because of majority voting, the attacker has to satisfy a set of $l/2+1$ constraints to control the ensemble output. Intuitively, this optimization problem is more constrained and will result in a larger perturbation compared to the less constrained problem of one slice-architecture combination. This property, however, only holds when gradients of the constraints are linearly independent. Otherwise, the same perturbation may be able to force models trained on two or more spectrum slices to misclassify when these models' gradients are linearly dependent. In EKOS, we encourage gradients to be linearly

independent with diverse architectures and by designing the filters to have little overlap (Sec. 5.3).

EKOS randomizes the slice-architecture selections at runtime to increase the cost of the attack. Given a set of M possible channel-slice-architecture combinations, the adversary has to attack the M combinations simultaneously to overcome the randomized ensemble and guarantee attack success, provided that poor transferability properties hold. This introduces a tradeoff between the attack success and the perturbation size. The attack success increases when the attacker covers more channel-slice-architecture combinations at the cost of constraining the optimization problem further. We evaluate the effect of inference time randomness on the attack in Fig. 5.

6 Evaluation

We evaluate EKOS in two scenarios: through (1) end-to-end open-source (white-box) models (Sec. 6.1) in a simulated environment and a physical over-the-air environment, and using (2) black-box commercial VAs (Sec. 6.2). We design the evaluation in each scenario to answer these questions:

1. *Q1*: Does EKOS reduce the accidental activation instances? – Sec. 6.1.2, 6.2.3.
2. *Q2*: Does EKOS increase the cost of generating an *adaptive* adversarial activation attack? – Sec. 6.1.3, 6.2.4.
3. *Q3*: What is the performance overhead of EKOS in terms of natural accuracy and latency? – Sec. 6.1.3, 6.1.4.

6.1 Open-Source Models

We implement EKOS on open-source models and datasets.

6.1.1 Experimental Setup

Keyword Spotting. We use Google's Speech Commands dataset [63] for training and testing KWS models. The dataset consists of approximately 65,000 one-second long utterances of 30 short words, from thousands of different speakers. Similar to prior work, we select 12 labels: {*yes, no, up, down, left, right, on, off, stop, go, silence, unknown*} [16, 58]. We split the data into: 80% training, 10% validation, and 10% testing (3081 samples). We use Choi et al.'s implementation of the dense (DS-CNN), 1D temporal ResNet (TC-ResNet), and 2D ResNet (TC-ResNet2D) models [16], which achieve the highest accuracy with a reduced inference time.

Simulated Environment. We simulate the over-the-air channel using *Pyroomacoustics* [51] python package[3]. This package implements the image-source model [4] to calculate the acoustic reverberation and generate the room impulse response (RIR). We generate 1000 unique RIR samples where the room dimensions, speaker, and microphones locations are drawn uniformly at random. During audio pre-processing, we apply background noise, RIR convolution, and random shift to the speech samples to approximate real-world scenarios.

Over-the-air Environment. In the physical setup, we evaluate EKOS on a set of commodity devices with varying background noise. We deploy EKOS on six devices (D1–D6): a MacBook Pro laptop, an iPad tablet, a Dell PC with a high-quality directional microphone (Blue Snowball)[4], a Dell laptop, a Google Pixel XL phone, and a Google Pixel 2 XL phone. The devices are distributed in a lab space (14.2x7x3.8m). All devices run a lightweight webserver and are connected to the same wireless network. The PC is the main VA (server): it requests and aggregates votes from other devices (clients).

We use two Echo Dot devices as Bluetooth speakers; the first plays the keywords and the second plays background noise at half the volume. We evaluate four background scenarios: (1) noise naturally found in the lab, including a humming AC, keyboard typing, and mouse click sounds; (2) popular English songs (music & speech); (3) Google Commands dataset noise files that include doing the dishes, biking, running water, miaowing, white and pink noise; and (4) classical music[5].

Attack against a single model. We build on Qin et al.'s implementation[6] [45] for imperceptible and over-the-air robust adversarial examples on ASR (Sec. 3.2). Note that this attack is robust only on *simulated* environments. In contrast with ASR, which involves sequence-to-sequence modeling, KWS is a single word classification task. Thus, we simply apply the cross-entropy loss (instead of the CTC loss) with a regularizer for either robustness or imperceptibility.

Attack against an ensemble. Alongside attacks on individual models, we evaluate an adaptive attacker. We consider the strongest possible threat model, where an adversary has full access to the ensemble details. This adversary targets EKOS ensemble as a whole: it calculates the overall loss by summing the predicted logits (i.e., the scores assigned to each class) across the ensemble models on the input to be attacked. Then, the attack is optimized directly on this combined loss.

6.1.2 Accidental Activation Evaluation

First, we evaluate EKOS's performance against accidental activations and compare it to the baseline (single KWS model) performance. Therefore, we exclude the feature slicing component from EKOS's pipeline in this experiment.

Simulated Evaluation. We evaluate an ensemble of l models, where each model experiences a unique channel (RIR). We evaluate two scenarios: (1) the same architecture is deployed on all l models, and (2) each model independently

[3] github.com/LCAV/pyroomacoustics

[4] bluemic.com/en-us/products/snowball/

[5] youtube.com/watch?v=y1dbbrfekAM

[6] github.com/tensorflow/cleverhans/tree/master/examples/adversarial_asr

Architecture	BL	$l=1$	$l=3$	$l=4$	$l=5$
DS-CNN-M	94.61	82.53 ± 1.63	84.31 ± 1	83.95 ± 0.85	84.12 ± 0.81
TC-ResNet14	96.43	91.74 ± 1.9	93.71 ± 1.34	93.43 ± 1.12	94.13 ± 0.97
TC-ResNet2D8	96.85	84.64 ± 1.74	86.27 ± 0.54	86.51 ± 0.87	86.97 ± 0.5
TC-ResNet8	96.50	92.99 ± 1.19	93.85 ± 0.74	94.57 ± 0.71	94.52 ± 0.43
Random Arch.	–	–	95.131 ± 0.81	94.606 ± 0.67	95.141 ± 0.93

Table 1: Accidental activation accuracy (%) (mean±std) of an ensemble (of size l) in a simulated environment without enabling feature slicing.

Attack	D1	D2	D3	D4	D5	$l=5$
PGD	33.78	28.67	39.0	34.33	33.33	46.89
PGD_RIR	37.22	36.44	48.89	41.11	34.0	54.33

Table 2: Over-the-air adversarial accuracy (%) of individual device and EKOS at $l=5$ against PGD and PGD with RIR attacks, 90 examples each.

