CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Adversarial training to prevent wake word jamming in Personal Voice Assistants
Authors	Sagi, Prathyusha;Sankar, Arun;Roedig, Utz
Publication date	2024-05
Original Citation	Sagi, P., Sankar, A. and Roedig, U. (2024) 'Adversarial training to prevent wake word jamming in Personal Voice Assistants', 20th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT 2024), Abu Dhabi, United Arab Emirates, 29 April - 1 May 2024. https://doi.org/10.1109/DCOSS-IoT61029.2024.00018
Type of publication	Conference item
Link to publisher's version	10.1109/DCOSS-IoT61029.2024.00018
Rights	© 2024, IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Download date	2026-11-17 11:52:19
Item downloaded from	https://hdl.handle.net/10468/15924

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Adversarial Training to Prevent Wake Word Jamming in Personal Voice Assistants

Prathyusha Sagi
School of Computer Science and Information Technology (CSIT)
University College Cork
Cork, Ireland
p.sagi@cs.ucc.ie

Arun Sankar
Department of Electronic Engineering and Communications
South East Technological University
Carlow, Ireland
arun.sankar@setu.ie

Utz Roedig
School of Computer Science and Information Technology (CSIT)
University College Cork
Cork, Ireland
u.roedig@cs.ucc.ie

***Abstract*—Wake word detection algorithms in Personal Voice Assistants (PVAs) are not designed to handle acoustic Denial of Service (DoS) attacks. We show that adversarial training can be used to improve the resilience of wake word detection against jamming attacks. We demonstrate that the inclusion of jammed wake word samples (adversarial samples) in the training phase of a wake word detection algorithm can defeat jamming attacks. The careful selection of the jamming signal type used during training ensures that wake word recognition is also resilient against jamming signals unknown during training; defeating a priori unknown jamming signal types is possible. We optimize the adversarial training effort by identifying areas of the wake word that are highly susceptible to acoustic interference, which guides our generation of adversarial training samples. We demonstrate the success of the proposed approach using a variety of wake words and two different wake word detection algorithms.**

***Index Terms*—Personal Voice Assistant (PVA), Wake Word Detection, Acoustic Jamming, Adversarial Training.**

I. Introduction

PVAs are deployed as standalone devices such as the Amazon Echo and Google Home or are integrated within electronic devices such as phones, Personal Computer (PC), and TVs. Increasingly PVAs are also found in safety-critical systems. For example, they are used to interact with your car and are found in surgical theaters to assist doctors, and have been used in the military as well [1]–[3]. As we come to rely on PVAs it is important to pay more attention to their resilient operations. A PVA continuously records sound in the neighborhood and is triggered by the detection of a *wake word*. Wake word detection is performed by a simplified Automatic Speech Recognition (ASR) within the PVA. Upon detection, the PVA records the following sound which is transported to a cloud-based backend where advanced ASR is used. A wake word detection failure prevents PVA usage.

Wake word detection can be impeded by acoustic interference and an adversary can use jamming to prevent the use of a PVA altogether leading to a DoS attack (see [4]–[6]). Commercial PVAs employ sophisticated noise cancellation methods, however, these are insufficient to suppress deliberate jamming efforts. Existing works have investigated wake word jamming using various audible and inaudible sound sources (see [4]–[7]) but there is a lack of defense mechanisms. Some existing work has focused on improving wake word detection in noisy environments (see [8], [9]) which may also help in defending against attacks. However, in this work, we focus on improving the resilience of wake word detection specifically against jamming attacks.

We consider a jamming signal (initially white noise) of varying duration (2ms up to 256ms) and power and slide this jamming signal window over the wake word. Thus, we obtain insight on the effectiveness of jamming signals. Using this analysis we identify *sensitive regions*, areas of the wake word signal that are very susceptible to jamming. We create *adversarial samples* by combining jamming signals of varying duration and power with wake word samples. To improve training efficiency we only consider adversarial samples where the jamming signal falls in the identified sensitive areas. We use the adversarial samples to improve wake word detection. Our experiments show that a model trained with adversarial samples is resilient to jamming even if the jamming signal used for training differs during an attack.

Our work uses the public available wake word detection algorithm by Supriya et al. [10] which we refer to as *LSTM-WD*. We also verified that the presented results apply to another algorithm called *MHAtt-RNN* designed by Oleg et al. [9]. We used the wake word 'Activate' and verified that results also apply to other wake words such as 'Alexa' and 'Marvin'. Thus, we believe that adversarial training to prevent wake word jamming presented in this paper is generically applicable.

The specific contributions of our work are:

1) *Wake Word Jamming*: We provide a comprehensive study of the effectiveness of acoustic jamming signals on wake word recognition algorithms. We show that a well placed jamming signal of only 2ms duration can prevent wake word detection and thus PVA operations entirely.
2) *Adversarial Training for Robust Wake Word Detection*: To the best of our knowledge, this is the first proposal to make a wake word detection more robust by using an adversarial training process.
3) *Adversarial Sample Selection*: We identify a suitable signal (i.e. 'babel noise') for adversarial sample generation such that attacks with a priori unknown signals are

detected. We identify which adversarial samples should be considered to reduce the training effort.

Next, we discuss related work. Section III describes the wake word detection algorithms and Section IV details the wake word jamming process. Section V outlines our adversarial training approach. We present detailed results using the LSTM-WD algorithm in Section V-B. In Section V-E we generalize our findings and report on experiments using a broader set of wake word detection algorithms and wake words. Section VI concludes the paper.

II. Related Work

Acoustic interference, in the form of background noise or a deliberate jamming signal, can cause interruption to wake word detection. Wake word jamming can be used not only as a means to conduct DoS attacks but also for consent management.

A Protection Jamming Device (PJD) is used in Peng et al [5] to prevent the PVA from recording conversations of the users in its vicinity. The PJD continuously monitors the channel for the wake word and emits the jamming signal when the wake word is spoken. By timing the jamming signal correctly, they could effectively prevent successful wake word detection. The study investigated the required signal overlap between the jamming signal and the wake word for successful jamming and examined the use of different sounds. Ke et al [4] look at jamming the acoustic channel to protect speech privacy from always-on microphones. This study uses a device called Micsheild that continuously emits a jamming signal to 'deafen' the PVA while it searches for wake word's phoneme pattern. Micsheild stops emitting the jamming signals as soon as it hears a wake word. The jamming signal is white noise in the 0-4kHz range. Li et al [6] describe a DoS attack on PVAs by jamming wake word. They employed an adversarial music synthesizer to generate guitar-like sounds for jamming purposes. The experiments showed that a jamming signal with a volume above 40dBA had an increased success rate in disrupting wake word detection. Continuous and long-duration jamming is very noticeable, so researchers investigated the use of inaudible sound sources in Guoming et al [7].

These differ from the work presented here. These works do not investigate how to improve wake word detection when jamming is present. In addition, specifics of the jamming signal such as duration, bandwidth, and signal strength are not thoroughly considered. In most works, continuous jamming signals are considered and less thought is given on how an attacker might create a short and efficient signal.

Some existing work has also aimed to improve the resistance of wake word detection against background noise. For example, work by Lim et al. [8] has proposed methods to improve the robustness of wake word detection against noise, using a cross-informed domain adversarial training by the inclusion of noisy wake word samples within the training process. Yixin et al. [11] proposes mixed-condition training, with clean speech, clean+reverb, and clean+reverb+noise speech samples to improve wake word detection in noisy settings. Qiuchen et al [12] proposed training the wake word model with utterances, where for each training utterance, to a random selection of frames time and frequency masking is applied to simulate noisy or interrupted audio conditions.

While these works look at wake word detection in a noisy environment or to improve overall wake word detection performance by masking techniques, they do not consider an attack in the form of deliberate jamming.

III. Wake Word Detection Algorithm

Wake word detection algorithms differ from Speech Recognition (SR) algorithms used in the back end. Wake word detection only has to detect one specific word, the wake word. Commercial ASR systems such as Alexa, Google and Siri use Deep Neural Network (DNN) based detection algorithms such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) or a combination of both (C-RNN) along with classifiers to predict the presence of a wake word. However, the implementation or specifics of commercial wake word detection algorithms are not publicly available. Our work is based on the public available wake word detection algorithm by Supriya et al. [10] which we refer to as *LSTM-WD*. The algorithm follows the approach, commercial products claim to be using. We verified our results using another public available algorithm by Oleg et al. [9] which uses a Multihead Attention RNN (referred to as *MHAtt-RNN*). The generic aspects of the wake word detection process are outlined in Fig. 1

A. LSTM-WD Wake Word Detection

The *LSTM-WD* by Supriya et al. [10] used in this study consists of a one-dimensional convolutional layer connected to two Gated Recurrent Unit (GRU) layers and a final dense+ sigmoid layer to predict the wake word. The model works on a 10-second audio input (see top of Fig. 1). The input audio is divided into equal length windows of duration 10ms, and a spectrogram is generated for each window. These spectrograms are then passed through a one dimensional convolutional layer to extract features from the audio. These extracted features are fed sequentially to the two GRU layers (GRU is a type of RNN) followed by a dense layer with sigmoid activation function to make predictions for each time step. The sigmoid function provides an estimate of the probability of the output to decide whether the spoken word is a wake word or not (bottom of Fig. 1).

The model produces a probability score for each time step of duration 7.2ms. The input audio of 10s (top of Fig. 1) generates 1375 time steps (10s/7.2ms) (bottom of Fig. 1). To determine if the audio input contains the wake word, a threshold of 0.5 is set. If any of the 1375 probability scores exceed this threshold the wake word is present.

We used the wake word 'Activate'. The training data set comprised 4000 samples each of duration 10s. Each training sample contains wake words, non wake words and background noise. Each wake word and non wake word are of average duration 700ms and 500ms respectively.

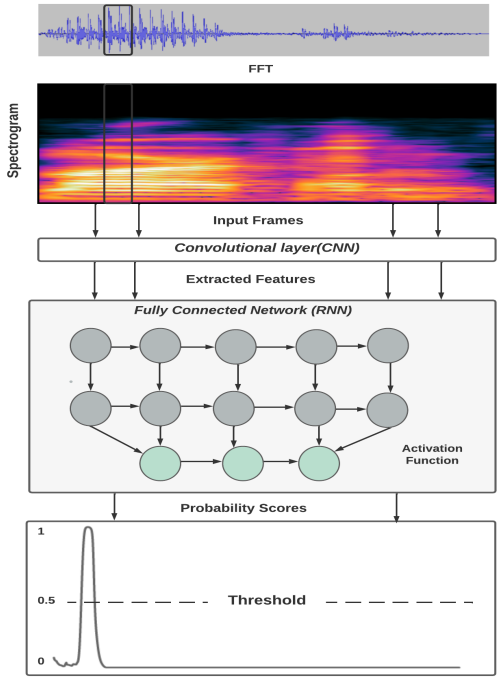


Fig. 1: The wake word detection process of *LSTM-WD*. The audio signal (top) is divided into frames of 10ms and features from each spectrogram are extracted and fed into an RNN. The resulting probability scores are compared to a threshold to decide on wake word presence (bottom).