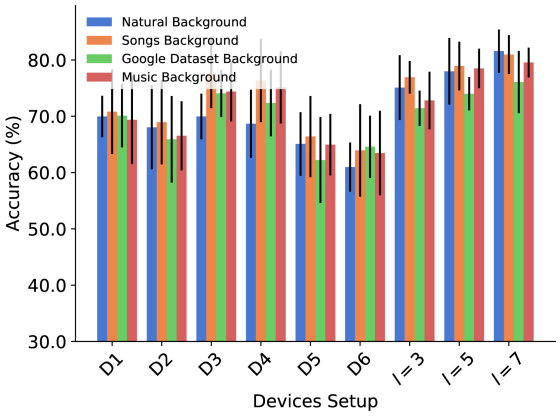


Figure 2: Over-the-air accuracy (mean and std) of EKOS against accidental activation for different background noises, ensemble size (l), and architecture selection. EKOS outperforms individual devices (D1–D6) under all scenarios.

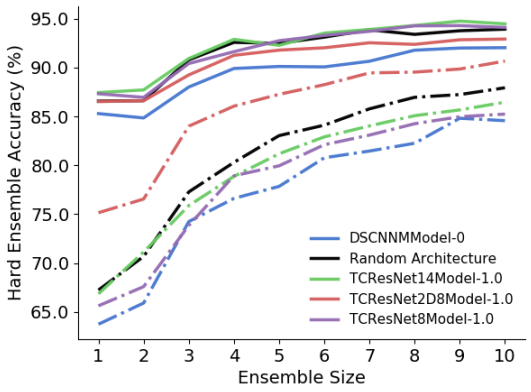


Figure 3: Natural mean accuracy (%) of EKOS with feature slicing (solid lines) and filter cutoff shift (dotted lines) at different architectures and ensemble sizes. The feature slices and cutoff shift are randomly selected at run-time.

selects an architecture at random with replacement. We run the evaluation 20 times to account for randomness.

Table 1 shows the mean and standard deviation accuracy at $l=1$, i.e., a single device, and at an ensemble of size $l=3,4,5$, versus the unrealistic baseline (BL) accuracy when the audio is directly fed to the model (digitally rather than physically). An ensemble of size 3 outperforms the single device for all architectures. There are diminishing returns for ensembles with more than three models. Random architecture selection also outperforms individual architectures. We thus confirm that an ensemble of diverse architectures and audio channels enhances the natural accuracy of any single model.

Note that Google Commands is a multi-class and balanced dataset with 12 classes. Classifying each keyword with high accuracy means fewer errors, hence, lower *accidental* (erroneous) activations. Thus, the classification accuracy on such a dataset is an indication of robustness to accidental activations.

Over-the-air Evaluation. We play the same 3081 test samples over the air and record the six devices' microphones. We feed these samples to the four KWS architectures. Fig. 2 shows individual devices (D1–D6) mean accuracy and standard deviation across architectures and background noise. Devices closer to the speaker (D1, D4) achieve higher accuracy than other devices (D2, D5, D6) for all noise types since they experience a higher signal-to-noise ratio (SNR). D3 also performs well since it utilizes a directional microphone.

Next, we randomly combine the six devices in an ensemble of size $l=3,5,7$. Each model in l selects its architecture independently at random with replacement. We repeat the evaluation ten times to account for architecture and device selection randomness. Fig. 2 shows that EKOS outperforms all the individual devices under all background scenarios. Hence, the physical evaluation matches the simulated evaluation and validates EKOS' robustness against accidental activations.

6.1.3 Adversarial Activation Evaluation

Second, we evaluate EKOS's performance against adversarial activations: we now include the feature slicing component.

Simulated Evaluation. Before we evaluate the robustness benefits of feature slicing, we ensure that our pipeline maintains its natural accuracy. Fig. 3 shows the performance of EKOS at different architectures and ensemble size, with two levels of inference time randomness; (1) random filters selection from the set G, and (2) random filter cutoffs shift at run time. We apply random shifts drawn uniformly from the range $\pm 200Hz$ to the 4 cutoff parameters (s_l, e_l, s_h, e_h).

First, when applying random feature slicing, an ensemble of only $l = 5$ improves the individual models' accuracy by an average of 6%—corresponding to 50% error rate reduction, at all architectures. Hence, the $l = 5$ ensemble accuracy approximates the models' accuracy in Table 1, where no feature slicing is applied. Second, when the $\pm 200Hz$ random cutoff shift is applied, it deteriorates the models' accuracy. Still, the ensemble accuracy increases with the ensemble size.

Next, we evaluate EKOS against an adaptive white-box attacker. We compare the performance of Projected Gradient Descent (PGD), PGD with frequency masking, and PGD with 20 RIRs attacks. The adaptive attack is performed over an ensemble of sizes 1, 3, and 5, repeated five times for each ensemble size. All attacks use 100 iterations to accurately approximate the shortest distance to the decision boundary.

Fig. 4 plots the false activation rate on adversarial examples as a function of the attack budget, with standard deviation computed over different keywords. The baseline shows TCResNet8 model trained on all the spectrum; we select TCResNet8 as it shows the highest robustness among the baseline architectures. The figures show that as the ensemble size increases, the adversary is unable to maintain the same attack performance compared to the baseline; i.e., the adversary needs a higher perturbation budget to reach a specific false activation rate. At higher attack budget, the perturbation power increases which leads to higher attack perceptibility. Therefore, EKOS increases the cost the adversary faces.

Most interestingly, we find that EKOS makes it hard to launch frequency masking attack efficiently (Fig. 4b). The mask constraint is no longer satisfied as frequency peaks are not necessarily used by individual models due to feature slicing. The PGD with frequency masking attack on EKOS is effectively relaxed to the less constrained PGD attack in Fig. 4a. For PGD with RIR attack, Fig. 4c shows a lower false activation rate, w.r.t. Fig. 4a and 4b, at the low attack budget due to RIR randomness. Note that PGD with RIR optimizes the perturbation over an EoT of the RIR transform (Eqn. 3); the perturbation is not guaranteed to succeed at run-time.

Fig. 8, in the Appendix, similarly shows the performance of the attacks in the presence of a random filter cutoff shift of $\pm$200Hz. The models exhibit behavior similar to Fig. 4 and cause an increased complexity for the attacker despite its relatively lower natural accuracy. Although hard to formally capture with the adaptive white-box attack evaluation, randomized filter cutoff introduces additional uncertainty for the attacker. Finally, we show in Fig. 9 in the Appendix that the attack results in an increase in the perturbation power received by individual models compared to the baseline.

Over-the-air Evaluation. We generate 90 adversarial examples from each of the PGD and PGD with RIR adaptive attacks against an ensemble of size $l = 5$. We play the adversarial examples over the air and capture the recordings from the commodity devices. We evaluate adversarial examples in a white-box setting, i.e., against the same exact models and feature filters that were used to generate them. However, the device-model assignment is done randomly. Thus, we repeat the evaluation ten times. Table 2 presents the average adversarial accuracy and confirms that EKOS' ensemble outperforms the individual devices against both attacks.