B. MHAtt-RNN Wake Word Detection Model

We also used another algorithm referred to as *MHATT-RNN* by Oleg et al. [9], which was originally trained for keyword spotting. The model architecture consists of a speech feature extractor and a neural network based classifier similar to the standard architecture for keyword spotting Sercan et al. [13]. The input audio signal is divided into overlapping frames of length 40ms with an overlap of 20ms. Each of these frames is fed to a CNN to extract the MFCC features of the audio. the extracted features are fed to multi-head attention RNN layers with 4 heads. The RNN layers used here are GRU layers followed by a dense layer with a softmax activation function.

We have taken the same model architecture and retrained the model to predict the presence of wake word in an incoming audio signal. We trained the model for the wake word 'Marvin'. The dataset is split into 80% of training data with 1293 samples and 20% of test data with 320 samples with an average duration of approximately 700ms. The dataset is taken from the speech commands dataset from [14].

IV. Wake Word Jamming

A. Threat Model

Voice assistants rely on wake word detection algorithms to wake up and process user requests. If the wake word detection algorithm can be interrupted, then it disrupts the voice assistant's ability to recognize the wake word and process user requests, resulting in a DoS attack. Existing research has explored jamming for privacy protection and DoS attacks [4]–[6] using continuous jamming signals. In this work, we focus on a more advanced selective and efficient jamming approach to target the wake word detection algorithm to cause a DoS attack while remaining undetected.

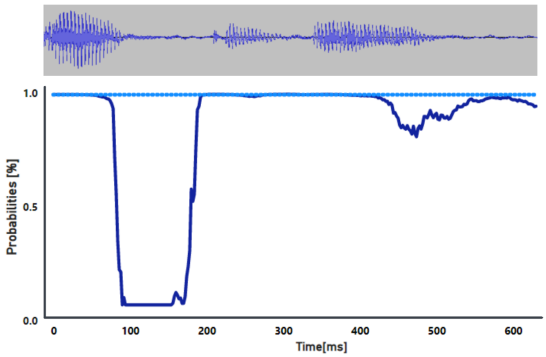


Fig. 2: Variation in wake word detection probability with and without superimposing white noise over the word 'Activate'. For every noise position across the wake word, the maximum probability score of LSTM-WD is given.

To successfully execute wake word jamming, an attacker would like to ensure the jamming signal is unnoticeable. Therefore, an attacker would like to limit the jamming signal in terms of duration, frequency and power. Furthermore, the attacker may design and time a jamming signal such that it is least noticeable while having a good impact on the wake word detection.

B. Jamming Signal

We initially consider *White Noise* as the jamming signal as it has a flat spectrum and a uniform impact on all frequency regions of the wake word. We also consider *Ping Noise* and *Babel Noise* as jamming signals. The noise dataset of ping noise is sourced from [15] and it has been segmented into desired durations. Babel noise is generated manually and includes background chatter, keyboard typing and cutlery clinking, sourced from [16] is then segmented into desired durations. Babel noise was chosen for robust adversarial training as it contains all possible sound types for jamming. We use the jamming signal types limited to the frequency band from 0kHz to 8kHz with a variable duration from 2ms to 256ms and variable power. The initial power is set at a Signal to Noise Ratio (SNR) of 0dB where both, the wake word and jamming signal are of equal strength. The jamming signal energy is increased by a factor until jamming is successful at a position of the wake word or the jamming signal energy exceeds a limit (500 times the energy of the wake word region over which the jamming signal is superimposed).

C. Jamming Methodology

Jamming is implemented by superimposing the jamming signal with the wake word. For experimentation, we slide the jamming signal of a selected duration and power in steps of

1ms over the entire wake word. For each jamming position, we evaluate the output of the LSTM-WD algorithm (1375 probability scores). If all of the 1375 scores are below the threshold of 0.5 jamming was successful. Fig. 2 shows the highest of the 1375 probability scores for each jamming signal position as purple curve (white noise, jamming duration of 32ms with 0dB). The blue line shows the same output when no jamming signal is applied. On top of Fig. 2 we show the speech signal of the used wake word 'Activate' in the time domain for comparison. Fig. 2 shows the result for one specific user speaking 'Activate'; the results differ from speaker to speaker but the shape of the curves is identical for different speakers.

The effect of jamming is clearly visible when the jamming signal is placed between 85ms and 180ms. Here the probability score is pushed close to 0 and jamming is successful.

D. Jamming Result

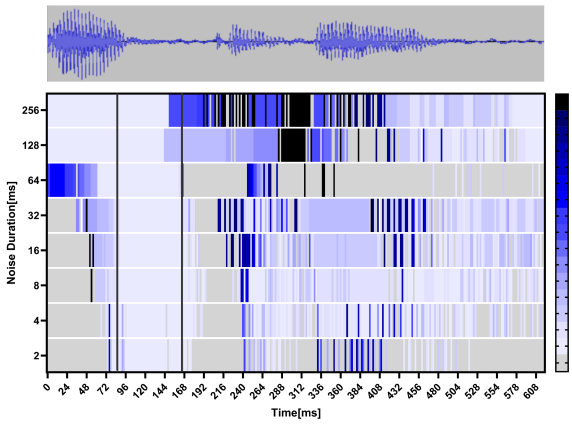


Fig. 3: Wake word detection performance when superimposing white noise over the wake word 'Activate'. The color intensity represents the noise energy level required for successful jamming. Grey represents areas where jamming was not possible. The sensitive region is marked by a box.

Fig. 3 shows the result of jamming the LSTM-WD model trained to recognize 'Activate' using white noise of varying duration, placement, and power in the form of a heat map. The results are shown for one specific speaker saying 'Activate'.

The X-axis represents time and covers the duration of the wake word and the Y-axis represents the different jamming signal duration. Initially, a jamming signal is superimposed over the wake word and the wake word detection is assessed. If the detection fails, the jamming signal moves by 1ms, and the assessment is repeated. If the jamming signal is not strong enough to interrupt the wake word detection, then the jamming signal energy is increased until jamming is successful or the energy limit is reached. The shade of blue indicates the energy of the jamming signal necessary to prevent wake word recognition. Darker shades of blue indicate high energy with the lightest blue representing the lowest used energy corresponding to SNR 0dB (noise power is equal to wake word power). Grey areas show where jamming was impossible.

Fig. 2 shows the highest probability score on the Y-axis while in Fig. 3 we only show jamming success. The line for the 32ms duration jamming signal in Fig. 3 corresponds to Fig. 2 for one power setting (SNR 0dB).

It can be seen in Fig. 3 that the wake word detection is very sensitive to jamming signals. A short jamming signal of 2ms duration with low power corresponding to SNR of 0dB can be sufficient (e.g. between 85ms and 150ms). Many variations of jamming signal duration, strength and power are successful in preventing wake word detection. Jamming success is not always observed for successive signal positioning and areas exist where a slight change in jamming positioning decides on jamming failure or success. However, areas exist where the duration and positioning of the jamming signal do not have to be accurate to prevent wake word detection. For example, jamming signals of all evaluated duration are effective in the area from 85ms to 150ms and we refer to this as *sensitive region*. The region is shown by a black box in Fig. 3 and is defined in the next paragraph.

E. Sensitive Region and Jamming Coverage

We define two metrics to describe jamming effectiveness: *Sensitive Region* R and *Jamming Coverage* C.

A sensitive region R is a time window defined by a start time t_S and an end time t_E with duration $T = t_E - t_S$. Within this time window, jamming is successful for $\tau\%$ of all test cases (placement, duration, power levels). Furthermore, a sensitive region must be longer than threshold T_{min} ($T > T_{min}$) to constitute a valid region. Fig. 3 exhibits one sensitive region R_1 with $T = 164ms - 88ms = 76ms$ for a threshold of $\tau = 99\%$. We use in the following study $\tau = 99\%$ and $T_{min} = 20ms$ to define sensitive regions. An experiment could have multiple sensitive regions or none at all. For each spoken wake word 'Activate' (i.e. different speakers and instances) the regions may vary. The sensitive region is important as it shows an area the wake word is particularly vulnerable. In addition, it is an area where an attacker requires less effort to place a jamming signal as timing constraints on positioning and signal duration are relaxed.

The jamming coverage C is defined as the area of a heat map where jamming is successful in relation to the total area. A jamming-resistant wake word detection should have a jamming coverage of zero percent. Fig. 3 has a jamming coverage of $C = 69.20\%$. Again, the jamming coverage will be different for each spoken instance of the wake word.

V. Adversarial Training

The idea behind adversarial training is to familiarize the algorithm with jamming signals that might be encountered.

A. Adversarial Training Data

Adversarial examples are created by superimposing jamming signals with wake words. Due to the large number of possible jamming signals defined by signal position , duration and power it is not efficient to generate them all and use them

TABLE I: Jamming Coverage C Before and After Adversarial Training LSTM-WD with Different Noises and Wake Words.

Wake Words	Adversarial Training	Interference Signal	
		White Noise	Ping Noise
Activate	X	69.20	72.64
Activate	White Noise	6.82	66
Activate	Babel Noise	7.07	9.80
Alexa	X	30.62	51.33
Alexa	White Noise	3.8	27.86
Alexa	Babel Noise	4.15	10.01
Marvin	X	43.5	50.78
Marvin	White Noise	14.30	14.36
Marvin	Babel Noise	5.26	7.45

all in training. To reduce the number of adversarial examples we only generate examples for sensitive regions. We observe that a sensitive region is always seen between $t_{S'} = 80ms$ and $t_{E'} = 200ms$ for different speakers and we chose this area R' to focus our training effort. By taking into account the different positions, duration and power of noise signals in R' we consider 24000 jamming signals (opposed to 16848000 jamming signals if we would not limit us to region R'). These noise signals are superimposed on the existing samples of thirteen different users speaking the word 'Activate'. Thus, a total of 312000 adversarial samples were created.