Next, we evaluate the attack against a randomized run of EKOS; i.e. the feature filters and KWS architectures are selected independently at random. The evaluation is repeated ten times. Fig. 5 shows the accuracy (mean and standard deviation) of individual devices and of EKOS's ensemble at size $l = 3, 5, 7$ for the adversarial examples and their benign samples as well. It is clear that all the devices' accuracy is higher than their values in Table 2 due to the randomized run.

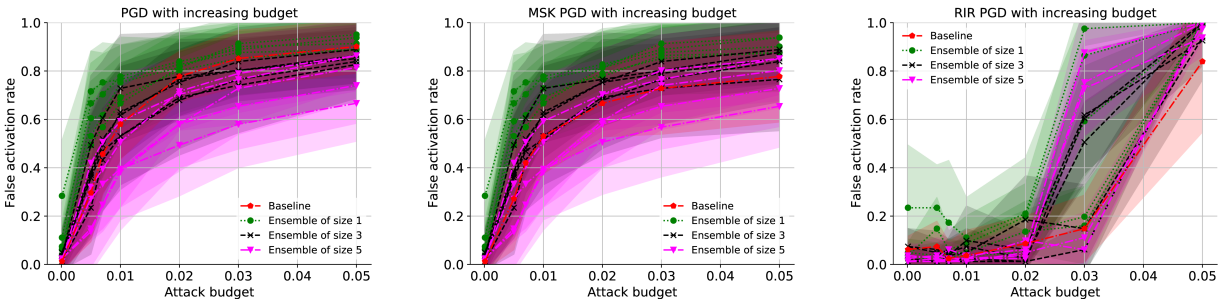
We observe from Fig. 5 that EKOS's ensemble outperforms individual devices on benign and adversarial samples. Although we perform feature slicing in this experiment, accuracy on benign samples matches that of EKOS without feature slicing (Fig. 2), especially at $l > 3$. This is consistent with our findings from the simulated setup (Fig. 3); the ensemble gain compensates for the accuracy drop due to feature slicing.

Although adversarial examples are successful in the simulated setup (the model's accuracy is 0), they do not always succeed over the air (accuracy > 0 in Table 2 and Fig. 5). Moreover, while the PGD with RIR attack takes the acoustic channel into consideration, its attack success rate (1-accuracy) is not always higher than the PGD attack (without RIR). We attribute these observations to multiple factors: (1) the RIR simulation is only an approximation of the physical acoustic channel; (2) there are other physical transformations not taken into account, such as the microphone's non-linearity and noise; and (3) the expectation over RIR optimization does not guarantee a successful perturbation across all RIR transforms (all devices and environments). These observations match the findings from Qin et al. [45], where their adversarial examples were successful only in a simulated environment.

6.1.4 System Integration Analysis

Finally, we assess EKOS's deployment in terms of devices integration and end-to-end latency. EKOS latency stems from two sources: (1) model inference and (2) vote communication and aggregation. EKOS is not sensitive to device synchronization errors since it combines votes, not signals. Since the ensemble models run simultaneously and independently, the first source is dominated by the slowest device-architecture pair. The latencies of EKOS's architectures are available in prior work [16] (Table 1 and 2) and range between 1.1ms and 10.1ms.[7] We measure the end-to-end latency ΔT by running an ensemble of size $l = 10$ on our set of devices; some devices run more than one model simultaneously. The setup is as follows: D1 runs three models, D4 runs two models, D5

[7] The inference time is measured on a Google Pixel 1 using the TensorFlow Lite Android benchmark tool. The authors forced the model to be executed on a single core in order to emulate the always-on nature of KWS.



(a) PGD (b) PGD with frequency masking (c) PGD with RIR EoT

Figure 4: False activation rate (%) of EKOS against 5 randomly selected ensembles of sizes $l = 1, 3, 5$ along-with a TCResNet8 baseline model under adversarial examples generated by PGD, PGD with frequency mask, and PGD with RIR attacks.

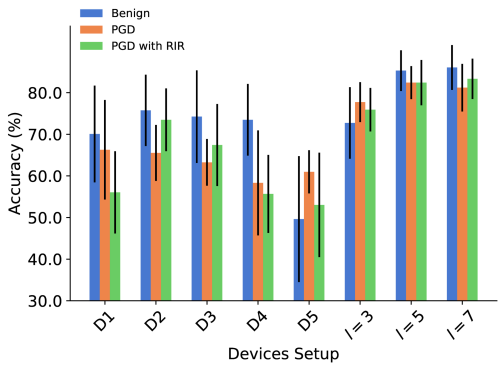


Figure 5: Over-the-air accuracy (mean and std) of EKOS under PGD and PGD-RIR attacks and their benign samples with random slicing filter and architecture selection.

and D6 run a single TensorFlow-Lite model each, and D3 (the main VA server) runs three models and aggregates the votes.

We performed 100 inference requests, the average latency ΔT is 0.32s±0.25s; the median, max, min are 0.21s, 1.53s, 0.20s, respectively. EKOS's latency ΔT is consistent with the latency window it takes the cloud KWS module to verify the local activation and to perform "Echo Spatial Perception[8]" to coordinate multiple Echo devices in the same environment. Hence, EKOS's latency does not degrade the user experience. Moreover, EKOS latency does not increase linearly with the number of devices; it is not accumulative. Thus, introducing more devices to EKOS will not necessarily increase its latency unless the new device forms a critical path.

6.2 Commercial Voice Assistants

In this section, we extend our evaluations of EKOS to commercial VAs and their keywords. This evaluation is challenging since we do not have access to their dataset, and there is no API access to the local KWS engine. Moreover, since the commercial models are not trained with feature slicing transformation, we cannot evaluate EKOS end-to-end; it is only feasible to assess the physical environment effect on accidental and adversarial activations. We do not apply any feature transformation or pre-processing on the evaluated keywords.

6.2.1 Experimental Setup

Our setup comprises 5 Echo devices: 4 Echo Dot (3^{rd} Gen), and one Echo tower (1^{st} Gen), distributed in a lab space (Fig. 6). The Bluetooth speaker is located in the middle of the room, and the Echo devices are located at 0.7, 3, 3, 2.7, and 2.6m away from the speaker. We choose Amazon's Echo devices because they can be activated by four different keywords: {*Alexa, Echo, Amazon, Computer*}, hence, enabling a comprehensive study. We automate the activation detection using a digital photosensitive sensor attached to the device's light rim. Once a device detects the keyword, its rim light turns on, and the sensor captures the change in light. The setup is controlled by a Raspberry Pi 4 Model B that plays the audio sample on the Bluetooth speaker and records the Echo devices' activations via the sensors' output.