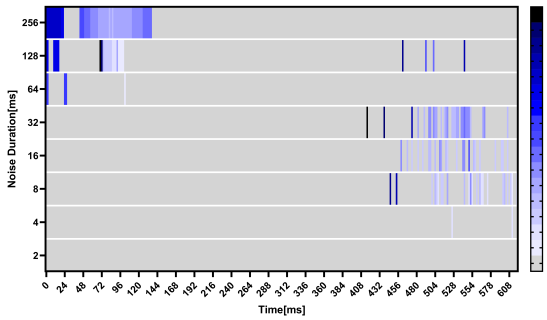
B. Adversarial Training Process

The pre-trained LSTM-WD model for the wake word 'Activate' is taken as the starting point for adversarial training and all model layers are retained. The adversarial examples are split into 20% for validation and 80% for training. The model is trained with a binary cross entropy loss function for 100 epochs with a batch size of 10 and the model generated from the last epoch is selected as the adversarial trained LSTM-WD model. This model is tested against jamming with white noise and the resulting heat map is given in Fig. 4a for one speaker uttering 'Activate'. Only a few regions in the heat map are colored blue and the jamming coverage is reduced from $C = 69.20\%$ (Fig. 3) to $C = 6.82\%$ which shows the success of adversarial training. It has to be noted that sensitive region is now absent.

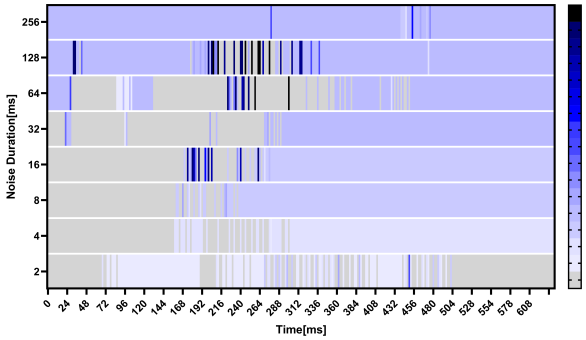
Next, we test the adversarial trained model using ping noise instead of white noise as an interference signal. This noise type is unknown to the LSTM-WD model as it was not included in the training set. The result given in Fig. 4b shows that most regions are colored light blue which shows that adversarial training using white noise is unable to defend against jamming using a ping noise.

C. Resistance to Noise Variation

To make the model more robust against jamming it must be trained with a noise that can cover a range of different noise types. We chose babel noise as a substitute for white noise for this. Other than changing the noise type, the generation of adversarial samples and subsequent model adaptation is performed in the same way as described in the previous section. The experimental results using white noise and ping noise as jamming signals are shown in Fig. 5. Both heat maps have most of the regions colored grey and hence the defense



(a) Jamming using white noise. The attack is considered unsuccessful ($C = 6.82\%$).



(b) Jamming using ping noise. The attack is considered successful ($C = 66.0\%$).

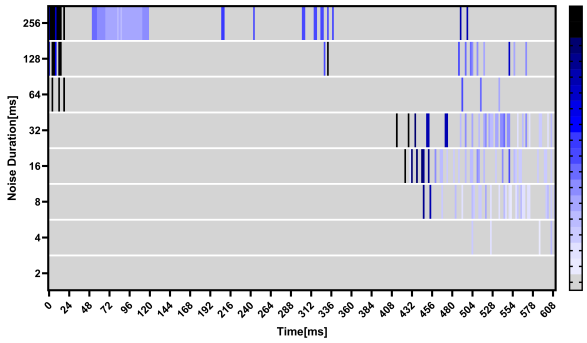
Fig. 4: Impact of jamming using ping and white noises on the performance of wake word detection using the wake word 'Activate' after adversarial training using white noise.

is successful. For ping noise, the adversarial training using babel noise decreased the jamming coverage C to 9.80% from 66.0% (see Table I). It has to be noted that training with a babel noise protects against jamming with different noise types (white noise and ping noise were evaluated). No sensitive region is now present for all noise types.

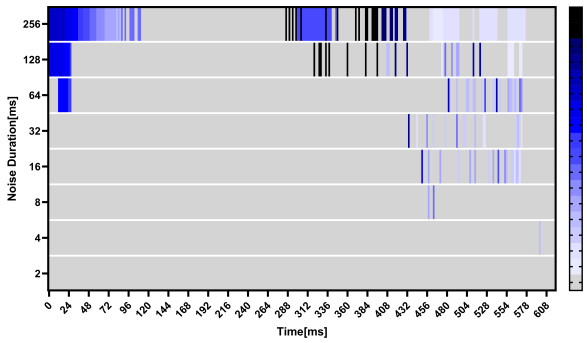
D. Practical Experimental Setup

We evaluated jamming in a practical scenario, using two mini Universal Serial Bus (USB) speakers and a microphone array alongside the wake word detection model. The two USB speakers play wake word and jamming signals separately. The audio from these speakers is recorded over the air using a *ReSpeaker Mic Array v2.0*, and the recorded audio is passed to the wake word detection model to detect wake word.

As shown in Fig. 6, the USB speaker 1 on the left played the wake word and the USB speaker 2 on the right played jamming signals of varying duration ranging from 2ms to 256ms. The USB speakers operate at 48kHz. These USB speakers are positioned equidistantly, 24cm apart from the microphone array, and are arranged on both sides of the microphone array. The *ReSpeaker Mic Array* is equipped with four microphones [17]. The microphone array records the audio signals played by the two speakers using all four microphones and combines them into a single channel. The



(a) Jamming using White noise on the Wake Word 'Activate'. The attack is considered unsuccessful ($C = 7.07\%$).



(b) Jamming using Ping noise on the Wake Word 'Activate'. The attack is considered unsuccessful ($C = 9.80\%$).

Fig. 5: Impact of jamming using white and ping noise on the performance of wake word detection using 'Activate' after adversarial training with babel noise.

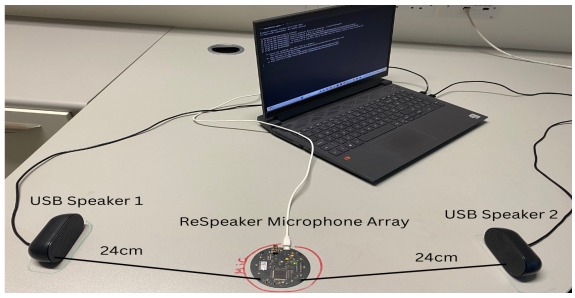
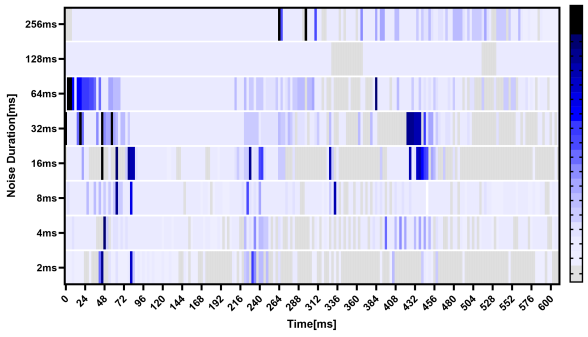
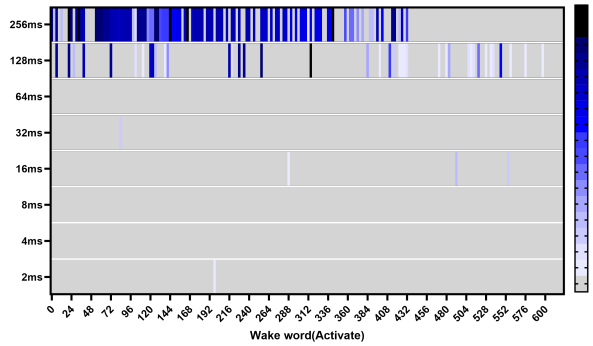


Fig. 6: Placement of devices in our experimental setup.

recorded mono audio is then provided as input to the wake word detection model for detecting the presence of wake words. The experimental setup is placed in a room measuring 3.8m in length and 3.4m in width. The wake word used in the experiment is 'Activate'. The jamming noise is played through USB Speaker 2 with increasing delay, shifting the interference signal progressively over the wake word. In instances where the jamming signal is unsuccessful, we increase the noise in steps until a threshold is reached (70 times greater than the initial jamming energy, corresponding to the practical limit of the speaker's sound output capacity).



(a) Jamming using Ping Noise on the Wake Word Activate in a practical experimental setup. The attack is considered successful ($C = 74.3389\%$).



(b) Jamming using Ping Noise on the Wake Word Activate in a practical experimental setup. The attack is considered unsuccessful ($C = 8.71\%$).

Fig. 7: Impact of jamming in a practical setting using ping noise on the performance of wake word detection using 'Activate' before and after adversarial training with babel noise.

We conducted experiments using ping noise as jamming signal on the wake word model before adversarial training and compared it with the model that underwent adversarial training with babble noise. The results are shown in Fig. 7.

It can be noticed from the heat maps that the jamming coverage has been reduced from 74.3389% to 8.71% using adversarial training. The experiment confirms that our adversarial training approach is useful in practical settings.

E. Generalization

We have also evaluated the outlined adversarial training with other wake words such as 'Alexa' and 'Marvin'. The behavior of the wake word detection under jamming before and after adversarial training is similar when using different wake words. Differences are observed in terms of the positioning of sensitive regions which are word dependent. Fig. 8a and Fig. 8b show the effect of jamming with ping noise on the wake word 'Alexa' before and after adversarial training using babel noise, it can be observed that the jamming was successful before the adversarial training of the model as shown in Fig. 8a with jamming coverage of 51.33% and was clearly unsuccessful after the training as shown in Fig. 8b with a jamming coverage of 10.01%.

Similarly Fig. 8c and Fig. 8d show the effect of jamming with ping noise on the wake word 'Marvin' before and after the adversarial training using babel noise. It can be observed from Fig. 8c and Fig. 8d, that the jamming coverage has reduced from 50.78% to 7.45% after adversarial training.

The jamming coverage C for different wake words is displayed in Table I and the values for ping noise are higher than white noise for all scenarios. Thus ping noise is a more successful candidate for jamming than white noise. All wake words exhibit a common behavior towards adversarial training. Adversarial training reduces the jamming coverage to a very low value for attacks using white noise considering all wake words. But for jamming with a ping noise only training with babel noise provides sufficient protection.

We also evaluated this approach with another wake word detection algorithm *MHatt-RNN* and the model's output before and after adversarial training is similar to that of the *LSTM-WD* algorithm. The positions of sensitive areas in the wake word are different, but the areas still exist, that can be identified and used for adversarial training. The wake word chosen for the experiment is 'Marvin' and the results of jamming, before and after the training are displayed in Fig. 9.