We run the experiment on the local (offline) and the local+cloud (online) KWS models. Since the VAs operation is non-deterministic [23, 53], we repeat the offline (online) experiment three (ten) times and report the average values.

6.2.2 Evaluation Dataset

Positive Samples. We generate two sets of positive samples for each of the four keywords, namely Positive–TTS and Positive–Speech. In the first set, Positive–TTS, we use text-to-speech APIs from Google, Amazon, and IBM to generate 77 samples for each of the four keywords in different voices (while synthetic, the samples sound natural to the ear).

We extract the second set, Positive–Speech, from conventional speech recognition datasets – Librispeech, VCTK, Common Voice, TED-LIUM, and M-AILABS. We search the

[8] developer.amazon.com/blogs/alexa/post/esp

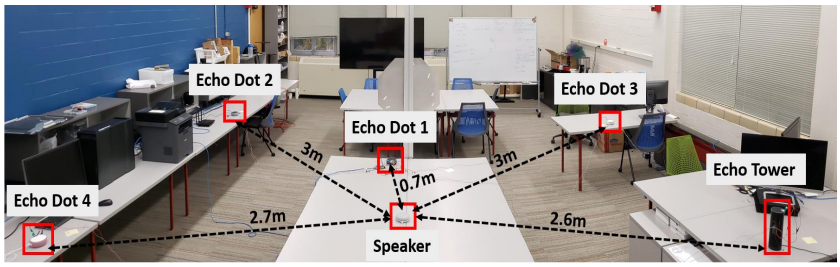


Figure 6: Commercial (Amazon's Echo) VAs setup of 4 Echo Dots, 1 Echo tower, and a Bluetooth speaker.

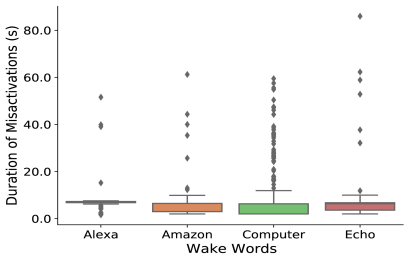


Figure 7: Duration (seconds) of Cloud-based Misactivations of Echo VAs per Keyword.

datasets transcription for the keywords. We use the Montreal Forced Aligner[9] to align the speech with its transcript and extract the keyword utterance. The Positive–Speech set has 104, 138, 405 samples for *Amazon*, *Echo*, and *Computer*, respectively. We found no samples for *Alexa*. Since these datasets were not manually validated, we curate them by discarding samples that do not activate any of the five devices.

Negative Samples. Generally speaking, any speech utterance other than the keyword is a negative sample. However, in this experiment, we focus on worst-case samples with a high probability of incorrectly activating the device. Thus, the false activation (FA) rates we report (e.g., in Table 3) are higher than expected normal operation values. We extract the accidental activation dataset that Dubois et al. [23] identified in TV shows. We also use Schönherr et al.'s crafted accidental activation triggers [53]. This dataset consists of n-gram English phrases that are phonetically close to the keyword. We use text-to-speech APIs to synthesis these phrases in different voices, as done for Positive–TTS samples.

Adversarial Examples. We use the adversarial examples generated by Devil's Whisper attack [15] on Amazon Echo[10], which is an over-the-air robust attack.

6.2.3 Accidental Activation Analysis

Local Activation. We evaluate the local KWS model performance by disconnecting the Echo devices from the Internet. When an utterance activates the local model, the rim light turns red, and the device plays an error message. Table 3 shows the true-positive (TP) and false-activation (FA) counts along with their class sizes for the individual devices and their ensemble. The ensemble decision is a majority vote; i.e., it is activated when at least 3 out of 5 devices are activated.

The devices closer to the speaker such as Echo Dot1, the closest device to the speaker, have high TP values and also high FA for all keywords, and vise versa such as Echo Dot4. Hence, the user's experience and privacy are at odds with respect to the device's proximity to the user. On the other hand, the majority vote ensemble has a lower number of misactivations (FA) than most of the devices. Table 4 in the Appendix shows the accuracy and F1 scores of the individual devices and their ensemble. The ensemble accuracy and F1 are higher than the five devices at all keywords except *Computer*. Hence, EKOS's ensemble achieves the two-fold objective: it protects the user's privacy with fewer FA without sacrificing the utility.

Cloud-based Activation. We reconnect the devices to the Internet to evaluate cloud KWS on both authorized and unauthorized samples. Table 3 shows that the numbers of FA are significantly lower than the local KWS model for almost all devices and keywords. Local activations not confirmed by the cloud model are transcribed on the voice history page with "*Audio was not intended for Alexa.*" Although the cloud KWS verification enhances the user experience by limiting unwanted and unexpected interactions with the user, it does not necessarily mitigate the privacy concerns: private conversations are still being sent to the cloud.

To quantify the privacy leakage, we analyze the misactivations duration, i.e., the time during which the rim light stays on. Fig. 7 shows the durations distribution per keyword. We find that the duration is concentrated from 1.6s to 10s with 6.33s median and 5.75s mean, with some samples reaching up to 86s. Hence, the misactivations are long enough to leak private conversations. The FA rate and the misactivation duration quantify the magnitude of privacy leakage.

We believe that our experiment could capture some worst-case misactivation durations, as compared to prior research [23]; the devices are distributed in a large room with uncontrolled background noise, unlike prior work [23] that places the VAs and the speaker in a small isolated cabinet.

The cloud TP values, however, are in the same range as the local KWS model. This observation is unsurprising since the positive activation is mainly controlled by the local model; the cloud model only confirms or discards the local activation. In other words, the false-negative (misdetection) rate, ($FNR = 1 - TPR$), cannot be improved by the cloud model, which explains why the local KWS model's design favors TPR rather than FPR. We attribute the small differences in the TP values between local and cloud models to the non-deterministic behavior of VAs and the uncontrolled environ-

[9] montreal-forced-aligner.readthedocs.io/en/latest/example.html
[10] github.com/RiskySignal/Devil-Whisper-Attack/tree/master/AEs

Devices	Alexa				Amazon				Computer				Echo			
	Offline		Online		Offline		Online		Offline		Online		Offline		Online	
	TP	FA	TP	FA	TP	FA	TP	FA	TP	FA	TP	FA	TP	FA	TP	FA
Echo Dot1	68	44	67	20	140	75	125	89	181	326	169	217	89	75	89	49
Echo Dot2	60	31	61	14	120	48	115	39	**208**	**136**	180	86	**79**	**23**	77	36
Echo Dot3	51	27	50	31	99	29	**111**	**22**	163	130	**172**	**55**	81	36	68	34
Echo Dot4	57	22	49	3	86	6	65	4	93	71	85	54	62	30	50	11
Echo tower	59	44	66	18	92	37	95	50	189	156	167	202	72	69	70	54
Ensemble	**61**	**21**	**63**	**9**	**113**	**23**	110	22	174	128	163	75	83	31	**77**	**24**
Class Size	*Pos:* 69 – *Neg:* 985				*Pos:* 147 – *Neg:* 808				*Pos:* 263 – *Neg:* 1738				*Pos:* 105 – *Neg:* 596			

Table 3: Amazon Echo's offline (local) and online (cloud) KWS performance at different keywords. TP = true-positive, FA = false-activation. *Pos:* the positive class sample size, and *Neg:* the accidental activation class sample size. The TP and FA values that result in the highest accuracy per keyword are displayed in **bold**.

mental variations in the lab, such as background noise.

Finally, similar to the local model, EKOS' ensemble outperforms the individual devices; it has a lower FA and a higher TP. Table 4 also shows that the ensemble accuracy and F1 scores are superior to most of the devices. Hence, the KWS ensemble is a practical solution that can enhance commercial VAs performance and preserve the user's privacy.

6.2.4 Adversarial Examples on Commercial VAs

We play Devil's Whisper 2 adversarial examples on the keyword *Echo*, which reports a 0.5 adversarial success rate (SR) in their original setup. Their adversarial SR on our setup (Fig. 6) is 0.7, 0, 0, 0, and 0.1 on the individual devices, and SR= 0 on the ensemble. We relocate the devices such that they are at a 1m distance from the speaker; the SR increases to 0.6, 0.7, 0.15, 0, and 0.9, respectively, and the ensemble SR= 0.45. Hence, the ensemble vote could reduce the adversarial activations as well, even without feature slicing and architecture diversity, in a non-adaptive attack setting.

7 Discussion

In the following, we discuss the tradeoffs in deploying EKOS as well as some limitations and future work directions.

7.1 Deployment Analysis

EKOS is a flexible system; it enables a custom-tailored operation with tunable utility, usability, and privacy trade-off.

Number of devices N: Ensemble models can be either centralized in the main VA or distributed over a set of smart devices in the environment. The former is a compact and, possibly, more usable setting. On the other hand, the latter minimizes the VA footprint via distributed computation and provides more spatial and hardware diversity. Thus, N is a control parameter the user can tune to optimize the robustness, utility, and computational cost. We evaluate EKOS on closely located microphones on a single VA in the appendix A.1.

Feature Slicing: The user can enable (disable) the feature slicing filters to include (exclude) the adversarial activation threat model. However, as shown in Fig. 3, 5, with $l \geq 5$, the ensemble with feature slicing enabled reaches the baseline natural accuracy without feature slicing (Table 1, Fig. 2). Also, the user can enable (disable) the random filter cutoffs shift for a more robust (accurate) operation – refer to Fig. 3.

Number of architectures K: As shown in Table 1 and Fig. 2, architecture diversity enhances the ensemble accuracy and robustness. Yet, channel diversity and feature slicing also contribute to EKOS' accuracy and robustness. Hence, the number of stored architectures can be set to only one without sacrificing EKOS's performance when device storage is limited.

Number of models l: As shown in Table 1 and Fig. 2, 3, the optimal number of models ranges from 3 to 5. Hence, EKOS is a practical system, it enhances the VA robustness at an overhead of $l \leq 5$ models deployed on $N \leq l$ devices.

Computation Overhead: EKOS adds computation overhead to the commodity devices. However, the client devices run the KWS model only when they receive a request from the main VA (Sec. 5.1). Hence, the computation overhead will be limited to infrequent true-positive and false-positive incidents. Moreover, the centralized mode of operation only loads the main VA, which is a plug-in device.

7.2 Limitations and Future Research

Distance Bounding. A device far from the user may increase the false-negative rate due to its low SNR. Hence, the ensemble devices should perform a periodic distance bounding check. Although we do not include this component in this work, distance bounding is a well-studied problem with existing engineering and cryptographic solutions [6, 41].

Other Robustness Measures. EKOS can be integrated with other measures to increase robustness such as adversarial training and data augmentation. Another interesting integration is to perform voice recognition where the VA only accepts commands that match the authorized user voiceprint.

Voice recognition filters out accidental and adversarial activations from unauthorized speakers, TV, YouTube videos, etc. An adversary would need to attack both systems jointly to initiate a successful mis-activation . Hence, voice recognition will increase the attack cost and enhance the robustness.

Other Attack Models. EKOS does not directly address attacks based on the microphone non-linearity such as light commands [56] and ultrasound signals [47, 67], yet, it increases their cost. For example, the light attack will require at least $l/2+1$ laser beams to target the majority of the devices simultaneously. Likewise, the ultrasound attack needs to be performed at close proximity to the microphones with speaker-microphone aligned orientation. Hence, the distributed ensemble will increase this attack cost as well.

Replay and spoofing attacks can fool the KWS using recorded or synthetic speech commands that contain the correct keyword. To address them, EKOS can incorporate a voice-liveness detector [3, 25] to distinguish human and machine-generated commands. We envision a hierarchical detection algorithm that rejects any non-human command.

8 Related Work

Privacy Measures for KWS. Cloud operators provide some privacy measures to minimize accidental activations. For example, VA providers give their users access to their voice commands history, where they can listen to and delete their past commands [20]. After the public backlash on manual transcription of voice commands, Google, Apple, Amazon, and Facebook suspended the default enrollment and are currently giving the users opt-in and opt-out choices [31]. Recently, Google has enabled a tunable keyword sensitivity setting to give the users control over the utility and privacy tradeoff [34]. Yet, these measures do not address the privacy threats of VA misactivations and do not meet the users' expectations of hands-free interactions. Recently, Apple has announced [28] that Siri will process speech locally on the user's device to mitigate these privacy concerns.

Researchers have also proposed external privacy control systems to reduce misactivation incidents. Karmann proposes a small add-on device that jams the VA's microphones and lifts up the jamming when it detects a user-customized keyword [30]. However, it suffers from the same privacy threats of KWS misactivation. Wu et al. [64] propose an audio-visual speech recognition by analyzing lip movement in a sensor-fusion algorithm. Similarly, Mhaidli et al. [42] use the gaze direction and voice volume level as interpersonal communication cues of the user's intent to activate the VA. These efforts, however, introduce another privacy threat by adding an always-on camera in the user's private environment. Coucke et al. propose Snips, a private-by-design system [18] that processes the commands locally without cloud interactions. Yet, it cannot be integrated into the commercial VAs.

Adversarial Example Defenses. Generic defenses like adversarial training [38] and verifiable robustness [17, 32, 50] are largely detached from the real world; they assume the adversarial capabilities to respect an ℓ_p-norm ball constraint. Yet, attacks in the physical space can map to large ℓ_p-norm perturbations in the digital space. Moreover, these approaches are hard to train, and come with significant performance loss limiting their adoption. Others have explored defenses specific to the audio domain. Bhattacharya et al. [7] propose a stochastic compression technique. Liveness detection distinguishes commands coming from a human or an audio speaker, either via spectrum analysis [3, 25] or motion sensing [33], or detecting the audio speaker magnetic field [14]. This technique introduces additional privacy threats of sensing human activities and is ineffective against accidental activations.