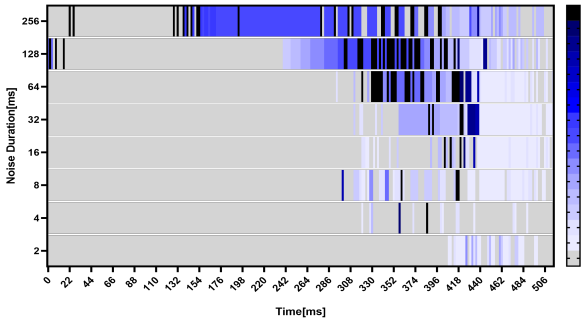
By comparing Fig. 8c and Fig. 9a *LSTM-WD* outperforms *MHatt-RNN* in terms of jamming resistance without adversarial training. After adversarial training *MHatt-RNN* performs better (see Fig. 8d and Fig. 9b. However, in both cases no sensitive region is present and jamming coverage is low.

We also used an Echo device to understand wake word jamming on commercial devices. As this commercial algorithm implementation is not accessible we evaluated only jamming (in part as automation is not possible and every experiment outcome must be recorded manually) but not adversarial training. Our experimentation confirmed that the Echo device is susceptible to jamming. When noise was introduced within the beginning 40ms–50ms of the wake word Alexa, Echo failed to respond. A sensitive area emerges, however, we found that interference signals need to be longer or stronger than in the aforementioned experiments. We assume that this is the case as the Echo device also includes advanced hardware features for noise cancellation not present in our experimentation. In summary, the Echo device exhibits similar behavior as the investigated algorithms and we assume that the adversarial training approach will also be useful here.

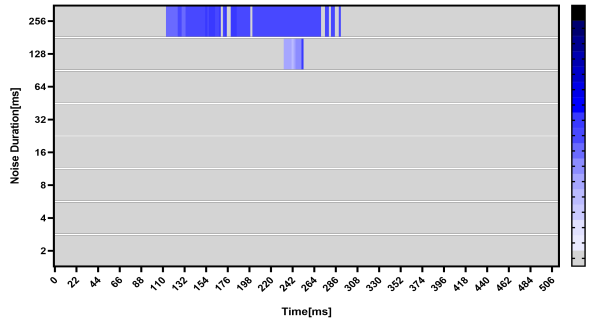
VI. Conclusion

PVAs have become ubiquitous, finding increasing application in critical scenarios. We therefore need to consider DoS attacks on these systems that use the acoustic channel and not only classical DoS attacks targeting the PVA's Internet connection. Most IoT devices have microphones included and it is easy for an attacker to use the acoustic channel once they have obtained control of any nearby device with a speaker. Our work shows how susceptible PVA systems are to DoS attacks and we outline a way forward to address this issue.

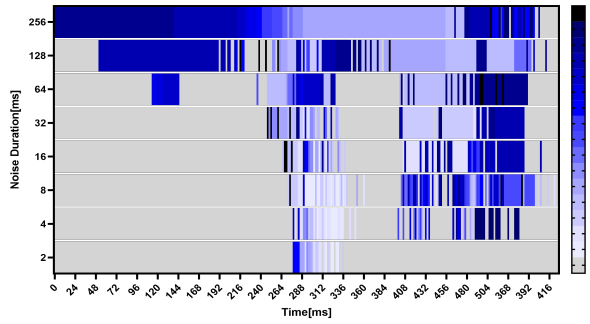
We have shown how adversarial training can protect wake word recognition from jamming attacks. We have shown that



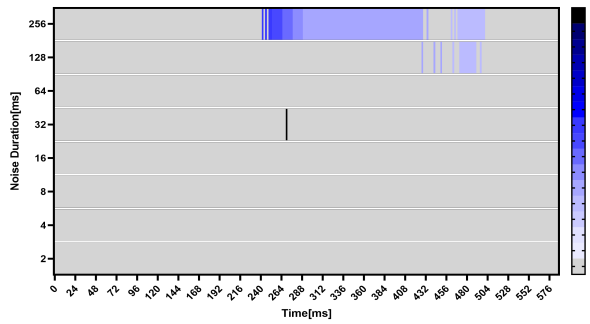
(a) Jamming using ping noise on the wake word "Alexa" before adversarial training. The attack is considered successful ($C = 51.33\%$).



(b) Jamming using ping noise on the wake word "Alexa" after adversarial training. The attack is considered unsuccessful ($C = 10.01\%$).

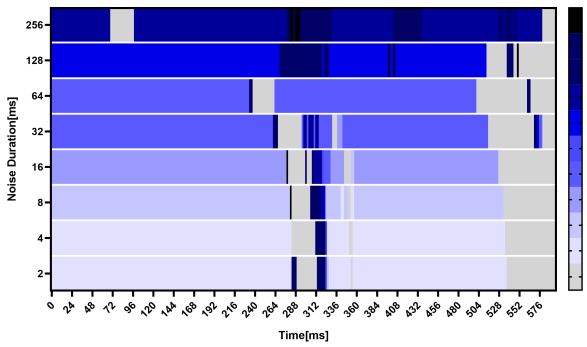


(c) Jamming using ping noise on the wake word 'Marvin' before adversarial training. The attack is considered successful ($C = 50.78\%$).

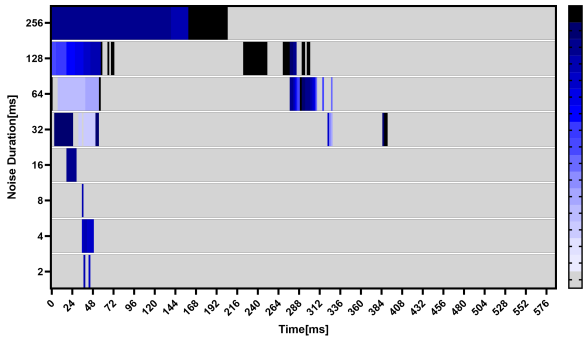


(d) Jamming using ping noise on the wake word 'Marvin' after adversarial training. The attack is considered unsuccessful ($C = 7.45\%$).