9 Conclusion

We took two complementary approaches to tackle two forms of misactivations: accidental and adversarial. Both approaches rely on a diversity of sensors and models used to perceive the environments. However, they differ in that adversarial activations require provable guarantees of increased costs, which we achieve by exploiting a physical property of the audio wave (replicas of speech at different harmonics) rather than taking an ML approach. We hope this paper paves the way for a new class of approaches that systematically characterize environmental constraints to improve ML robustness.

Acknowledgement

This work was supported by DARPA (through the GARD program), the NSF through awards: CNS-1838733, CNS-1942014, and CNS-2003129, CIFAR (through a Canada CIFAR AI Chair), by NSERC (under the Discovery Program, and COHESA strategic research network), by a gift from Intel, and by a gift from the NVIDIA Corporation. We also thank the Vector Institute's sponsors. Ilia Shumailov was supported by Bosch Forschungsstiftung im Stifterverband. Finally, we would like to thank the anonymous reviewers for their useful comments and Yuan Tian for shepherding this paper.

References

[1] Noura Abdi, Kopo M Ramokapane, and Jose M Such. More than smart speakers: security and privacy perceptions of smart home personal assistants. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*, 2019.

[2] Hadi Abdullah, Kevin Warren, Vincent Bindschaedler, Nicolas Papernot, and Patrick Traynor. The faults in our asrs: An overview of attacks against automatic speech

recognition and speaker identification systems. *IEEE Symposium on Security and Privacy*, 2021.

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

[4] Jont B Allen and David A Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, 1979.

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

[6] Gildas Avoine, Muhammed Ali Bingöl, Ioana Boureanu, Srdjan Čapkun, Gerhard Hancke, Süleyman Kardaş, Chong Hee Kim, Cédric Lauradoux, Benjamin Martin, Jorge Munilla, et al. Security of distance-bounding: A survey. *ACM Computing Surveys (CSUR)*, 51(5):1–33, 2018.

[7] Sourav Bhattacharya, Dionysis Manousakas, Alberto Gil CP Ramos, Stylianos I Venieris, Nicholas D Lane, and Cecilia Mascolo. Countering acoustic adversarial attacks in microphone-equipped smart home devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(2):1–24, 2020.

[8] Miles Brignall. Amazon hit with major data breach days before black friday. https://tinyurl.com/yyu4mmj8, 2018.

[9] Joseph Bugeja, Andreas Jacobsson, and Paul Davidsson. On privacy and security challenges in smart connected homes. In *2016 European Intelligence and Security Informatics Conference (EISIC)*, pages 172–175. IEEE, 2016.

[10] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks, 2019.

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

[12] Varun Chandrasekaran, Suman Banerjee, Bilge Mutlu, and Kassem Fawaz. Powercut and obfuscator: An exploration of the design space for privacy-preserving interventions for smart speakers. In *Seventeenth Symposium on Usable Privacy and Security ({SOUPS} 2021)*, pages 535–552, 2021.

[13] Guoguo Chen, Carolina Parada, and Georg Heigold. Small-footprint keyword spotting using deep neural networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4087–4091. IEEE, 2014.

[14] S. Chen, K. Ren, S. Piao, C. Wang, Q. Wang, J. Weng, L. Su, and A. Mohaisen. You can hear but you cannot steal: Defending against voice impersonation attacks on smartphones. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 183–195, 2017.

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

[16] Seungwoo Choi, Seokjun Seo, Beomjun Shin, Hyeongmin Byun, Martin Kersner, Beomsu Kim, Dongyoung Kim, and Sungjoo Ha. Temporal convolution for real-time keyword spotting on mobile devices. *arXiv preprint arXiv:1904.03814*, 2019.

[17] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[18] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*, 2018.

[19] Joseph Cox. Amazon fired employee for leaking customer emails. https://tinyurl.com/hvv36tcj, 2020.

[20] Ry Crist and Andrew Gebhart. Everything you need to know about the amazon echo. https://tinyurl.com/r2ura464, 2018.

[21] Matt Day, Giles Turner, and Natalia Drozdiak. Amazon workers are listening to what you tell alexa. https://tinyurl.com/audw32jw, 2019.

[22] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[23] Daniel J Dubois, Roman Kolcun, Anna Maria Mandalari, Muhammad Talha Paracha, David Choffnes, and Hamed Haddadi. When speakers are all ears: Characterizing

misactivations of iot smart speakers. *Proceedings on Privacy Enhancing Technologies*, 2020(4):255–276, 2020.

[24] Jide S Edu, Jose M Such, and Guillermo Suarez-Tangil. Smart home personal assistants: a security and privacy review. *arXiv preprint arXiv:1903.05593*, 2019.

[25] Y. Gong and C. Poellabauer. Protecting voice controlled systems using sound source identification based on acoustic cues. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9, 2018.

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

[27] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*, 2017.

[28] Alex Hern. Smart speaker statistics. https://tinyurl.com/3nnu6n84, 2021.

[29] Jean-Marc Jot, Laurent Cerveau, and Olivier Warusfel. Analysis and synthesis of room reverberation based on a statistical time-frequency model. In *Audio Engineering Society Convention 103*. Audio Engineering Society, 1997.

[30] Bjørn Karmann. Project alias. https://bjoernkarmann.dk/project_alias, 2018.

[31] Ravie Lakshmanan. Apple and google suspend monitoring of voice recordings by humans (update: Amazon too). https://tinyurl.com/y34hjv3c, 2019.

[32] Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE S&P 2019*, 2018.

[33] Xinyu Lei, Guan-Hua Tu, Alex X. Liu, Chi-Yu Li, and Tian Xie. The insecurity of home digital voice assistants - amazon alexa as a case study. *CoRR*, abs/1712.03327, 2017.

[34] Abner Li. 'hey google' sensitivity setting now official, gradually rolling out. https://9to5google.com/2020/04/23/hey-google-sensitivity/, 2020.

[35] Juncheng Li, Shuhui Qu, Xinjian Li, Joseph Szurley, J Zico Kolter, and Florian Metze. Adversarial music: Real world audio adversary against wake-word detection system. In *Advances in Neural Information Processing Systems*, pages 11908–11918, 2019.

[36] Chavie Lieber. Amazon's alexa might be a key witness in a murder case. https://tinyurl.com/4v6te95z, 2018.

[37] Dorian Lynskey. 'alexa, are you invading my privacy?' – the dark side of our voice assistants. https://tinyurl.com/y5velcwz, 2019.

[38] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[39] Nathan Malkin, Joe Deatrick, Allen Tong, Primal Wijesekera, Serge Egelman, and David Wagner. Privacy attitudes of smart speaker users. *Proceedings on Privacy Enhancing Technologies*, 2019(4):250–271, 2019.

[40] Leena Mary and G Deekshitha. *Searching Speech Databases: Features, Techniques and Evaluation Measures*. Springer, 2018.

[41] S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua. Distance-bounding protocols: Verification without time and location. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 549–566, 2018.

[42] Abraham Mhaidli, Manikandan Kandadai Venkatesh, Yixin Zou, and Florian Schaub. Listen only when spoken to: Interpersonal communication cues as smart speaker privacy controls. *Proceedings on Privacy Enhancing Technologies*, 2020(2):251–270, 2020.

[43] Yuantian Miao, Minhui Xue, Chao Chen, Lei Pan, Jun Zhang, Benjamin Zi Hao Zhao, Dali Kaafar, and Yang Xiang. The audio auditor: User-level membership inference in internet of things voice services. *Proceedings on Privacy Enhancing Technologies*, 2021(1):209–228, 2020.

[44] Assaf Hurwitz Michaely, Xuedong Zhang, Gabor Simko, Carolina Parada, and Petar Aleksic. Keyword spotting for google assistant using contextual speech recognition. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 272–278. IEEE, 2017.

[45] Yao Qin, Nicholas Carlini, Ian Goodfellow, Garrison Cottrell, and Colin Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. *arXiv preprint arXiv:1903.10346*, 2019.

[46] Anirudh Raju, Sankaran Panchapagesan, Xing Liu, Arindam Mandal, and Nikko Strom. Data augmentation for robust keyword spotting under playback interference. *arXiv preprint arXiv:1808.00563*, 2018.

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

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

[49] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. Updates-leak: Data set inference and reconstruction attacks in online learning, 2019.

[50] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems*, pages 11292–11303, 2019.

[51] Robin Scheibler, Eric Bezzam, and Ivan Dokmanić. Pyroomacoustics: A python package for audio room simulation and array processing algorithms. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 351–355. IEEE, 2018.

[52] Lea Schönherr, Thorsten Eisenhofer, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Imperio: Robust over-the-air adversarial examples for automatic speech recognition systems. *arXiv preprint arXiv:1908.01551*, 2019.

[53] Lea Schönherr, Maximilian Golla, Thorsten Eisenhofer, Jan Wiele, Dorothea Kolossa, and Thorsten Holz. Unacceptable, where is my privacy? exploring accidental triggers of smart speakers. *arXiv preprint arXiv:2008.00508*, 2020.

[54] Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson. Sponge examples: Energy-latency attacks on neural networks. In *6th IEEE European Symposium on Security and Privacy (EuroS&P)*, 2021.

[55] Siddharth Sigtia, Rob Haynes, Hywel Richards, Erik Marchi, and John Bridle. Efficient voice trigger detection for low resource hardware. In *Interspeech*, pages 2092–2096, 2018.

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

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

[58] Raphael Tang and Jimmy Lin. Deep residual learning for small-footprint keyword spotting. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5484–5488. IEEE, 2018.

[59] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

[60] Stephen B Vardeman and Max D Morris. Majority voting by independent classifiers can increase error rates. *The American Statistician*, 67(2):94–96, 2013.

[61] Tim Verheyden, Denny Baert, Lente Van Hee, and Ruben Van Den Heuvel. Google employees are eavesdropping, even in your living room, vrt nws has discovered, 2019.

[62] Chen Wang, S Abhishek Anand, Jian Liu, Payton Walker, Yingying Chen, and Nitesh Saxena. Defeating hidden audio channel attacks on voice assistants via audio-induced surface vibrations. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, page 42–56, New York, NY, USA, 2019. Association for Computing Machinery.

[63] Pete Warden. Launching the speech commands dataset. Available: https://tinyurl.com/yy4sa92z, August, 2017.

[64] Pingping Wu, Hong Liu, Xiaofei Li, Ting Fan, and Xuewu Zhang. A novel lip descriptor for audio-visual keyword spotting based on adaptive decision fusion. *IEEE Transactions on Multimedia*, 18(3):326–338, 2016.

[65] Almos Zarandy, Ilia Shumailov, and Ross Anderson. Hey alexa what did i just type? decoding smartphone sounds with a voice assistant, 2020.

[66] Eric Zeng, Shrirang Mare, and Franziska Roesner. End user security and privacy concerns with smart homes. In *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*, pages 65–80, 2017.

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

Devices	Alexa				Amazon				Computer				Echo			
	Offline		Online		Offline		Online		Offline		Online		Offline		Online	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Echo Dot1	95.73	75.15	97.92	86.05	91.45	77.42	88.36	69.17	79.63	47.03	84.46	52.44	87.07	66.12	90.83	73.55
Echo Dot2	96.20	75.00	97.97	85.10	92.11	76.06	92.6	76.46	**90.44**	**68.48**	91.53	67.94	**93.06**	**76.56**	90.8	70.56
Echo Dot3	95.70	69.24	95.33	67.24	91.97	72.09	**93.97**	**79.31**	88.51	58.63	**92.70**	**70.10**	91.39	72.68	89.86	65.72
Echo Dot4	96.77	77.15	97.84	81.10	92.98	72.02	91.04	60.14	87.97	43.59	88.37	42.07	89.63	63.04	90.67	60.55
Echo tower	94.88	68.68	97.92	85.75	90.33	66.57	89.32	65.00	88.51	62.16	85.12	53.09	85.4	58.68	87.35	61.42
Ensemble	**97.19**	**80.40**	**98.57**	**89.32**	**94.03**	**79.85**	93.8	78.65	89.16	61.54	91.23	64.96	92.49	75.94	**92.57**	**74.8**
Class Size	*Pos:* 69 – *Neg:* 985				*Pos:* 147 – *Neg:* 808				*Pos:* 263 – *Neg:* 1738				*Pos:* 105 – *Neg:* 596			

Table 4: Amazon Echo's offline (local) and online (cloud) KWS performance at different keywords in terms of the accuracy and F1 scores (%). *Pos:* the positive class sample size, and *Neg:* the accidental activation class sample size. The highest Acc and F1 scores per keyword are displayed in **bold**.

A Appendix

A.1 Circular Microphones Spatial diversity

Here, we evaluate EKOS in the setting of a single centralized VA that has $m = 4$ or $m = 7$ microphones — similar to commercial VA devices. We use the circular microphone array model from *Pyroomacoustics* package to generate the RIR at closely located microphone in a circle of 5cm radius, representing the device board.