Fig. 8: Impact of jamming using ping noise on the performance of wake words 'Alexa' and 'Marvin' before and after adversarial training with babel noise.



(a) Jamming using ping noise on the wake word 'Marvin' before adversarial training. The attack is considered successful ($C = 85.27\%$).



(b) Jamming using ping noise on the wake word 'Marvin' after adversarial training. The attack is considered unsuccessful ($C = 10.48\%$).

Fig. 9: *MHatt-RNN* model: Impact of jamming using ping noise on the performance of wake word detection of the word 'Marvin' before and after adversarial training with babel noise.

(i) a well-placed short jamming signal of only 2ms can disrupt wake word detection; (ii) adversarial training can significantly improve the robustness of wake word detection; (iii) babel noise is a suitable signal for adversarial training and that training only has to focus on sensitive regions. We executed jamming attack on the wake word detection model using an over-the-air experimental setup before and after adversarial training and the results closely match the simulated offline approach. We demonstrated usability using two wake word detection algorithms and different wake words.

ACKNOWLEDGEMENT

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 19/FFP/6775 and 13/RC/2077_P2.

REFERENCES

[1] E.-W. V. Lo and P. Green, "Development and evaluation of automotive speech interfaces: Useful information from the human factors and the related literature," *International Journal of Vehicular Technology*, vol. 2013, pp. 1–13, January 2013.

[2] R. H. Taylor, A. Menciassi, G. Fichtinger, P. Fiorini, and P. Dario, "Medical robotics and computer-integrated surgery," in *Springer Handbook of Robotics*, ser. Springer Handbooks, B. Siciliano and O. Khatib, Eds., 2016, pp. 1657–1684.

[3] D. T. Williamson, M. H. Draper, G. L. Calhoun, and T. Barry, "Commercial speech recognition technology in the military domain: Results of two recent research efforts," *Int. J. Speech Technol.*, vol. 8, no. 1, pp. 9–16, 2005. [Online]. Available: https://doi.org/10.1007/s10772-005-4758-6

[4] K. Sun, C. Chen, and X. Zhang, ""alexa, stop spying on me!": speech privacy protection against voice assistants," in *SenSys '20: The 18th ACM Conference on Embedded Networked Sensor Systems, Virtual Event, Japan, November 16-19, 2020*, J. Nakazawa and P. Huang, Eds. ACM, 2020, pp. 298–311. [Online]. Available: https://doi.org/10.1145/3384419.3430727

[5] P. Cheng, I. E. Bagci, J. Yan, and U. Roedig, "Towards reactive acoustic jamming for personal voice assistants," in *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security*, Toronto, ON, Canada, October 2018, pp. 12–17.

[6] J. Li, S. Qu, X. Li, J. Szurley, J. Z. Kolter, and F. Metze, "Adversarial music: Real world audio adversary against wake-word detection system," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., Vancouver, Canada, December 2019, pp. 11 908–11 918.

[7] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, Dallas, USA, November 2017, pp. 103–117.

[8] H. Lim, Y. Kim, and H. Kim, "Cross-informed domain adversarial training for noise-robust wake-up word detection," *IEEE Signal Processing Letters*, vol. 27, pp. 1769–1773, September 2020.

[9] O. Rybakov, N. Kononenko, N. Subrahmanya, M. Visontai, and S. Laurenzo, "Streaming keyword spotting on mobile devices," in *INTERSPEECH 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event*, H. Meng, B. Xu, and T. F. Zheng, Eds., Shanghai, China, October 2020, pp. 2277–2281.

[10] K. Supriya, A. Divya, G. R. Sai, and B. Vinodkumar, "Trigger word recognition using lstm," *INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT)*, vol. 09, no. 6, pp. 263–270, June 2020.

[11] Y. Gao, Y. Mishchenko, A. Shah, S. Matsoukas, and S. Vitaladevuni, "Towards data-efficient modeling for wake word spotting," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona,Spain, May 2020, pp. 7479–7483.

[12] Q. Yu and R. Zhou, "Wake word detection based on res2net," *arXiv preprint arXiv:2209.15296*, 2022.

[13] S. Ö. Arik, M. Kliegl, R. Child, J. Hestness, A. Gibiansky, C. Fougner, R. Prenger, and A. Coates, "Convolutional recurrent neural networks for small-footprint keyword spotting," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, F. Lacerda, Ed. ISCA, 2017, pp. 1606–1610. [Online]. Available: http://www.isca-speech.org/archive/Interspeech_2017/abstracts/1737.html

[14] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," *ArXiv e-prints*, Apr. 2018. [Online]. Available: https://arxiv.org/abs/1804.03209

[15] P. Uglow. (2008) Ping Noise :part of the pack Jingles and pings. [Online]. Available: https://freesound.org/people/BristolStories/sounds/51702/

[16] D. I. S. Pigeon. (2020) The Babel Noise Generator. [Online]. Available: https://mynoise.net/NoiseMachines/babbleNoiseGenerator.php

[17] S. Mischie and G. Găspăresc, "Respeaker mic array 2.0 for speech processing algorithms," in *2020 International Symposium on Electronics and Telecommunications (ISETC)*, 2020, pp. 1–4.