Table 5 shows the ensemble natural accuracy of EKOS's pipeline at $l = m = \{4, 7\}$, where l is the number of ensemble models. We assign a model to each microphone, and apply inference time randomness: (1) filter selection and (2) filter cutoffs random shift within $\pm 200Hz$. As the table shows, the ensemble accuracy is only slightly lower than the values at Fig.3. We observe that the 4 microphones case is slightly better than 7 microphones, probably because they experience more spatial diversity than the 7 microphones setup leading to a higher majority vote accuracy.

Architecture	Random filters		Random cutoff shift	
	$m = 4$	$m = 7$	$m = 4$	$m = 7$
TC-ResNet8	91.32	87.34	77.02	78.79
TC-ResNet14	92.06	93.04	78.64	77.19
TC-ResNet2D8	90.24	87.51	85.71	82.85
DS-CNN-M	89.18	89.90	74.84	74.56
Random Arch.	89.20	89.10	79.208	84.846

Table 5: EKOS's accuracy (%) when deployed on a single VA that has m microphones and $l = m$ models, at different architectures and run-time randomness.

A.2 Perturbation imperceptibility

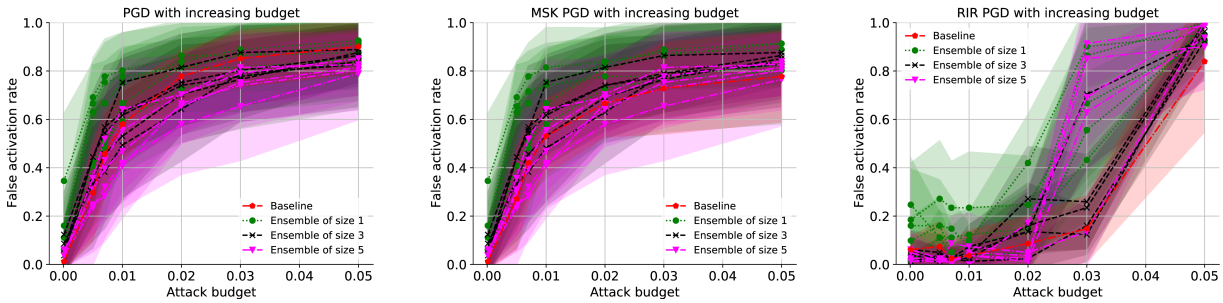
We investigate the relationship between imperceptibility of the attack and the ensemble size l. In EKOS, a successful attack needs to trick $\frac{l}{2} + 1$ models. Thus, increasing the number of models l requires a higher perturbation size to reach the same attack success rate (false activation rate) as shown in Fig. 9.

A.3 Adversarial Examples Transferability between different slice-architecture combinations

Figure 10 shows the transferability map of the PGD attack with 100 epochs and 0.05 perturbation budget across 9 architectures and 9 filters. Note that with this budget, models usually reach near-random guess performance.

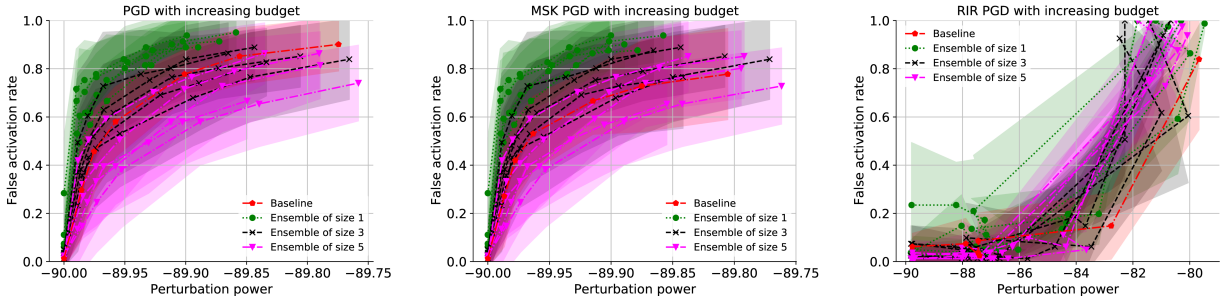
A.4 Commercial Voice Assistants

Table 4 shows the performance of the setup in Fig. 6 in terms of the accuracy and F1 scores (%). The ensemble accuracy is higher than the individual devices at all keywords except *Computer*. For the keyword *Echo*: The ensemble accuracy is very close to Echo Dot2 and is higher than the other four devices.



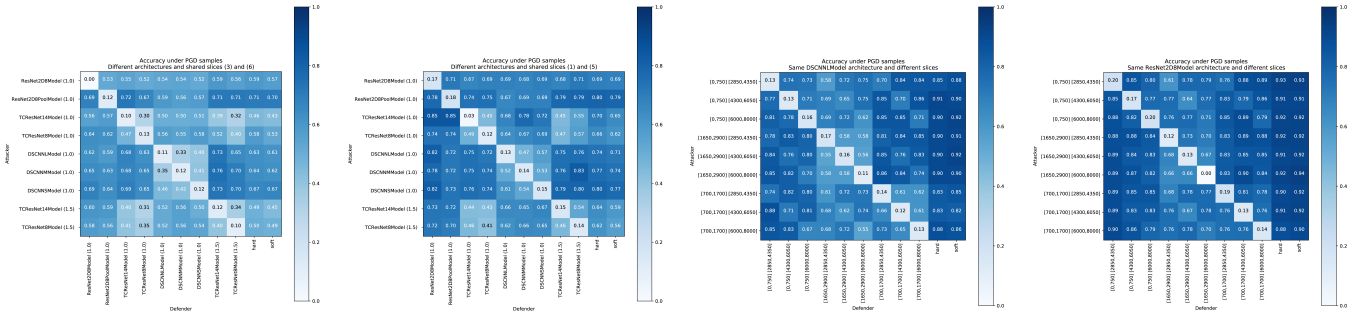
(a) PGD (b) PGD with frequency masking (c) PGD with RIR EoT

Figure 8: False activation rate (%) of EKOS at 5 randomly selected ensembles of sizes $l = 1,3,5$ with $\pm 200Hz$ random cutoff shift, along-with a TCResNet8 baseline model, under adversarial examples generated by (a) PGD, (b) PGD with frequency mask, and (c) PGD with RIR attacks.



(a) PGD (b) PGD with frequency masking (c) PGD with RIR EoT

Figure 9: False activation rate (%) versus the average perturbation power (dB_w) received by EKOS, at ensembles of sizes $l = 1,3,5$ along-with a TCResNet8 baseline model under (a) PGD, (b) PGD with frequency mask, and (c) PGD with RIR attacks.



(a) Filter Slice 3 and 6 (b) Filter Slice 1 and 5 (c) DS-CNN-L Model (d) ResNet2D Model

Figure 10: Transferability of PGD with 100 epochs and 0.05 perturbation budget across (a,b) 9 architectures with shared filter slices, and (c,d) across 9 filters with a single shared architecture.